Frame Body Concept

Frame Body Concept

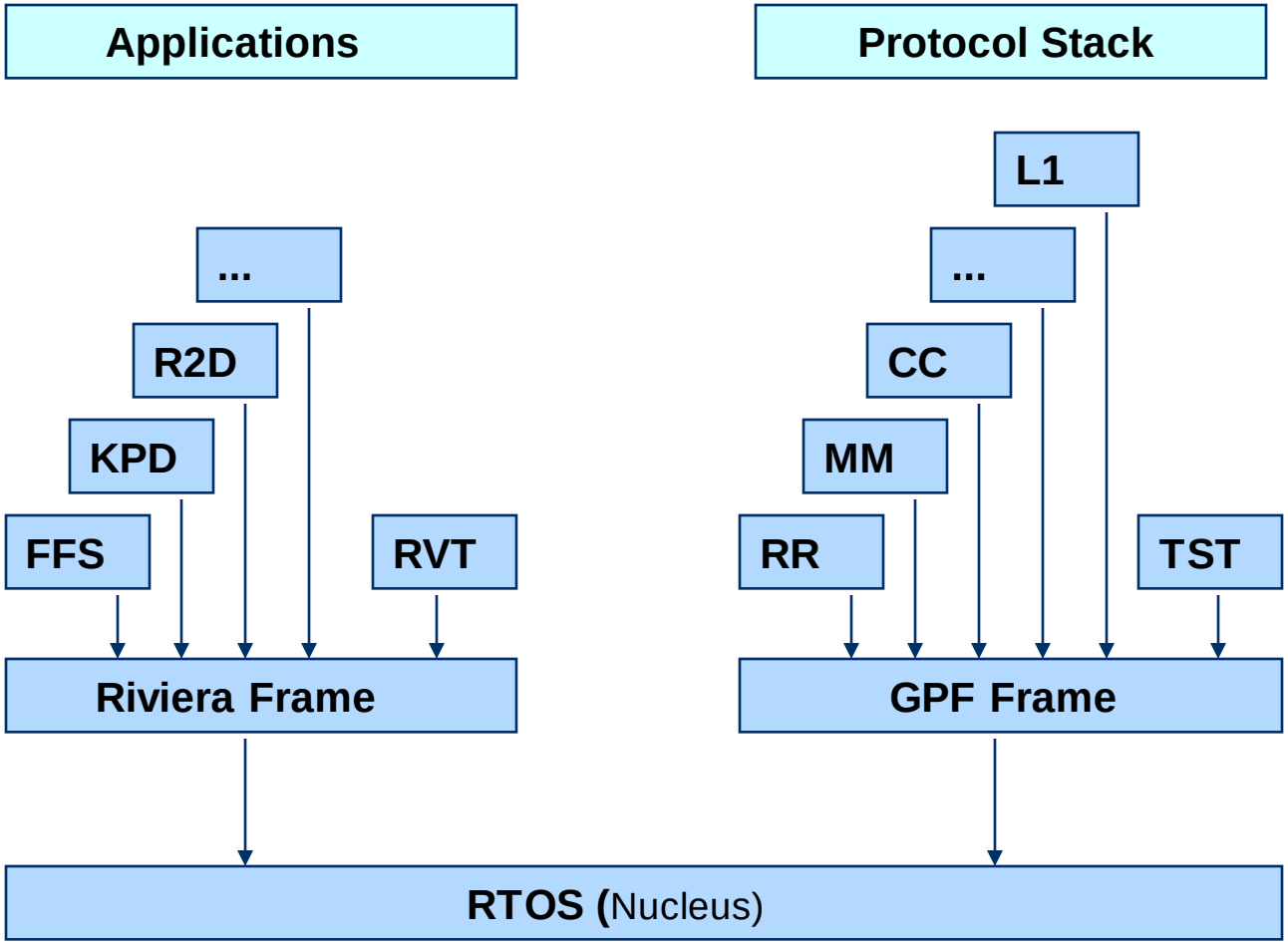
- Overview
- Frame Functionality & Interfaces
- Test Interface
- System Setup
- Layer 1 Interface

Frame Body Concept

- **Overview**
- Frame Functionality & Interfaces
- Test Interface
- System Setup
- Layer 1 Interface

