PPSi – A Free Software PTP Implementation

P. Fezzardi, M. Lipiński, A. Rubini, A. Colosimo

CERN | Warsaw University of Technology | Università degli Studi di Pavia

ISPCS2014
25th September 2014
Austin, Texas

Outline

Outline

White Rabbit extension to PTP

- Next-generation CERN control and timing

White Rabbit extension to PTP

- Next-generation CERN control and timing
- Open Hardware and Open Software

White Rabbit extension to PTP

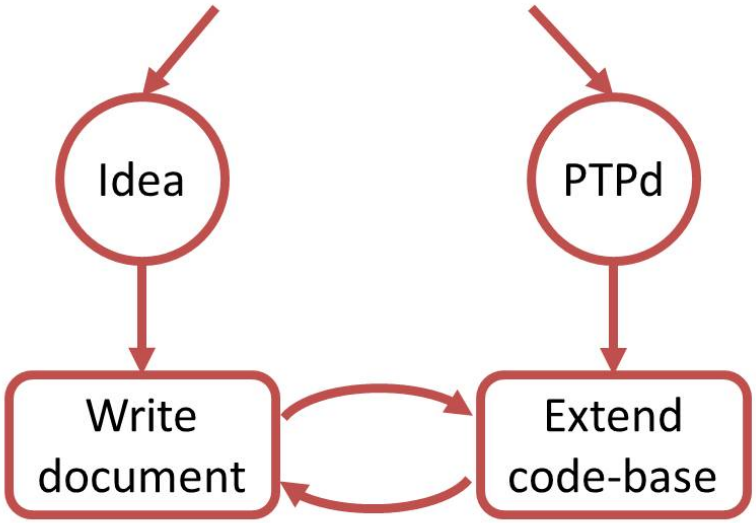
- Next-generation CERN control and timing
- Open Hardware and Open Software
- PTP extension defined as a Profile

White Rabbit extension to PTP

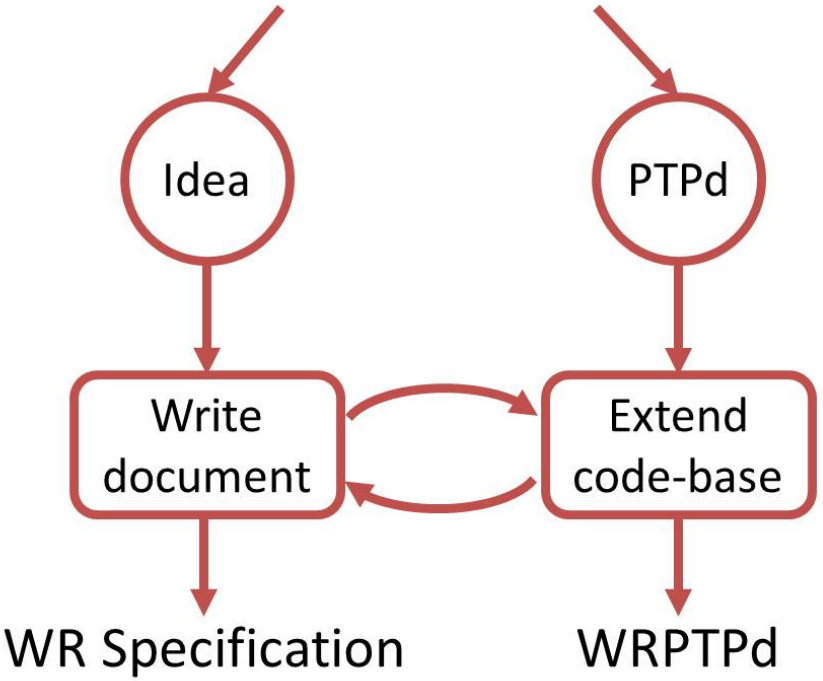
- Next-generation CERN control and timing
- Open Hardware and Open Software
- PTP extension defined as a Profile
- Synchronization:
  - **sub-nanosecond accuracy**
  - **tens of picoseconds precision**

White Rabbit extension to PTP

White Rabbit extension to PTP



White Rabbit extension to PTP

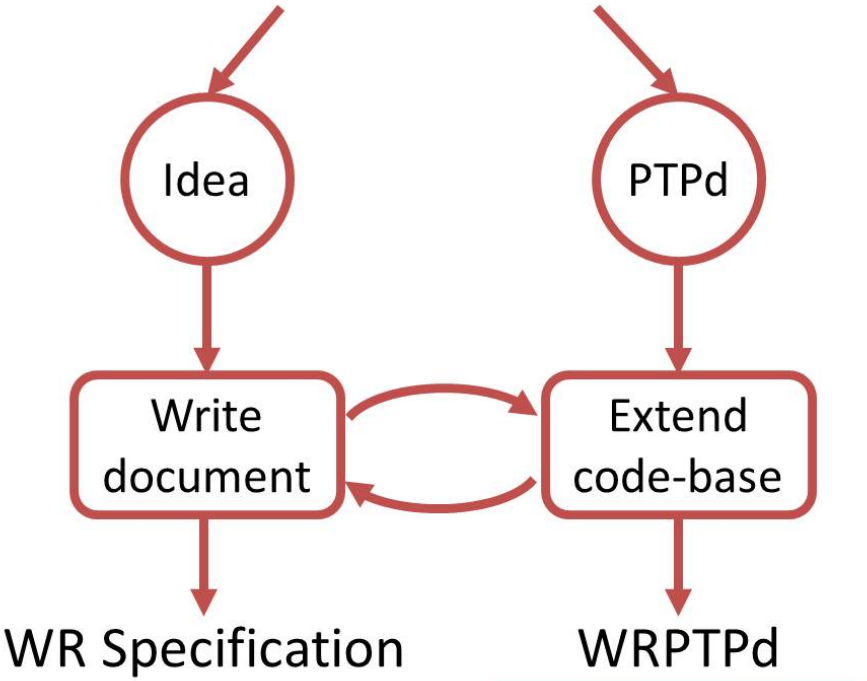



White Rabbit Specification:
Draft for Comments

July 2011

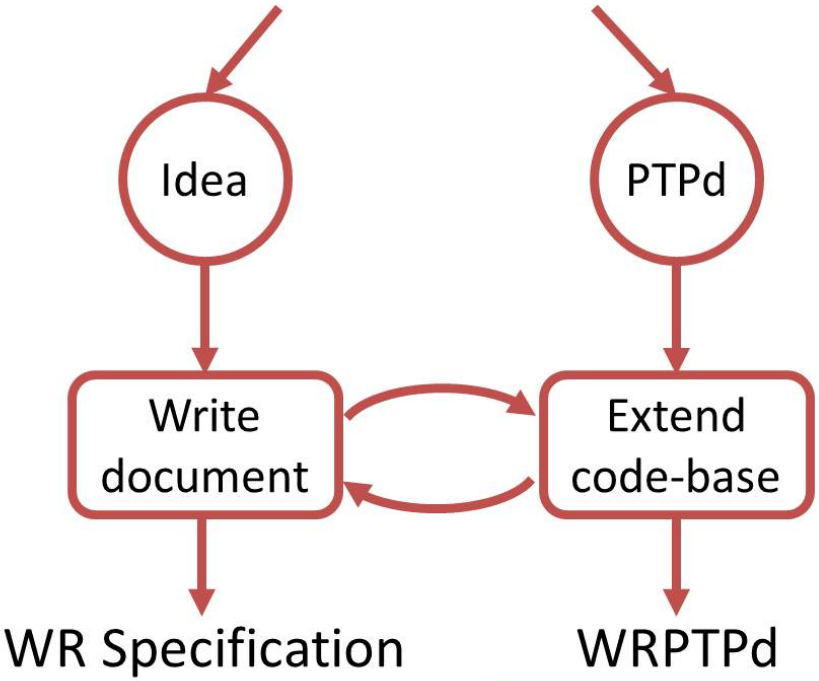
- Unix
- Embedded Linux
- Soft-core in FPGA

White Rabbit extension to PTP




White Rabbit Specification:
Draft for Comments

White Rabbit extension to PTP



Maintain and debug endlessly

Redesign !!!

PTP Ported To Silicon (PPSi)

- Portable
- Modular
- Extensible
- Free software

PTP Ported To Silicon (PPSi)

- Portable
- Modular
- Extensible
- Free software
- Maintainable
- Beautiful code
- Easy to debug

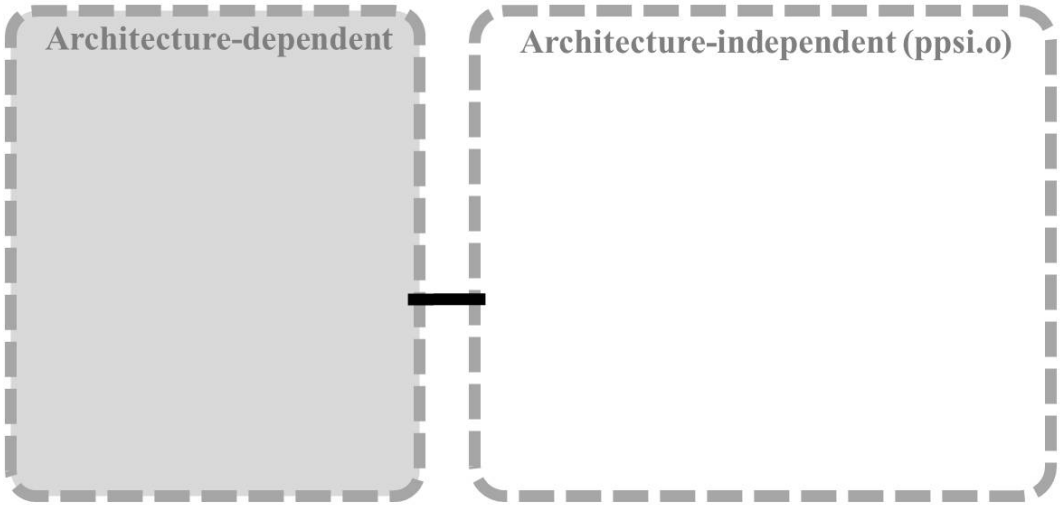
In short ...

A well-designed and well-coded application that we are proud of.