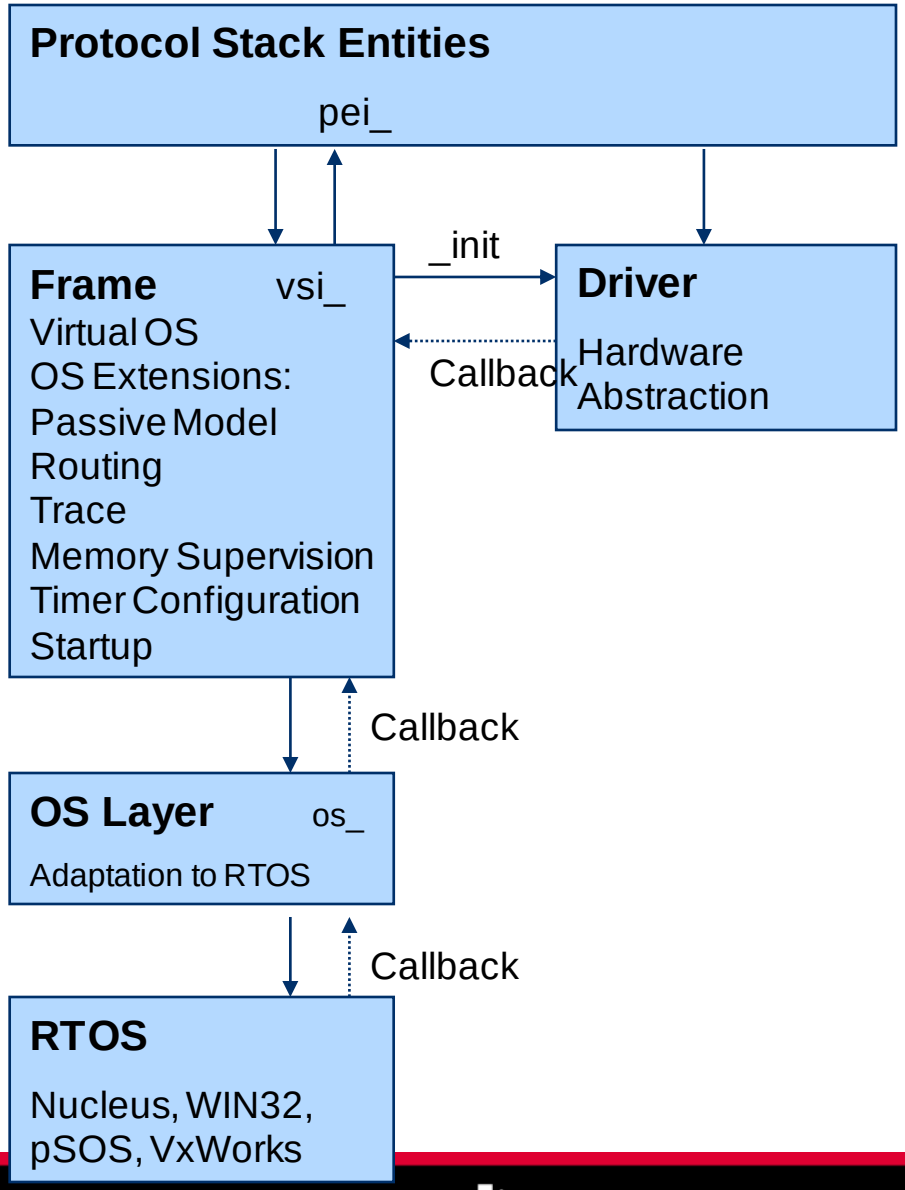
Frame Body Concept - Overview

Software Architecture



Frame Body Concept - Overview

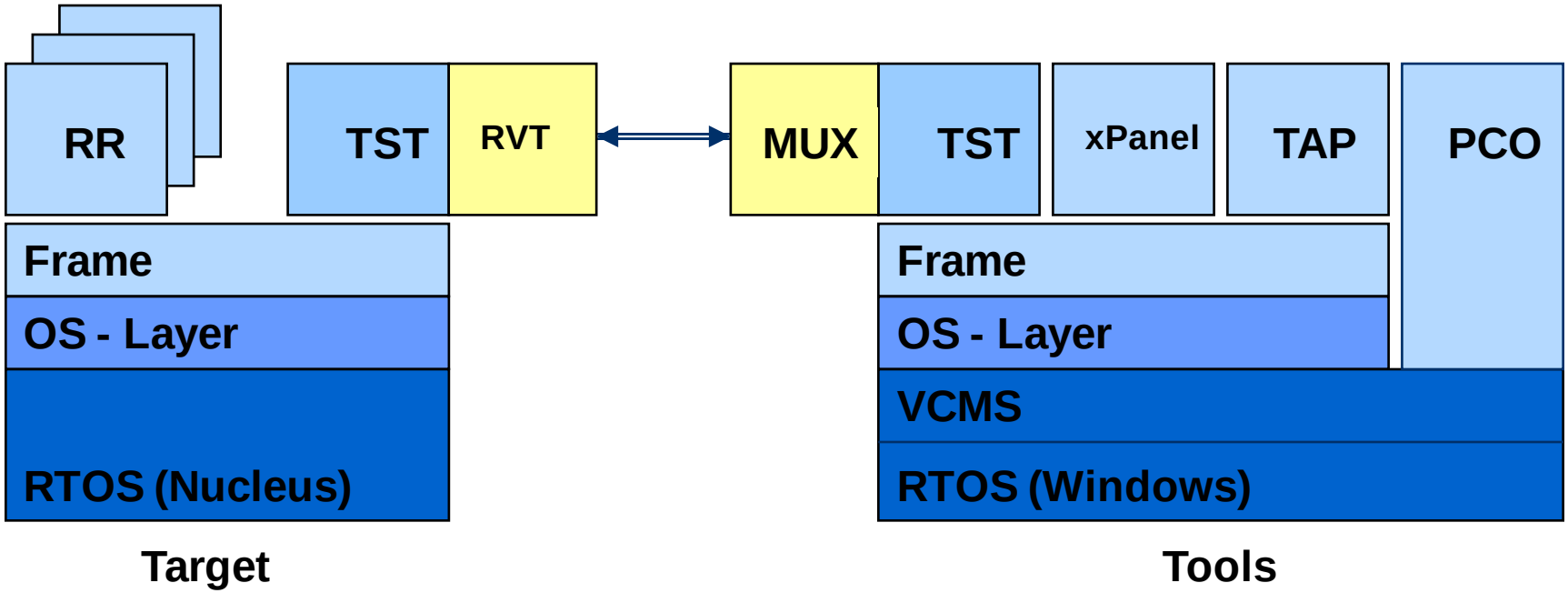
Software Architecture



- Virtual System Interface (VSI)
- Protocol Stack Entity Interface (PEI)
- Operating System Interface (OS)

Frame Body Concept - Overview

Frame in Target and Tools



The same frame is used for target and tools !

Frame Body Concept - Overview

Protocol Stack Entities

- Entities implement the protocol functionality
- State machine
- Logic of the GSM/GPRS/UMTS specification (e.g. GSM 04.08)
- No direct access to RTOS resources
- Main loop of process (active body)

Frame Body Concept - Overview

Frame Functionality

- Task Management
- Memory Management
- Communication
- Timer Management
- Routing
- Tracing
- Test Interface
- RTOS Access
- Main loops of processes (passive body)

Frame Body Concept - Overview

OS Layer Functionality

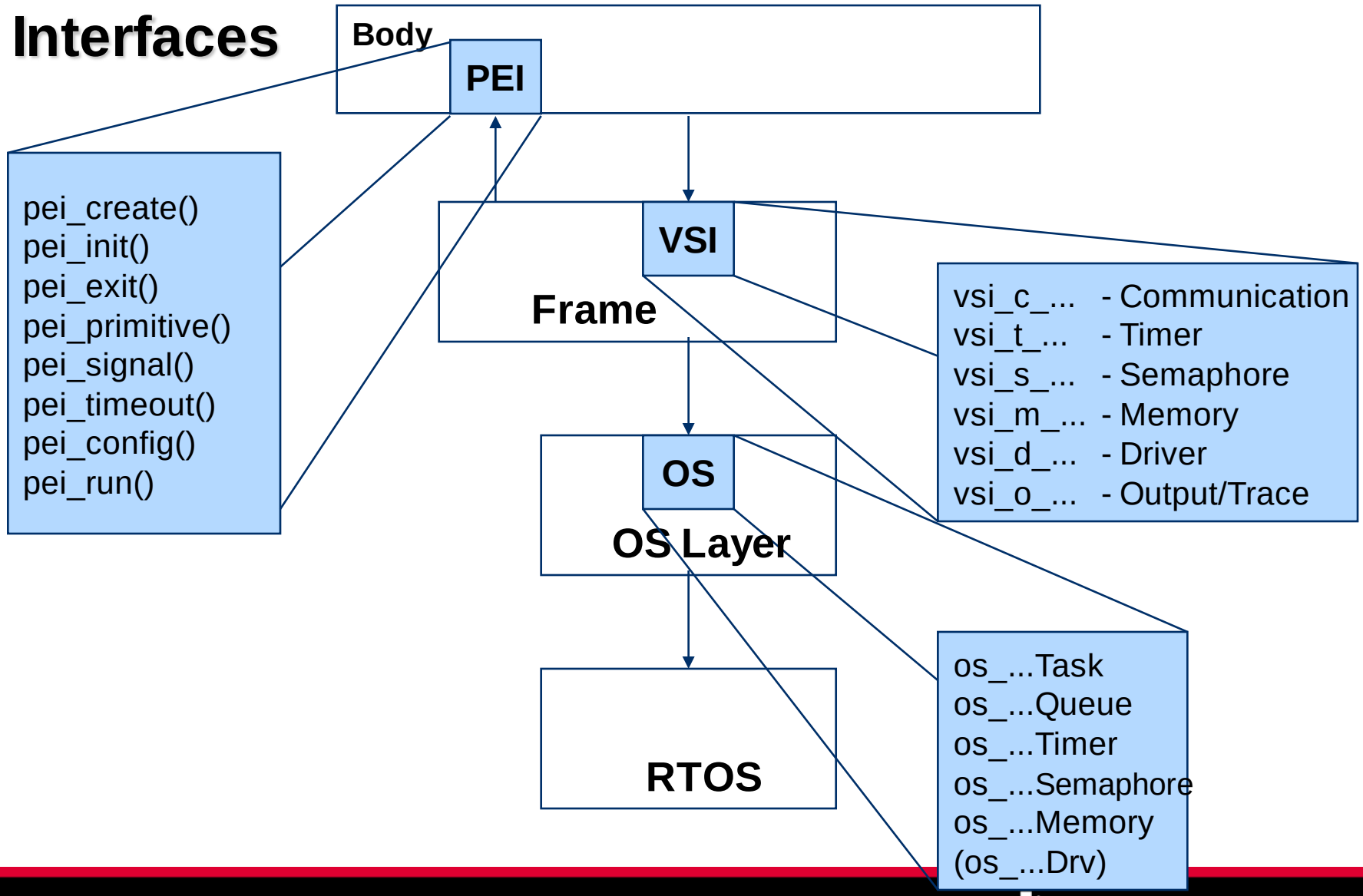
- Adaptation to the RTOS
- RTOS dependent diagnose
- Platform dependencies – Nucleus MNT & ARM7
- OS interface is independent of RTOS

Frame Body Concept - Overview

Real-time Operating System (Nucleus)

- Tasks
- Memory
  - Dynamic Memory
  - Partition Memory
- Timer
- Semaphores
- Interrupt handling (HISR)

Frame Body Concept - Overview



Frame Body Concept

- Overview
- **Frame Functionality & Interfaces**
- Test Interface
- System Setup
- Layer 1 Interface

Frame Body Concept – Entity Interface

Protocol Stack Entity Interface (PEI)

- pei_create() - export entity creation parameters
- pei_init() - open communication channels, initialize entity
- pei_exit() - close communication channels, free resources
- pei_primitive() - process primitives
- pei_timeout() - process timeouts
- pei_signal() - process signals
- pei_config() - dynamic entity configuration
- pei_run() - main loop for active body configuration

Frame Body Concept – Entity Interface

Parameters exported by pei_create()

```
static T_PEI_INFO pei_info =
        {
         CC_NAME,
         {                    /* pei-table */
           pei_init,
           pei_exit,
           pei_primitive,
           pei_timeout,
           NULL,                        /* pei_signal,  */
           NULL,                        /* pei_run,     */
           NULL,                        /* pei_config,  */
           pei_monitor
         },
         CC_STACK_SIZE,
         CC_QUEUE_ENTRIES,
         CC_PRIORITY,
         CC_TIMERS,
         CC_PROC_TYPE                 /* ACTIVE/PASSIVE body */
        };
```

Frame Body Concept - Startup

System Startup (1) – Root Task Context

- ⇨Nucleus initialization
- ⇨Call of Application_Initialize()
- ⇨Call of StartFrame()
  - ⇨Initialize Frame
    - ⇨Create memory pools
    - ⇨Initialize global tables
  - ⇨ Create Tasks
    - ⇨ Call pei_create() functions from task list
    - ⇨ Create tasks with the parameters exported by pei_create()
  - ⇨Start Tasks

Frame Body Concept - Startup

System Startup (2) – Created Task Context

- TaskEntry (TaskHandle) called by the RTOS

```
{
   Create input queue
   Entity initialization -> pei_init()
   If ( ACTIVE_BODY)
      pei_run()
   while (1)              /* PASSIVE_BODY */
   {
      Wait for message
      Process message
   }
}
```

- The same task entry function for all tasks
- Each task has its own stack, stack size and priority

Frame Body Concept - Startup

Task List

```
const T_COMPONENT_ADDRESS sim_list[] =
{
  { sim_pei_create,      NULL,   ASSIGNED_BY_CONDAT },
  { NULL,                NULL,   0 }
};
const T_COMPONENT_ADDRESS mmgmm_list[] =
{
  { mm_pei_create,       NULL,   ASSIGNED_BY_CONDAT },
  { gmm_pei_create,      NULL,   ASSIGNED_BY_CONDAT },
  { NULL,                NULL,   (int)"MMGMM" }
};
const T_COMPONENT_ADDRESS rr_list[] =
{
  { rr_pei_create,       NULL,   ASSIGNED_BY_CONDAT },
  { NULL,                NULL,   0 }
};
const T_COMPONENT_ADDRESS *ComponentTables[]=
{
  ...
  sim_list,
  mmgmm_list,
  rr_list,
  ...
}
```

Frame Body Concept - Startup

Active/Passive Body

	Active Body	Passive Body
Main Loop Location	Entity	Frame
Main Loop Function	pei_run()	pf_TaskEntry()
Functionality	customer defined	waiting for message
		processing message

Selection of active or passive is done with flags exported by pei_create()

Frame Body Concept - Startup

Create New Entity

- add pei interface by using the template xxx_pei.c
- add address of xxx_pei_create() to task list
- add source files for further functionality

Frame Body Concept - Startup

Task Management

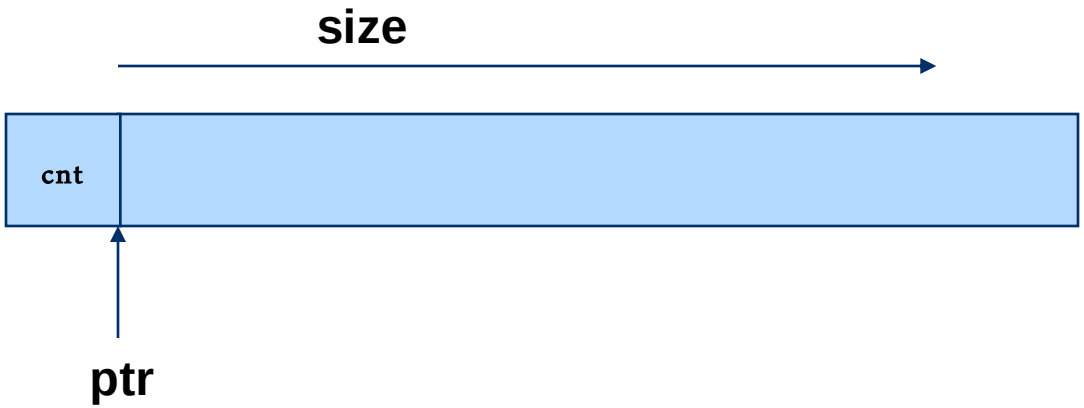
- **Protocol Stack**
  - Protocol stack entities run in their own RTOS task or may be grouped in one task, e.g. CC, SS, SMS, SM are grouped to the CM task.
    - Tasks are created at initialization
    - Static priorities set during initialization
    - Scheduling is done by RTOS (preemptive multitasking)
- **Tools**
  - Frame as shared DLL
  - Different applications are dynamically started and stopped