Outline

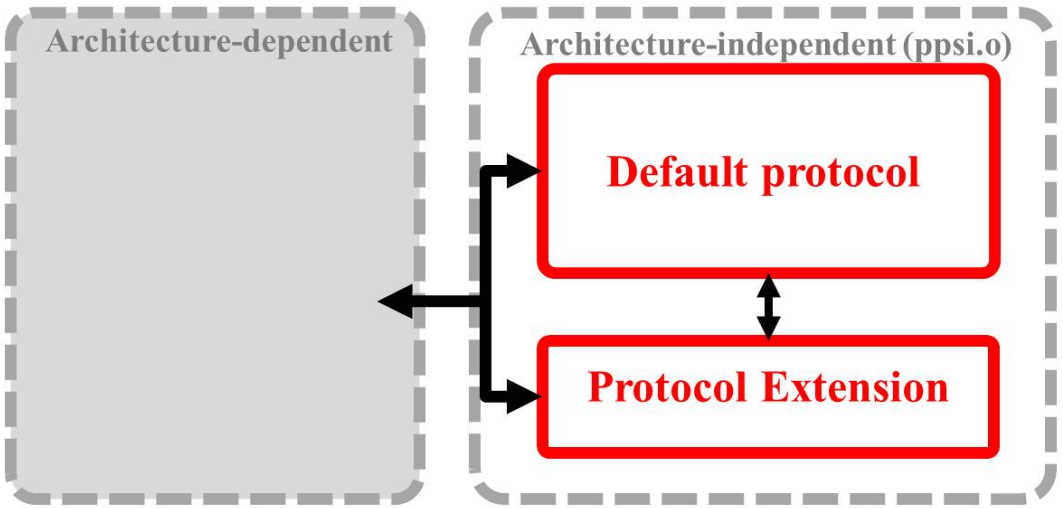
PPSi design overview

- Arch-independent
- Arch-dependent



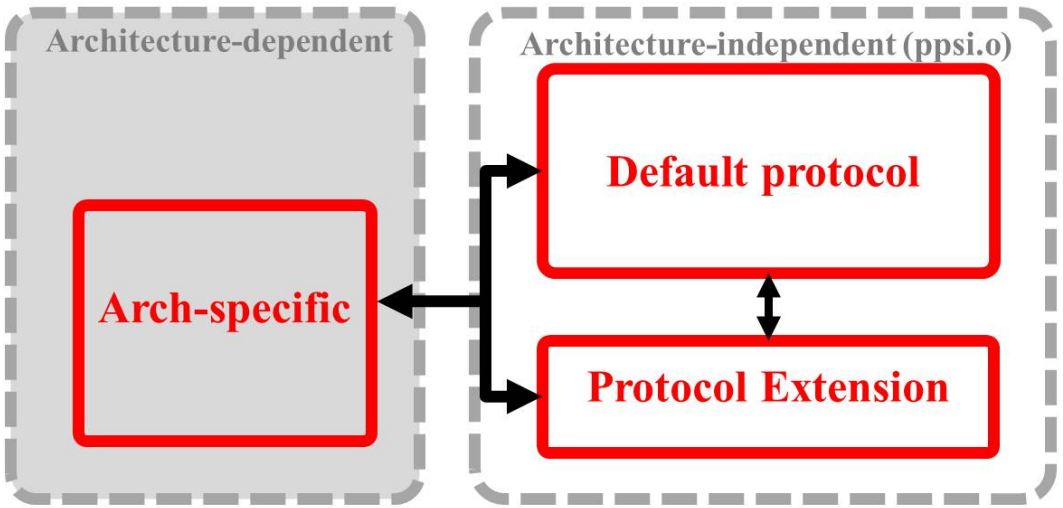
PPSi design overview

- Arch-independent
  - self-containment
  - single entry point
  - non-blocking execution
  - library-like properties
- Arch-dependent



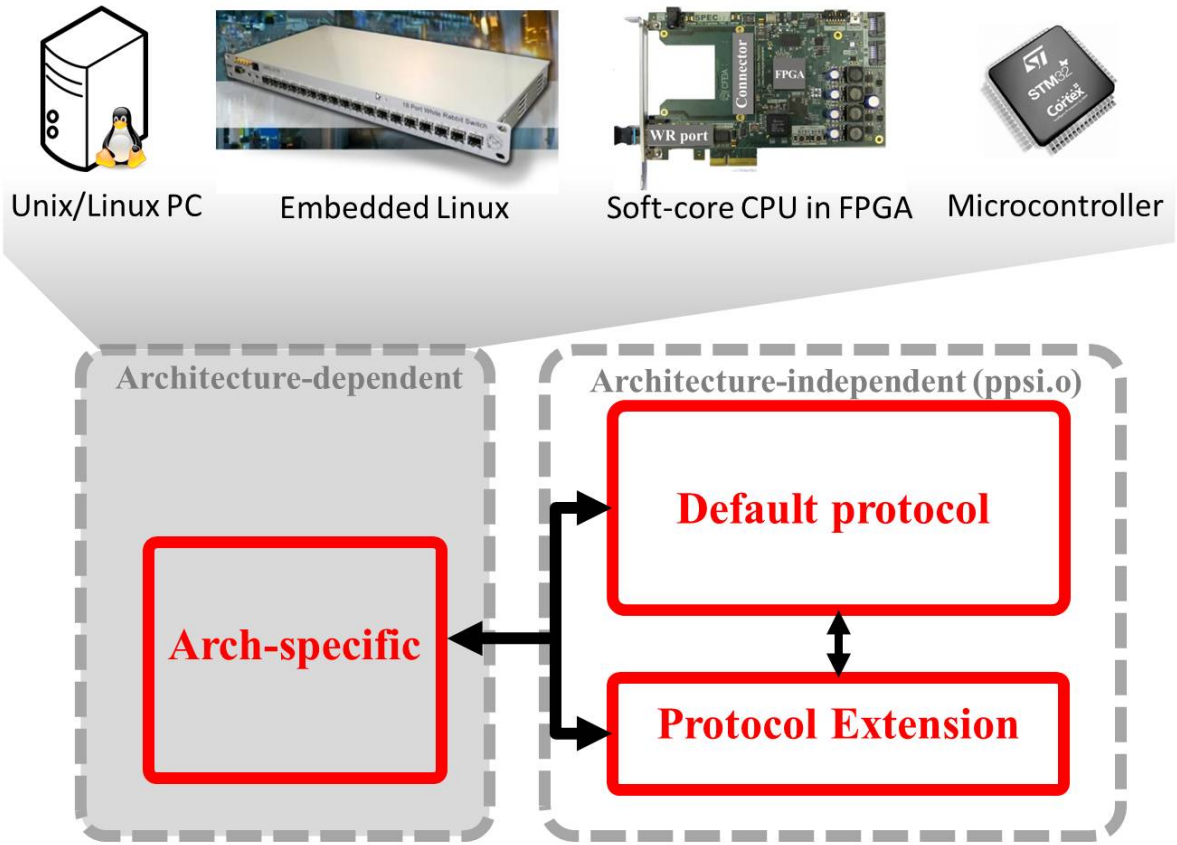
PPSi design overview

- Arch-independent
  - self-containment
  - single entry point
  - non-blocking execution
  - library-like properties
- Arch-dependent
  - network & time I/F



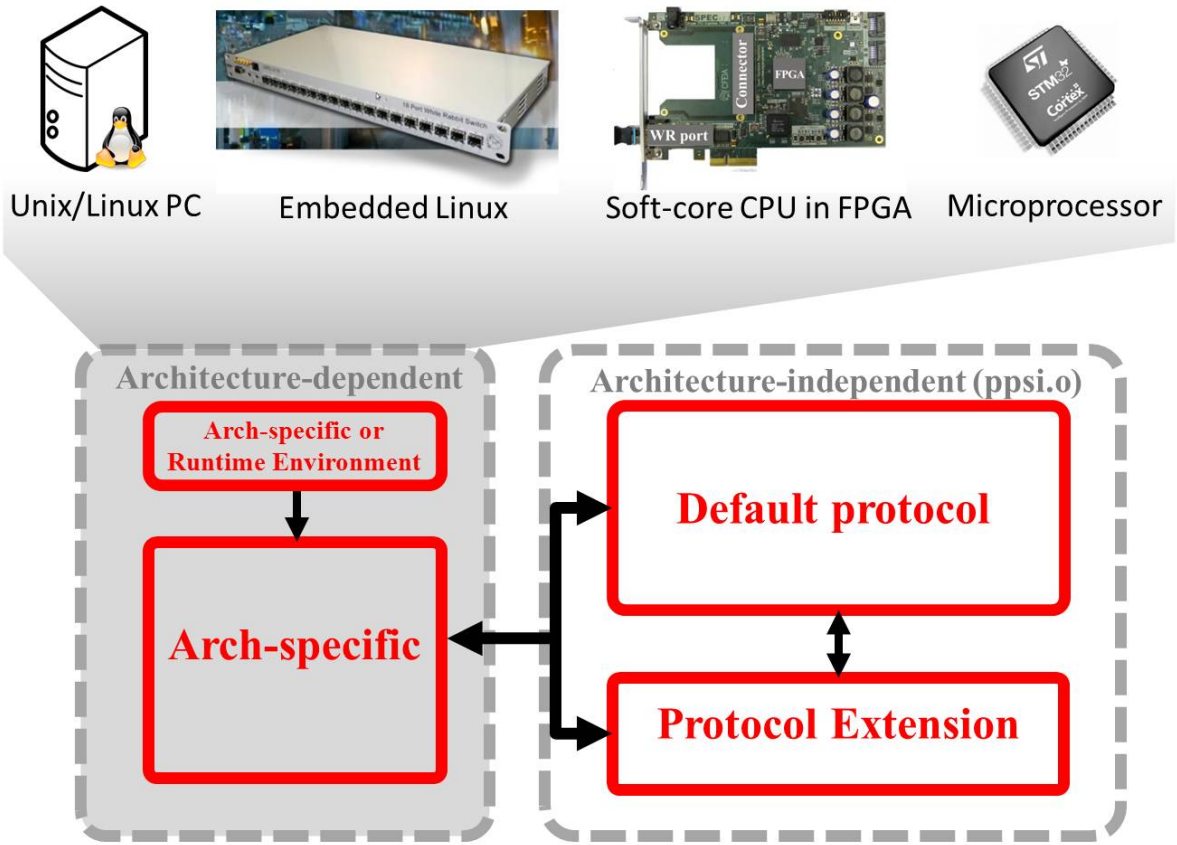
PPSi design overview

- Arch-independent
  - self-containment
  - single entry point
  - non-blocking execution
  - library-like properties
- Arch-dependent
  - network & time I/F