Frame Body Concept – Memory Management

Partition Memory Management (1)

- Fixed sized buffers of different sizes ->xxxconst.h

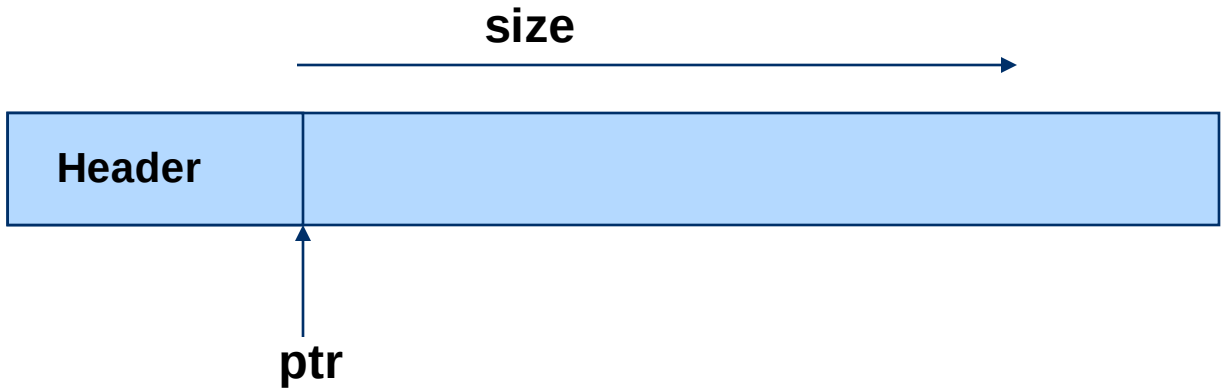
- Dynamic memory : MALLOC(size) -> vsi_m_new(size)
  MFREE(ptr) -> vsi_m_free(ptr)



Frame Body Concept – Memory Management

Partition Memory Management (2)

- Static primitives: PALLOC(name)-> vsi_c_new(size)
  PFREE(ptr) -> vsi_c_free(ptr)

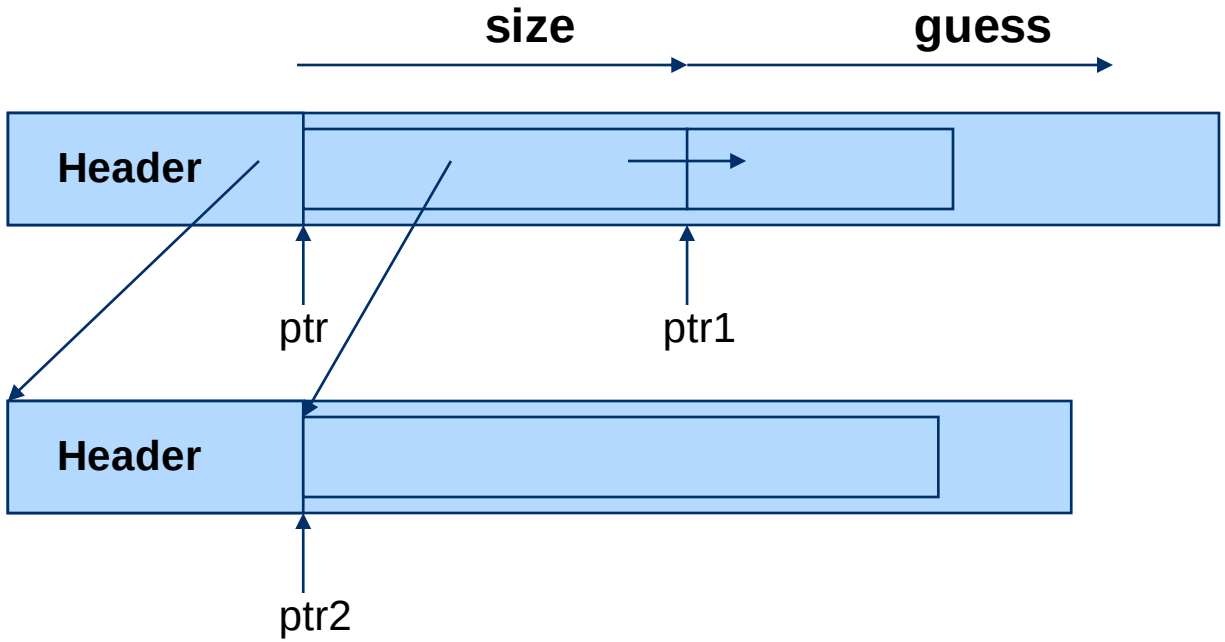


Primitive header is initialized by the VSI functions

Frame Body Concept – Memory Management

Partition Memory Management (3)

- Dynamic pimitives:
- DRPO_ALLOC(name,guess) -> vsi_drpo_new(size,opc,guess)
- DP_ALLOC(ptr,size) -> vsi_dp_new(ptr,size)
- FREE(ptr) -> vsi_free(ptr)

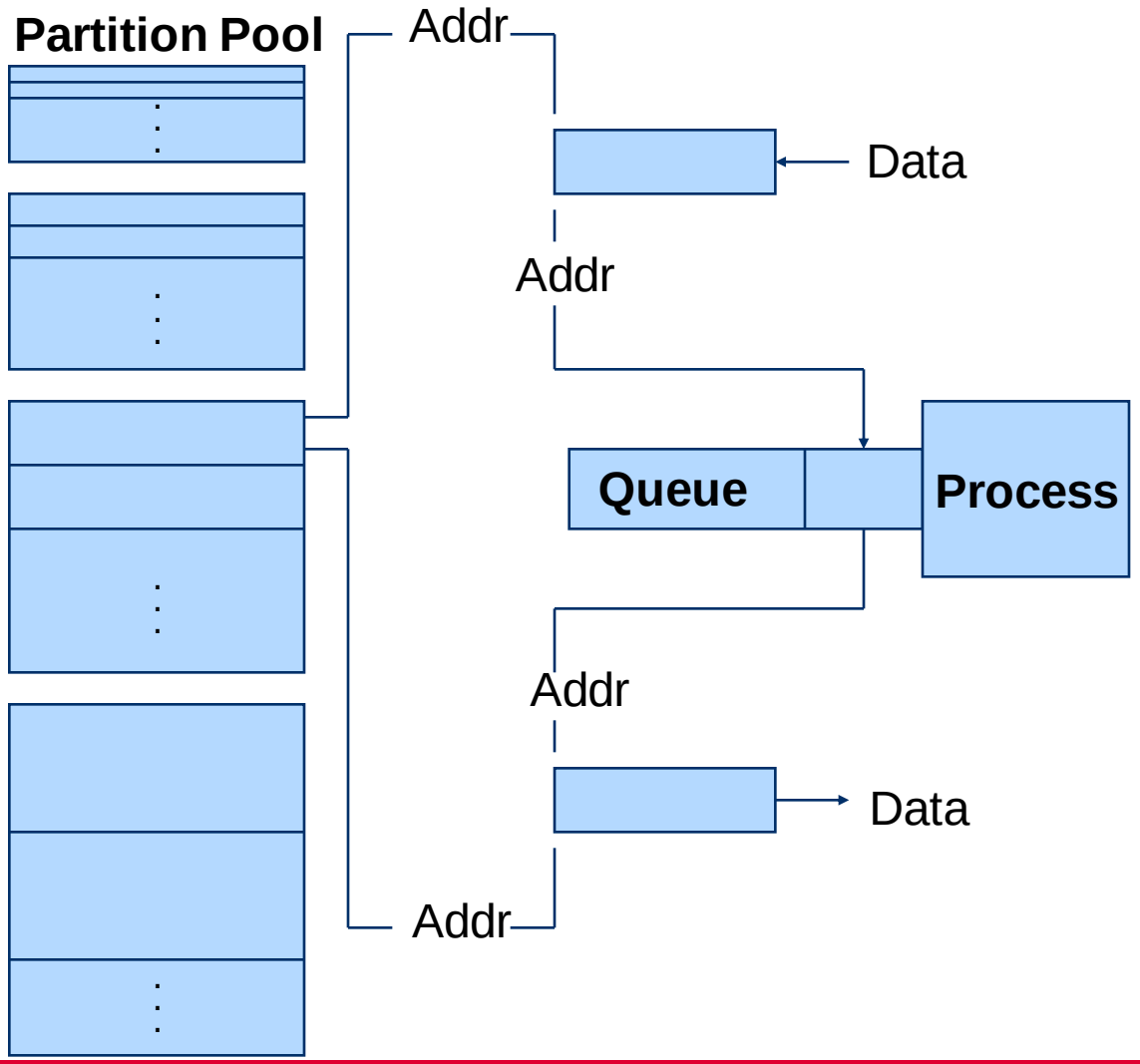


Frame Body Concept - Communication

- 1 Input Message Queue for each Entity
- Primitives -> low priority
- Signals -> high priority
- Communication between Entities with primitives
  - Sender
    - Allocate a memory partition (PALLOC)
    - Store primitive data in partition
    - Write partition address to destination queue (PSEND)
  - Receiver
    - Read partition address from input queue
    - Process primitive - Call pei_primitive()
    - Free partition (PFREE)

Frame Body Concept - Communication

Example Interprocess Communication

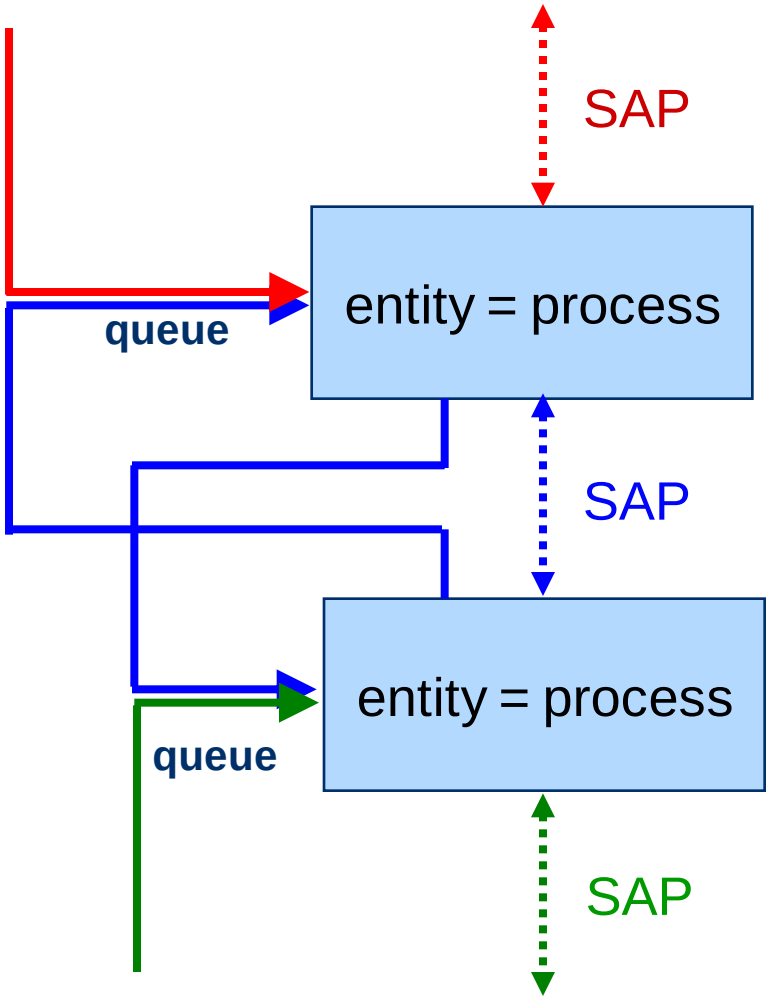


1. Allocate Partition
2. Store Primitive Data
3. Send Primitive
4. Receive Primitive
5. Process Primitive
6. Deallocate Partition

Frame Body Concept - Communication

⇨ Expressions

- entity = process
- primitive = struct
- SAP = number
- 1 input queue per process



Frame Body Concept - Communication

Primitive Processing

Header in each primitive

- ⇨ operation code (opc) 16/32 bit
  - ⇨ SAP number
  - ⇨ primitive number
  - ⇨ uplink/downlink
  - ⇨ protocol/system primitive
- ⇨ length
- ⇨ …

Frame Body Concept – Timer Management

Timer Management

- Start
  - Timer start request by an entity
  - Frame starts RTOS timer
  - ATTENTION: timer index has to be in the range between 0 and the number of timers exported by pei_create()
- Expiration
  - RTOS calls timeout routine in the frame
  - Frame calls pei_timeout() in the body
- Configuration
  - speed up/slow down/suppress

```
SHORT vsi_t_start  ( T_HANDLE caller, USHORT index, T_TIME time )
SHORT vsi_t_stop   ( T_HANDLE caller, USHORT index )
SHORT vsi_t_status ( T_HANDLE caller, USHORT index, T_TIME *time )
SHORT vsi_t_config (T_HANDLE caller, USHORT index, UBYTE mode, ULONG value )
```