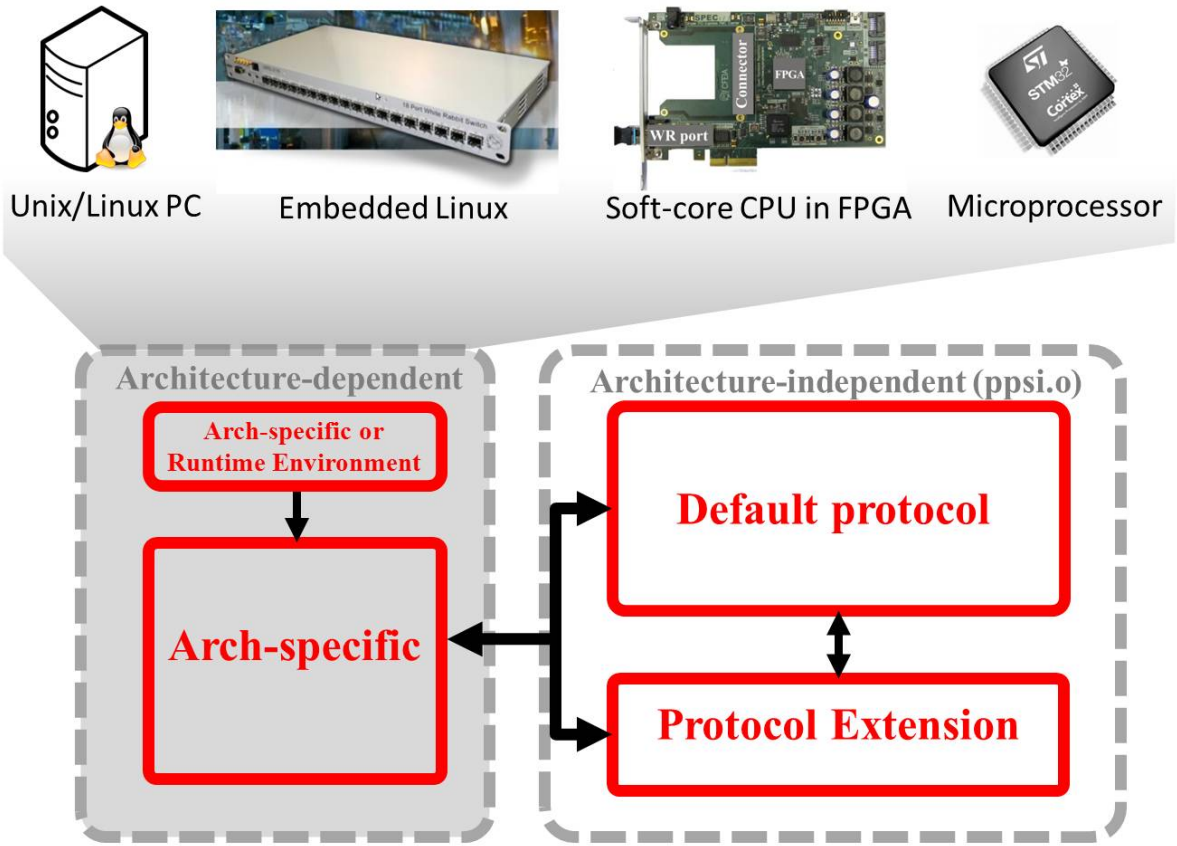
PPSi design overview

- Arch-independent
  - self-containment
  - single entry point
  - non-blocking execution
  - library-like properties
- Arch-dependent
  - network & time I/F
  - main function or real-time loop



PPSi design overview

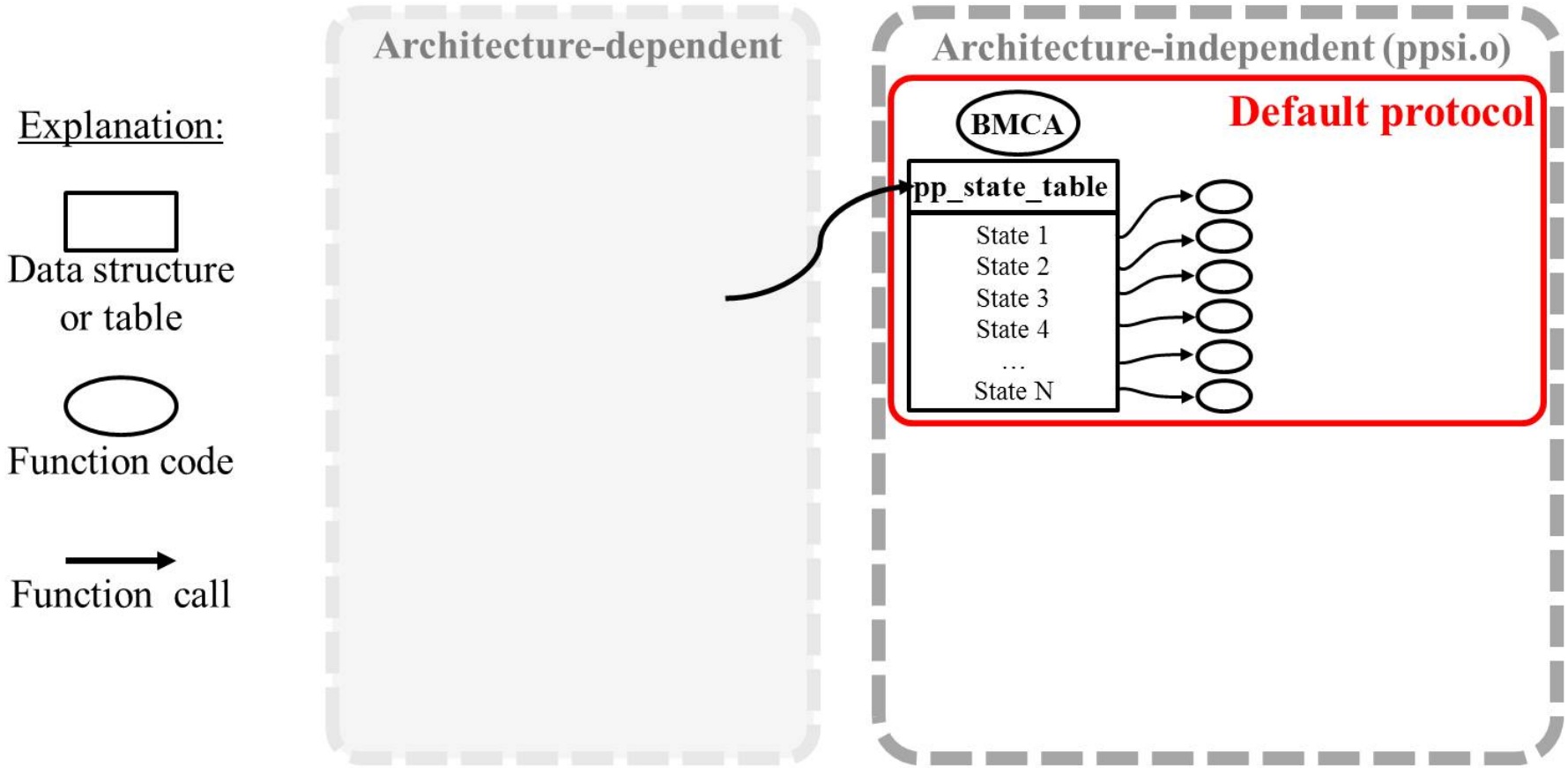
- Arch-independent
  - self-containment
  - single entry point
  - non-blocking execution
  - library-like properties
- Arch-dependent
  - network & time I/F
  - main function or real-time loop



Reflect the PTP layered approach

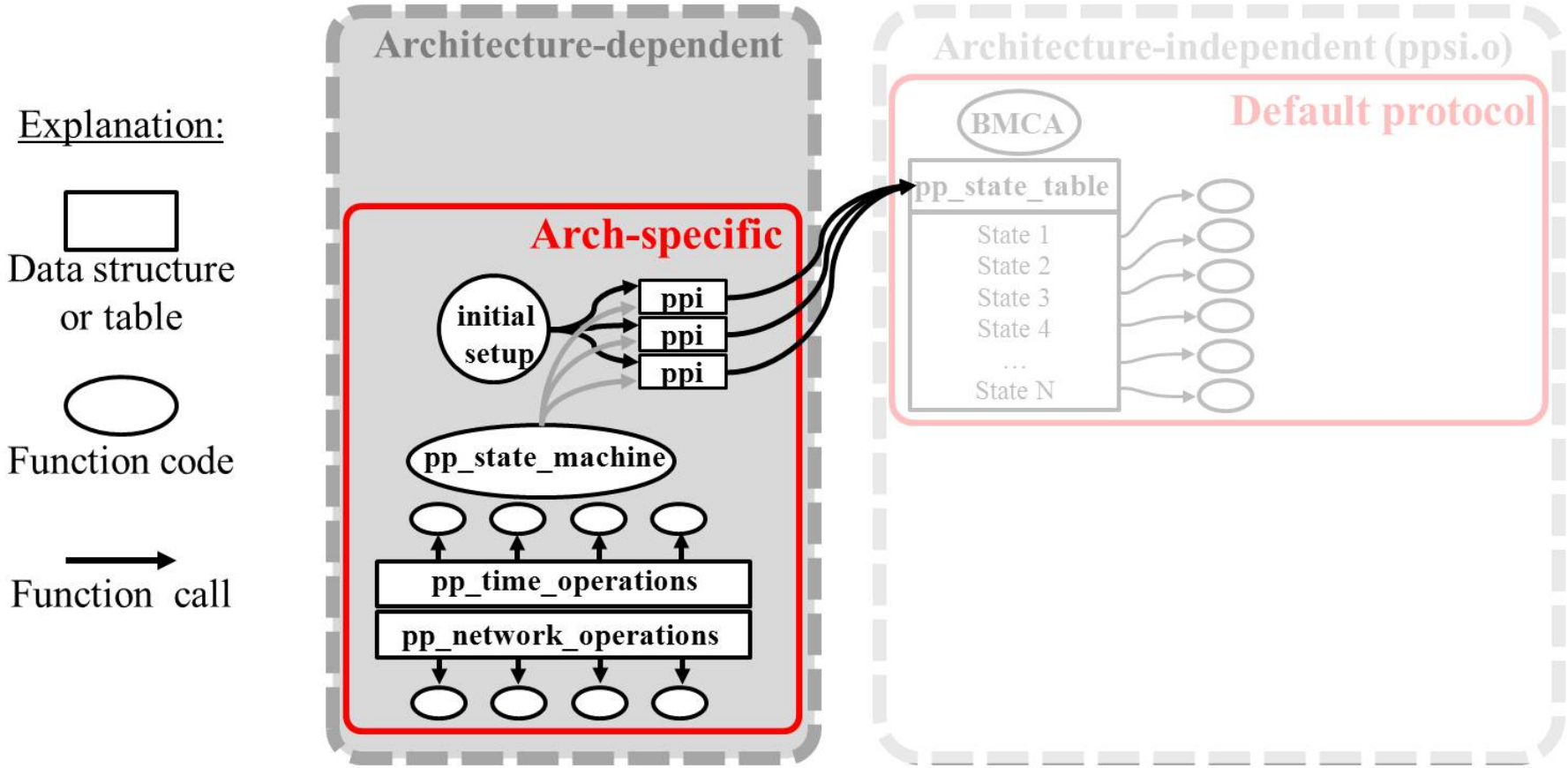
Default protocol code - shared by all architectures