Frame Body Concept – Semaphore Management

Semaphore Management

- open semaphore
  - create if not existing
  - return handle if existing
  - initial count
- obtain
- release

```
T_HANDLE vsi_s_open ( T_HANDLE caller, char *name, USHORT count )
SHORT    vsi_s_get ( T_HANDLE caller, T_HANDLE handle )
SHORT    vsi_s_release ( T_HANDLE caller, T_HANDLE handle )
SHORT    vsi_s_status (T_HANDLE caller, T_HANDLE handle, USHORT *count )
```

Frame Body Concept - Routing

Routing

- ⇨ Redirection and duplication of primitives
- ⇨ Dynamic configuration via test interface
- ⇨ Filters available to select a subset of primitives

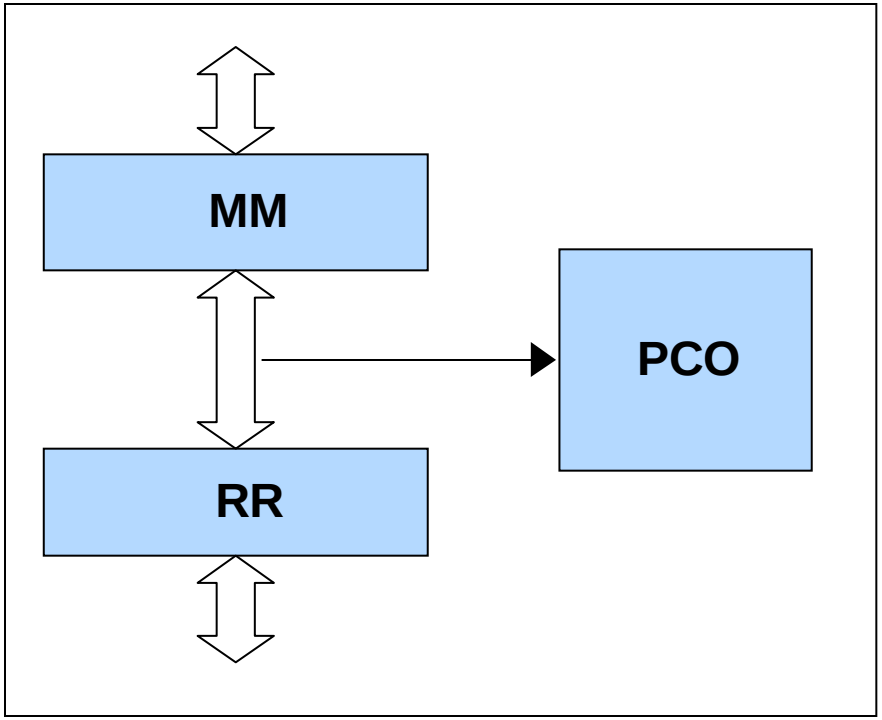
Examples

RR DUPLICATE MM PCO	all primitives RR sends to MM are duplicated to PCO
RR REDIRECT ALL TAP	all primitives RR sends are redirected to TAP
MM DUPLICATE RR 0000011000000001 PCO	duplicate RR_ACTIVATE_REQ to PCO
RR REDIRECT MM NULL	all primitives RR sends to MM are discarded
RR REDIRECT CLEAN	clean all redirections of RR

Frame Body Concept - Routing

Routing - Observation

Monitoring of primitive communication



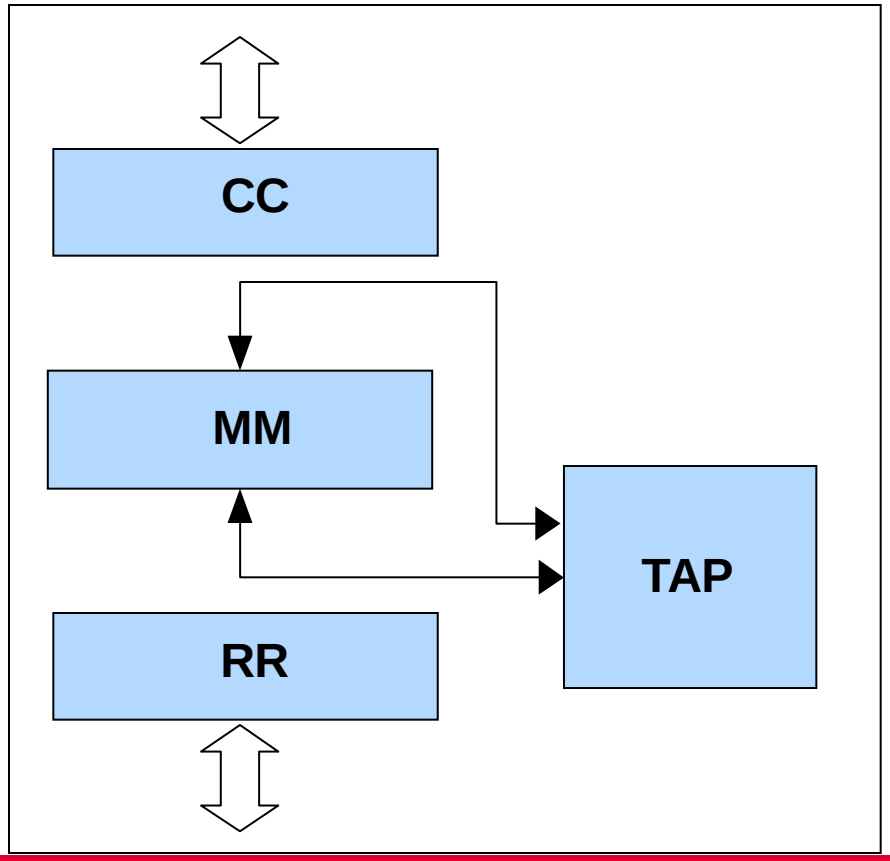
RR DUPLICATE MM PCO

MM DUPLICATE RR PCO

Frame Body Concept - Routing

Routing - Test

Test of a single entity in the protocol stack



MM REDIRECT CC TAP

MM REDIRECT RR TAP

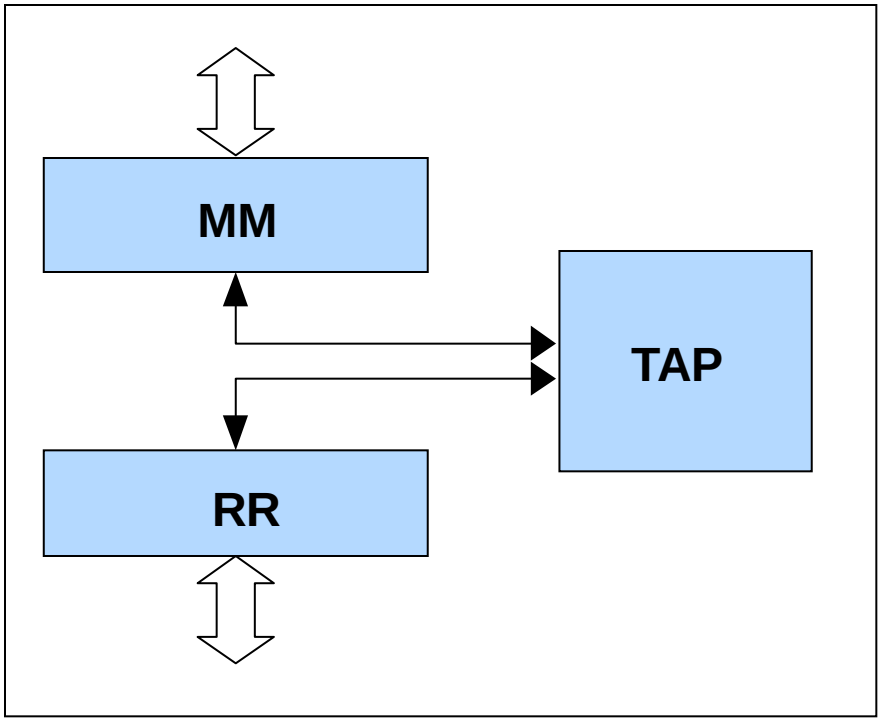
RR REDIRECT MM NULL

CC REDIRECT MM NULL

Frame Body Concept - Routing

Routing - Manipulation

Manipulation of primitive contents



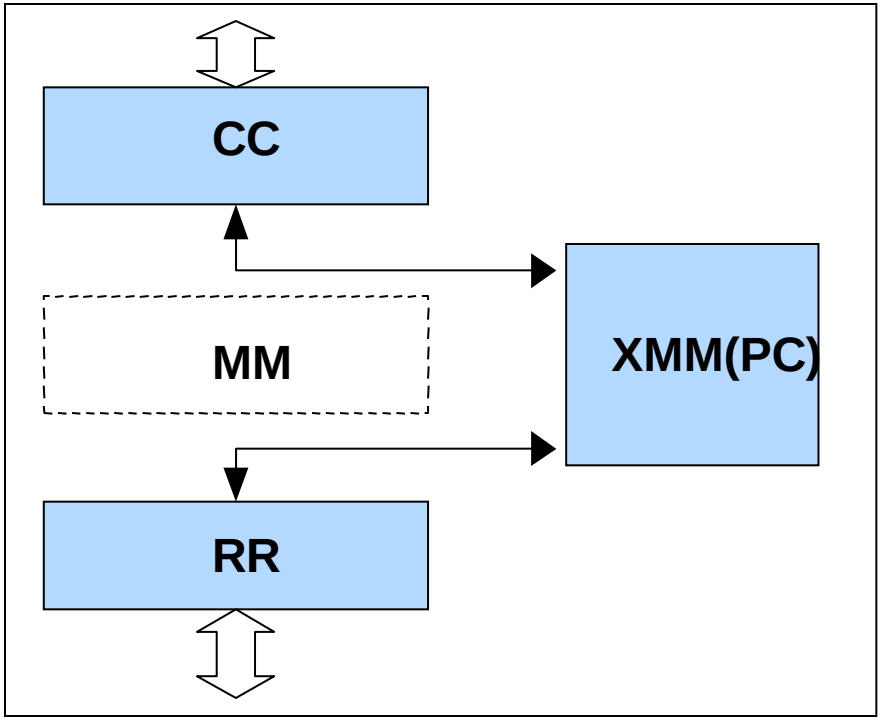
RR REDIRECT MM TAP

MM REDIRECT RR TAP

Frame Body Concept - Routing

Routing - Simulation

Run entities on different platforms (target, PC)