- **Implements**: BMCA, message handling, state machine
- **Executes**: immediately, message– or timeout–triggered
- **Returns**: re-entry delay



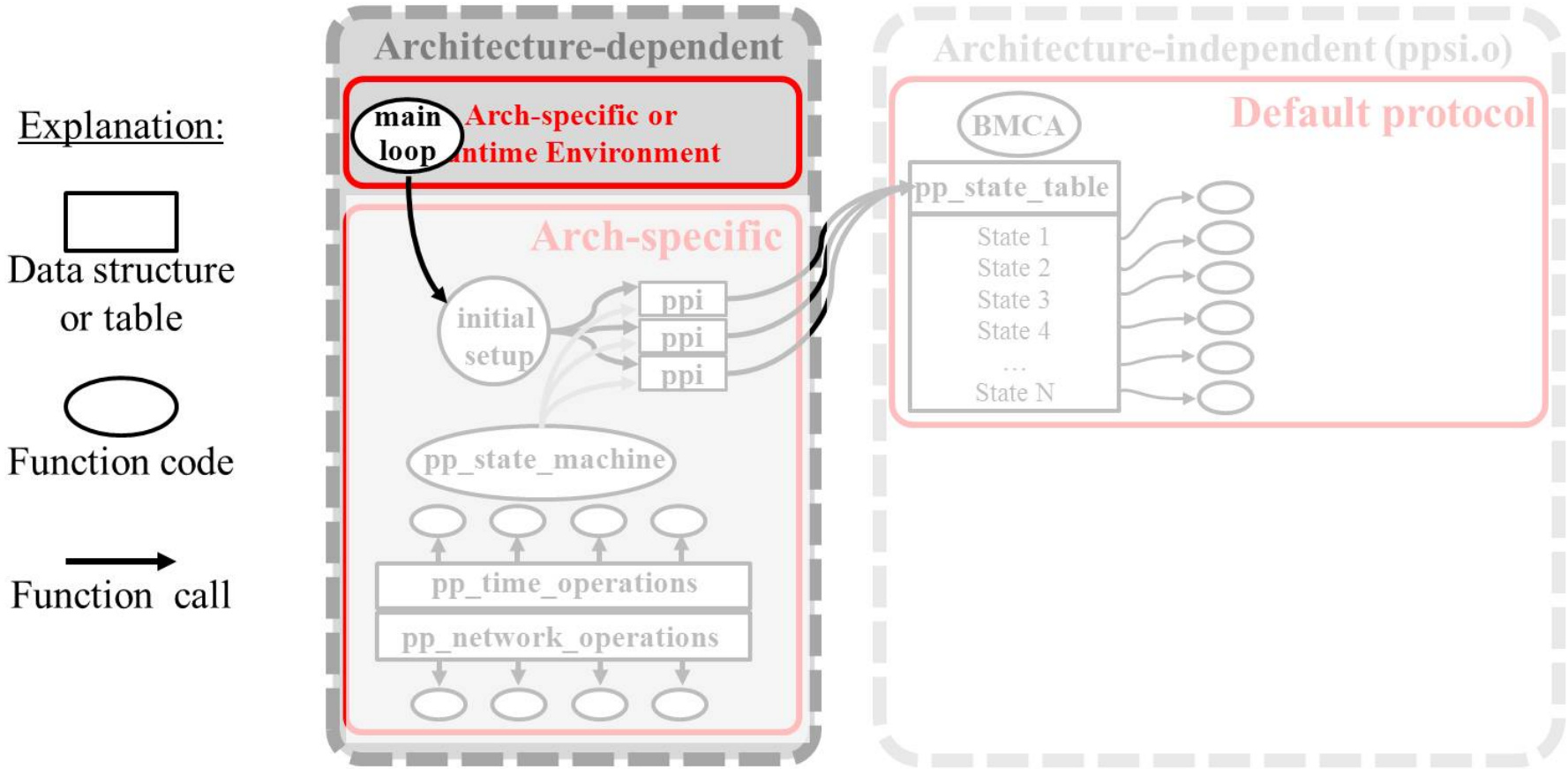
System architecture specific code

- **PPSi instance ppi**: per-port config & runtime data
- **Network operations**: arch– & mapping–specific functions
- **Time operations**: arch– & hardware–specific functions



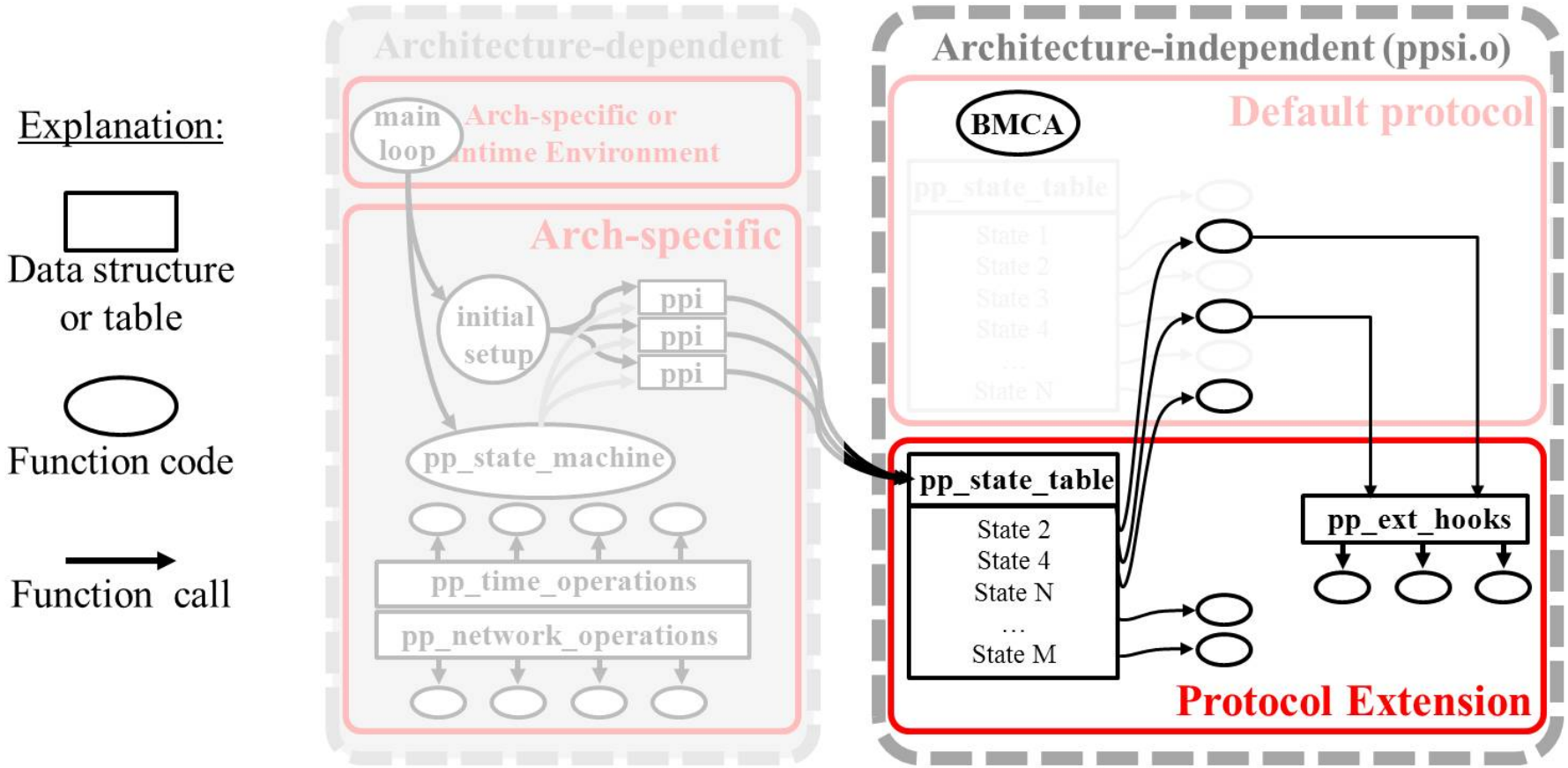
Runtime environment specific code

- **Hosted environment**: main loop of daemon, library access
- **Freestanding environment**: no main function & no library, real-time loop, library-like usage



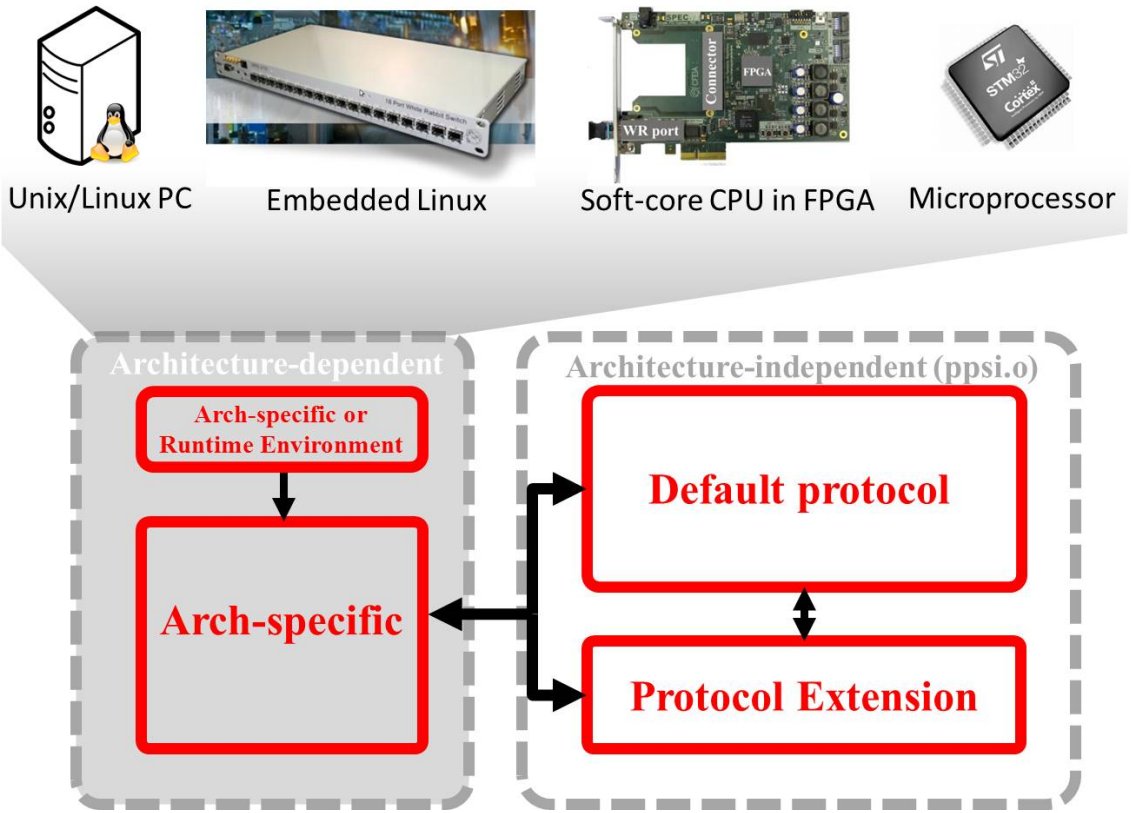
Extension specific code

- **Defines** custom states
- **Redefines** table with calls to default & custom states
- **Defines** hooks in default protocol code



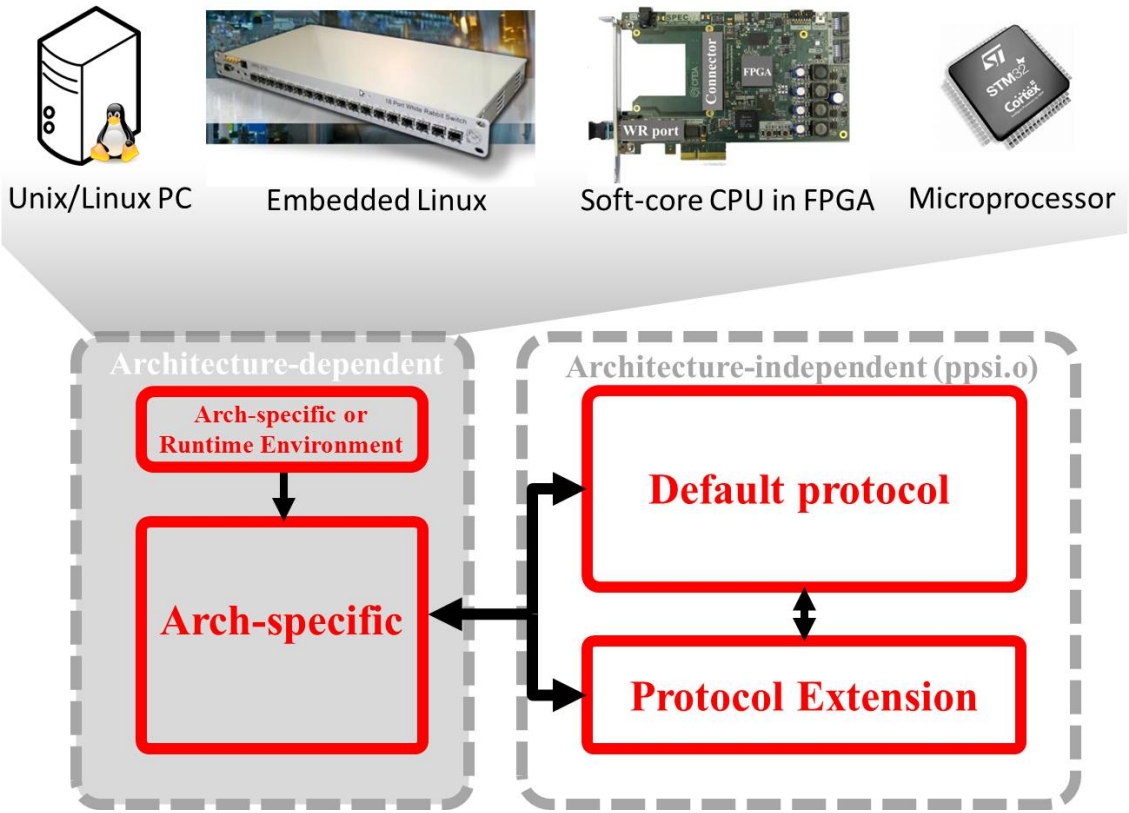
Portability: supported “arch”

- arch–unix
  - Linux with GNU libraries



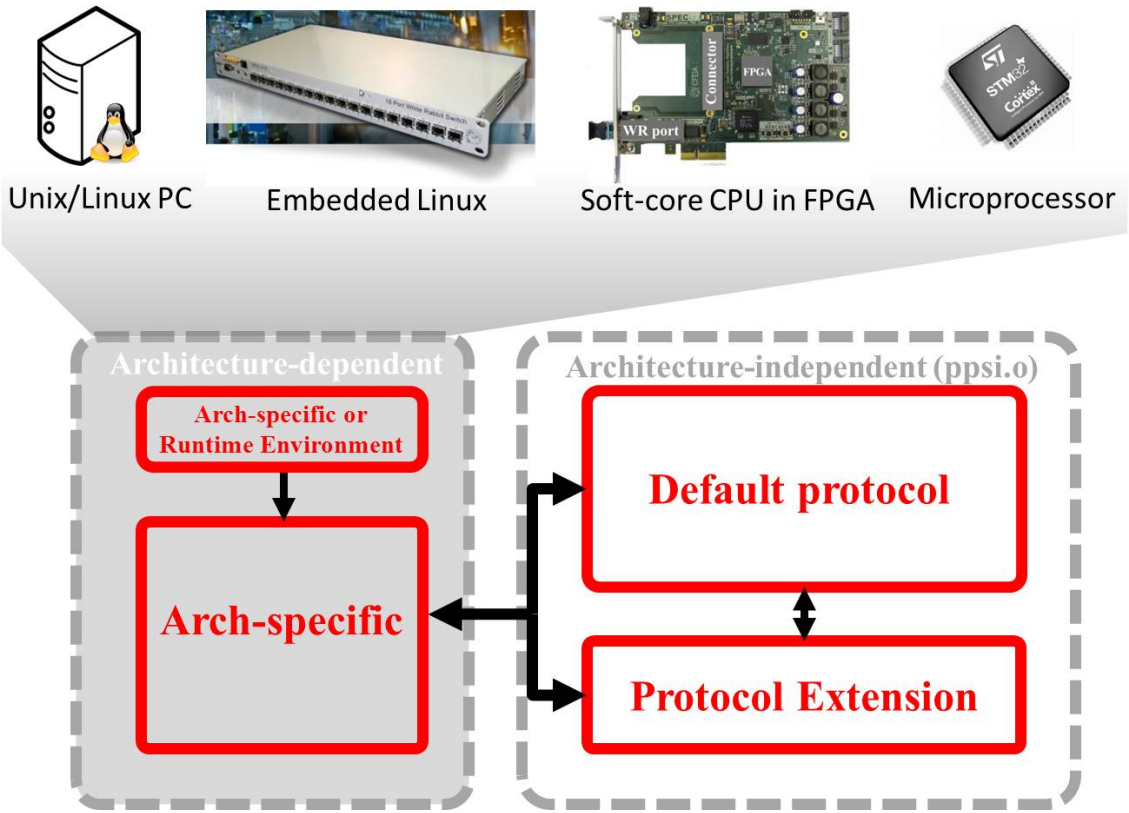
Portability: supported “arch”

- arch–unix
  - Linux with GNU libraries
- arch–wrs
  - embedded Linux
  - hardware timestamping



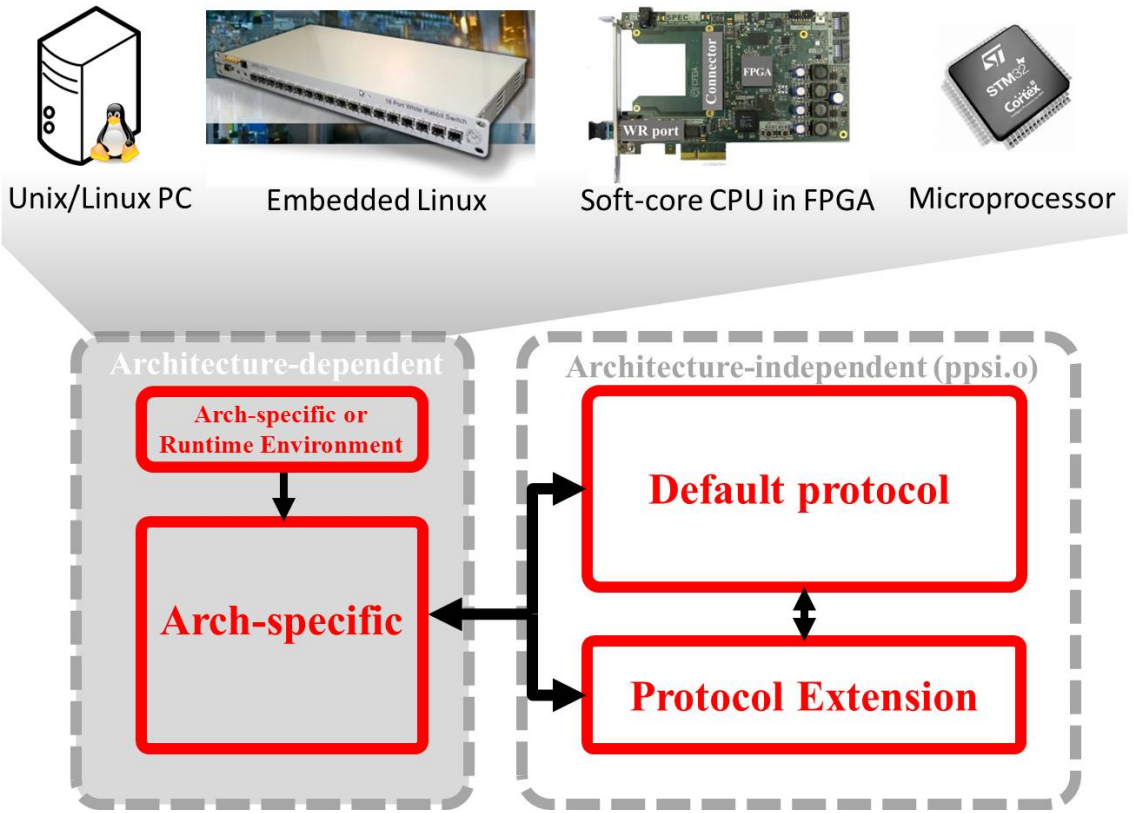
Portability: supported “arch”

- arch–unix
  - Linux with GNU libraries
- arch–wrs
  - embedded Linux
  - hardware timestamping
- arch–wrpc
  - soft–core CPU in FPGA
  - no libraries, Real-Time loop



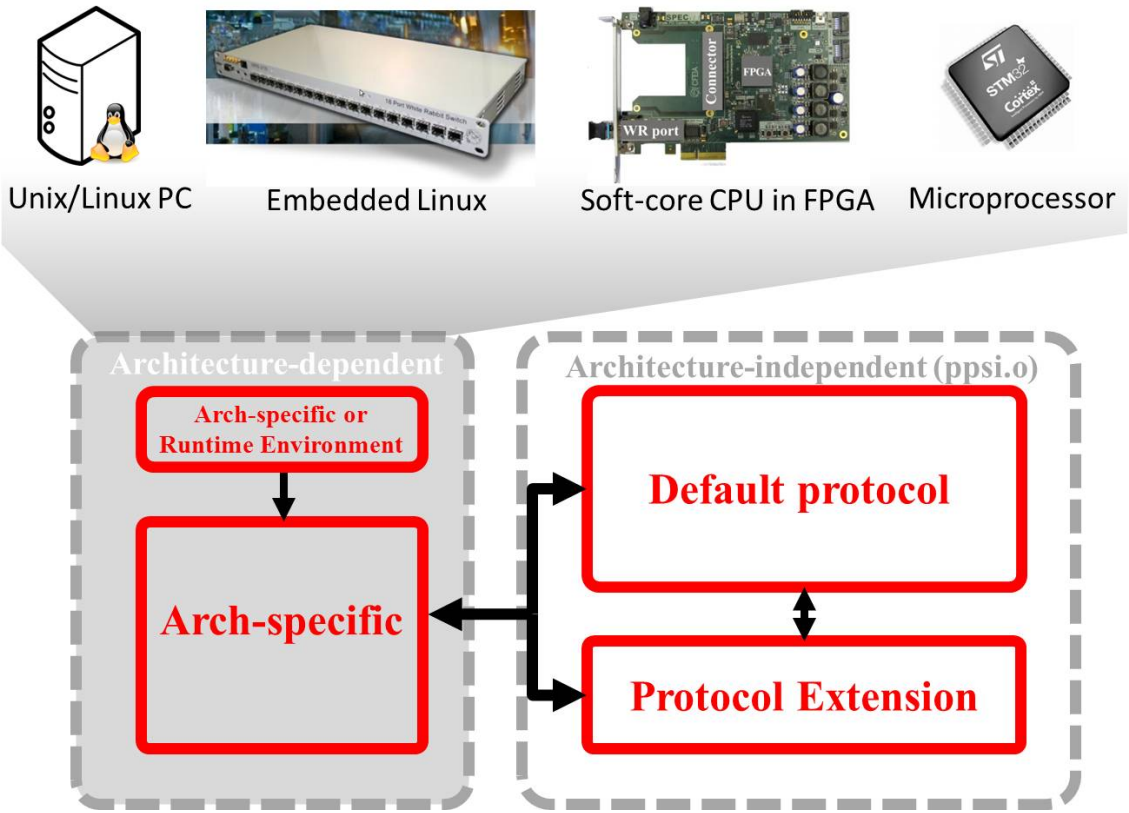
Portability: supported “arch”

- arch–unix
  - Linux with GNU libraries
- arch–wrs
  - embedded Linux
  - hardware timestamping
- arch–wrpc
  - soft–core CPU in FPGA
  - no libraries, Real-Time loop
- arch-bare–{i386, x86_64}
  - Linux build
  - no dependency on libraries



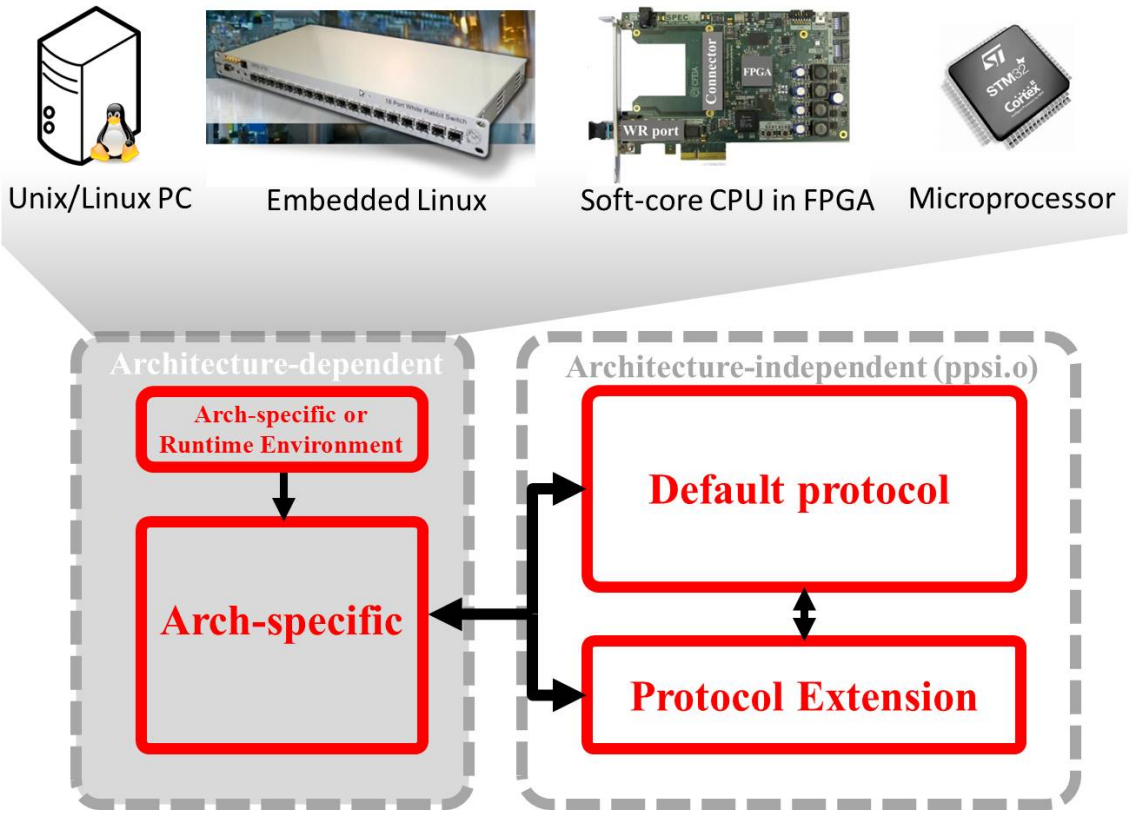
Portability: supported “arch”

- arch–unix
  - Linux with GNU libraries
- arch–wrs
  - embedded Linux
  - hardware timestamping
- arch–wrpc
  - soft–core CPU in FPGA
  - no libraries, Real-Time loop
- arch-bare–{i386, x86_64}
  - Linux build
  - no dependency on libraries
- arch ongoing/independent



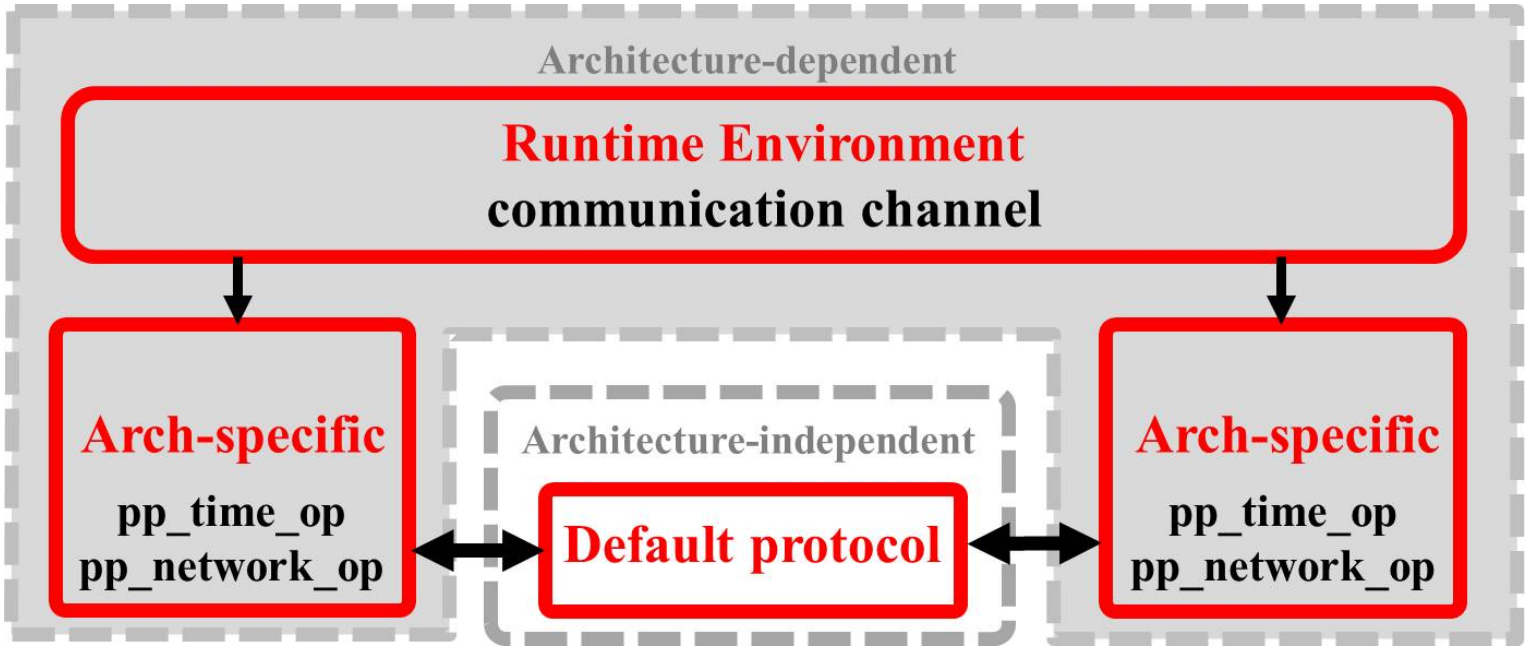
Portability: supported “arch”

- arch–unix
  - Linux with GNU libraries
- arch–wrs
  - embedded Linux
  - hardware timestamping
- arch–wrpc
  - soft–core CPU in FPGA
  - no libraries, Real-Time loop
- arch-bare–{i386, x86_64}
  - Linux build
  - no dependency on libraries
- arch ongoing/independent
- arch-sim
  - hosted on single Unix PC
  - simulation of two nodes



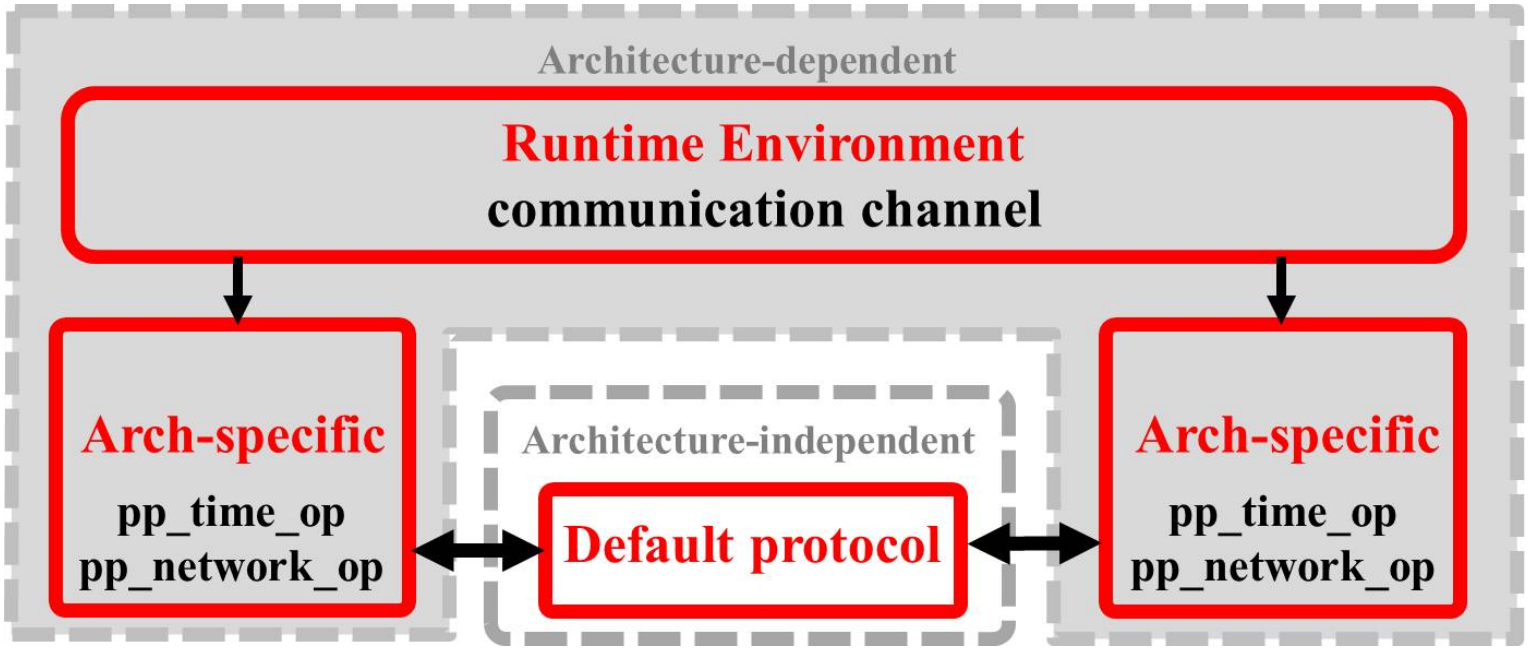
Outline

Overview of simulation “arch”



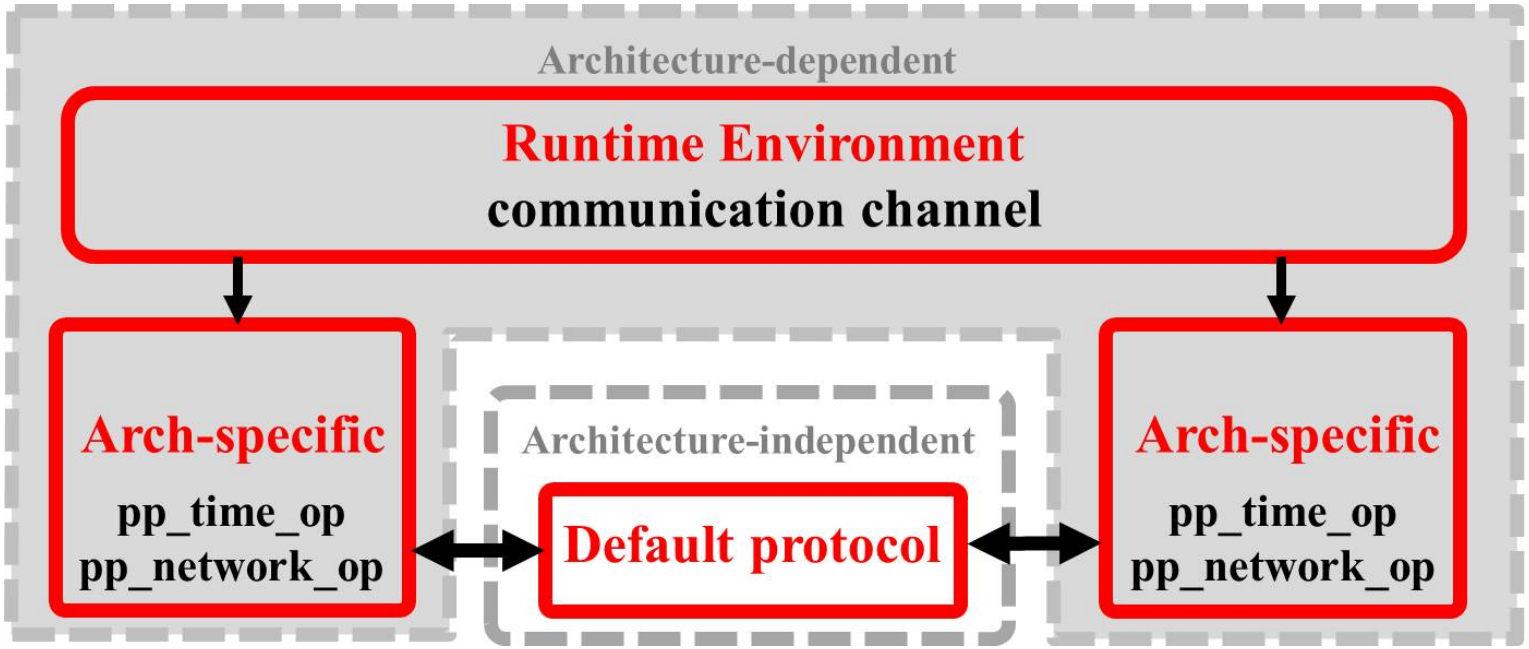
- Developed by Pietro Fezzardi to test non-WR servo

Overview of simulation “arch”



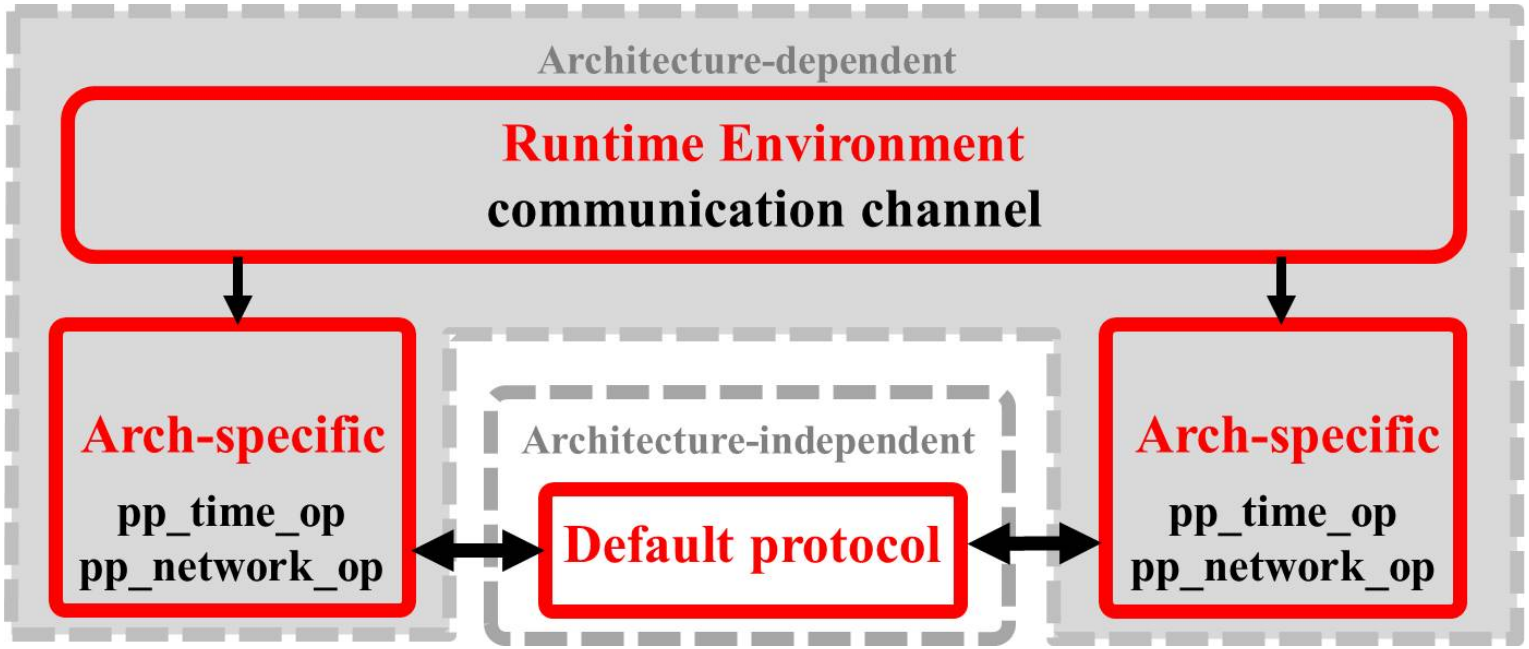
- Developed by Pietro Fezzardi to test non-WR servo
- Provides time fast-forward benefiting

Overview of simulation “arch”



- Developed by Pietro Fezzardi to test non-WR servo
- Provides time fast-forward benefiting
  - network– & timeout–driven protocol core
  - synchronous & immediate execution of state machine

Overview of simulation “arch”



- Developed by Pietro Fezzardi to test non-WR servo
- Provides time fast-forward benefiting
  - network– & timeout–driven protocol core
  - synchronous & immediate execution of state machine
- Simulates thousands of PTP iterations per second

Simulation to tune and debug non-WR servo

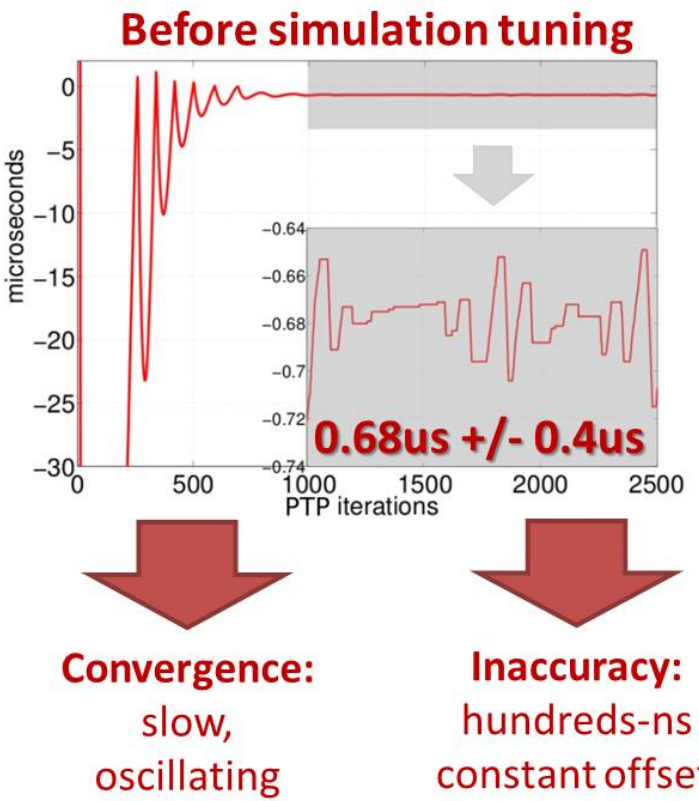
Simulation parameters:

- Range of packet delay variation
- Initial frequency offset

Simulation to tune and debug non-WR servo

Simulation parameters:

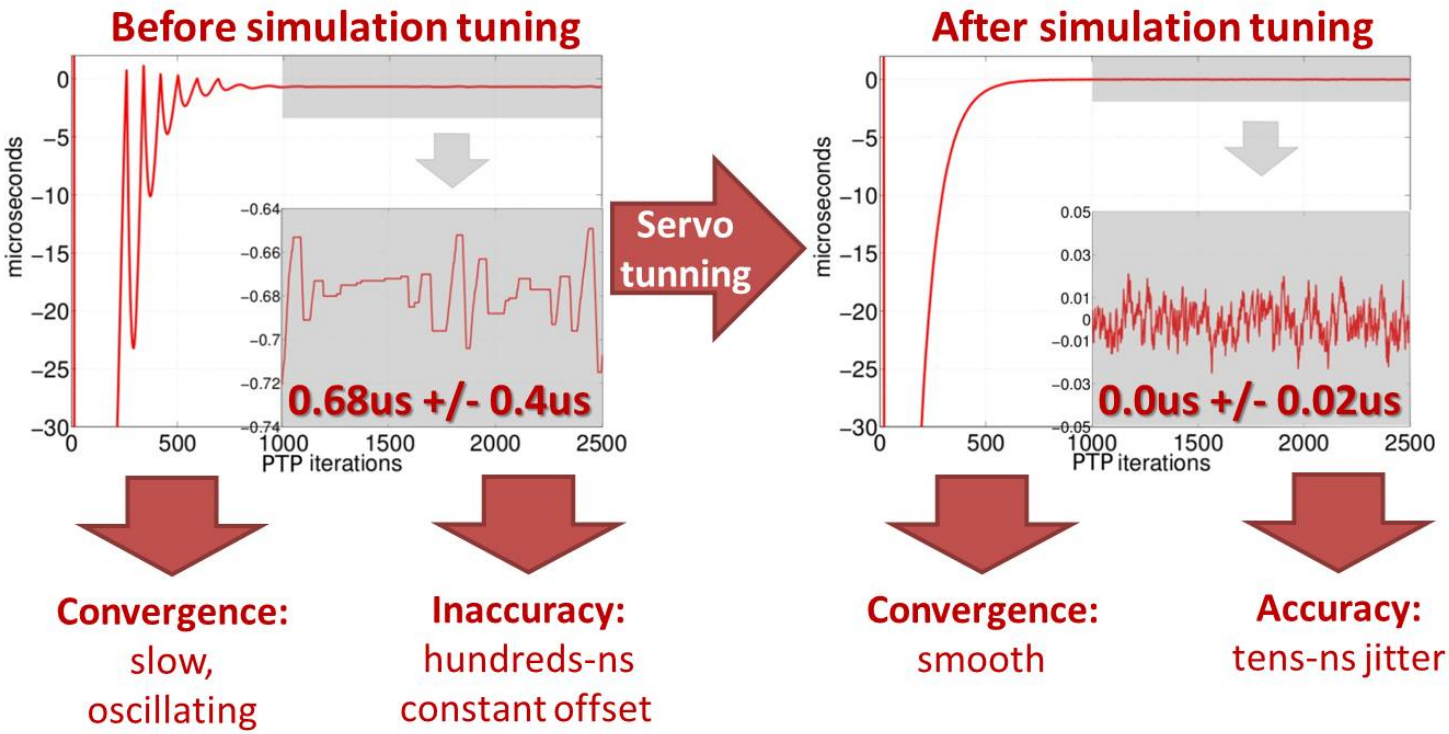
- Range of packet delay variation
- Initial frequency offset



Simulation to tune and debug non-WR servo

Simulation parameters:

- Range of packet delay variation
- Initial frequency offset



Random fault injection

- Bug report:
  - problem on freestanding node
  - WR PLL not locking when messages lost
- Debugging: random message drop implementation
- PPSi design benefit:
  - available in all architectures
  - useful for non-WR servo simulation-testing

Outline

Summary

- Clean and well-coded PTP implementation

Summary

- Clean and well-coded PTP implementation
- Unmatched in Free Software world
  - portability
  - extensibility

Summary

- Clean and well-coded PTP implementation
- Unmatched in Free Software world
  - portability
  - extensibility
- Layered design

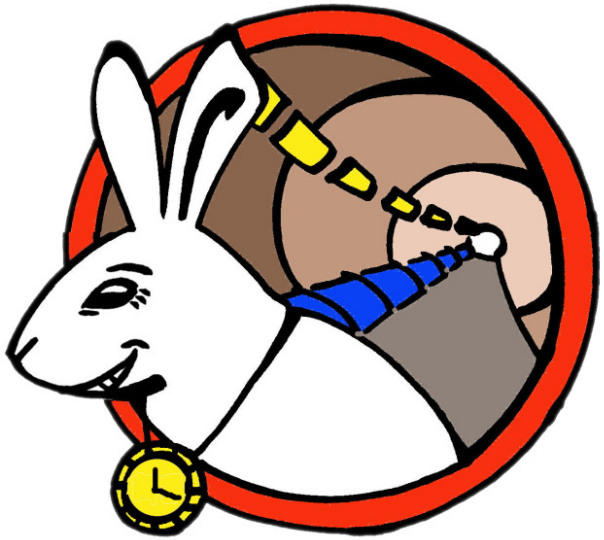
Summary

- Clean and well-coded PTP implementation
- Unmatched in Free Software world
  - portability
  - extensibility
- Layered design
- Abstracted interactions between layers

Summary

- Clean and well-coded PTP implementation
- Unmatched in Free Software world
  - portability
  - extensibility
- Layered design
- Abstracted interactions between layers
- Active and ever-growing users community

Questions and answers



Thank you

`www.ohwr.org/projects/ppsi`