MM REDIRECT RR NULL

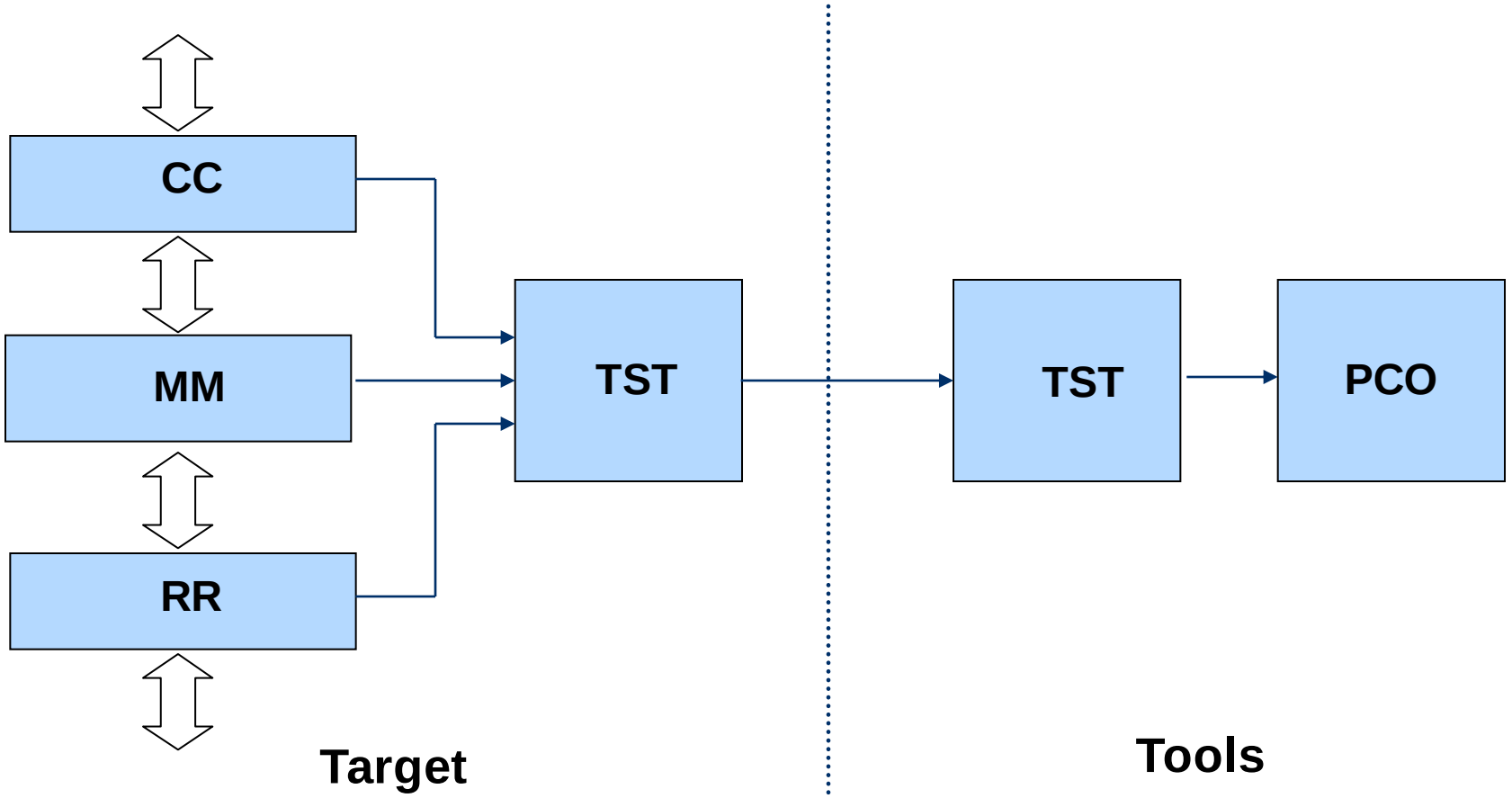
MM REDIRECT CC NULL

CC REDIRECT MM XMM

RR REDIRECT MM XMM

Frame Body Concept - Tracing

A Way to Follow the System State without Debugger



Frame Body Concept - Tracing

- **Trace Functionality**
  - Trace Classes
    - Function Traces — TRACE_FUNCTION()
    - Primitive Traces — PTRACE_IN(), PTRACE_OUT()
    - State Traces — GET_STATE(), SET_STATE()
    - Error Traces — TRACE_ERROR()
    - Event Traces — TRACE_EVENT()
    - System Traces — within Frame
    - CCD Traces — within CCD
  - Comparison of trace class with trace mask in the frame
  - Dynamic activation of traces via system primitives
  - Binary coding of strings to be traced
  - Customer can define his own trace classes

Frame Body Concept – System Primitives

Read Resource Information

STATUS TASK

Name: RR Stat: 6 Count: 288 Prio: 205 Stack: 871240 Size: 1736 Untouched: 808

STATUS QUEUE

Name: RR Addr: 879188 Entries: 10 Used: 0 MaxUsed: 4

STATUS PARTITION

Name: POOL10 Addr: 85c998 PoolSize: 7128 PartSize: 64 Free: 86 Used: 13

STATUS MEMORY

Name: CONDAT Addr: 86e634 Size: 60000 Min: 4 Free: 12548 Suspend: 6

STATUS SEMAPHORE

Name: CCD_COD Count: 1 Suspend: 11 Waiting: 0

Frame Body Concept – Partition Supervision

Partition Pool Monitor - Functionality

Monitoring of the partition states

- ⇨ state
- ⇨ owner
- ⇨ last user (file,line)
- ⇨ time

Statistics on partition usage

- ⇨ Sizes of used partitions
- ⇨ Numbers of used partitions
- ⇨ Dump via test interface

Frame Body Concept – Partition Supervision

Partition Pool Monitor - Output

⇨ Allocated Partitions

[PPM]: PARTITION 0x85ed88, OPC 0x0 STATE ALLOCATED OWNER UART, TIME 670, uart_dtxf.c,245

[PPM]: PARTITION 0x85edd8, OPC 0x0 STATE ALLOCATED OWNER MMI, TIME 666, aci_lst.c,147

⇨ Statistics

[PPM]: POOL NAME: PRIM

[PPM]: POOL 0 (size 60)

[PPM]: MAXBYTE pool 0: 1849 bytes => 61 %

[PPM]: MAXBYTE partitions pool 0: 5, 0, 3, 41, 1

[PPM]: MAXPART pool 0: 50 partitions => 50 %

[PPM]: MAXPART partitions pool 0: 5, 0, 3, 41, 1

[PPM]: MAXRANGE partitions pool 0: 6, 3, 11, 41, 4

[PPM]: TOTAL partitions pool 0: 17, 23, 541, 497, 115

Frame Body Concept – Error Handling

Error Handling – System Warnings

SYSTEM WARNING: Waited for space in *task* queue

SYSTEM WARNING: Waited for partition in *task*, size *size*, opc *opc*

SYSTEM WARNING: Bigger partition allocated than requested, *task, size*

SYSTEM WARNING: Partition Deallocation failed in *task*

SYSTEM WARNING: *task* freed partition stored in *task* queue

SYSTEM WARNING: Freed partition sent in *task, opc*

SYSTEM WARNING: Receiver process *task* unknown

SYSTEM WARNING: Invalid system primitive string

SYSTEM WARNING: Number of written > requested partition size in *file, line*

System warning results in

⇨**Dump of system state**

⇨**dar_diagnose_write (warning_string, DAR_WARNING, CONDAT_FRM_USE_ID)**

Frame Body Concept – Error Handling

Error Handling - System Errors

SYSTEM ERROR: Number of Timers > MAX_TIMER

SYSTEM ERROR: Number of entities > MAX_ENTITIES

SYSTEM ERROR: Number of started timers > MAX_SIMULTANEOUS_TIMER

SYSTEM ERROR: Number of created semaphores > MAX_SEMAPHORES

SYSTEM ERROR: Partition Guard Pattern destroyed *task, ptr, opc*

SYSTEM ERROR: No Partition available, *task, size*

SYSTEM ERROR: *task* write attempt to *dest* queue failed

SYSTEM ERROR: *task* Stack overflow

SYSTEM ERROR: Error at creating *task* Queue

SYSTEM ERROR: TimerIndex > NumOfTimers for *task*

System error results in

⇨**Dump of system state**

⇨**dar_generate_emergency (error_string, ..., CONDAT_FRM_USE_ID)**

Frame Body Concept – Error Handling

Error Handling - System State Dump

TST: task state:	NU_READY	queue state: 0 of 50 used
TST:	NU_READY - Normal suspend, Link Register = 0x81186b	
TST: Processed Message:	SIGNAL opc: 0x8, time: 26149 ptr: 0x85f7bc	
MMI: task state:	NU_QUEUE_SUSPEND (MMI)	queue state: 0 of 20 used
CST: task state:	NU_READY	queue state: 2 of 10 used
CST:	NU_READY - Normal suspend, Link Register = 0x810ded	
CST: Messages in queue:	TIMEOUT opc: 0x1, time: 0 ptr: 0x0	
CST: Messages in queue:	PRIMITIVE opc: 0x5600, time: 27916 ptr: 0x863c24	
SIM: task state:	NU_QUEUE_SUSPEND (SIM)	queue state: 0 of 10 used
CM: task state:	NU_QUEUE_SUSPEND (CM)	queue state: 0 of 10 used
MMGMM: task state:	NU_READY	queue state: 1 of 10 used
MMGMM:	NU_READY - IRQ suspend, Program Counter = 0x1ad54c	
MMGMM: Processed Message:	PRIMITIVE opc: 0x8000, time: 26149 ptr: 0x85f82c	
MMGMM: Messages in queue:	PRIMITIVE opc: 0x4608, time: 26690 ptr: 0x85ff2c	

Frame Body Concept

- Overview
- Frame Functionality & Interfaces
- **Test Interface**
- System Setup
- Layer 1 Interface

Frame Body Concept – Test Interface

- **Running as a Task**
- **Test Interface Functionality**
  - Input from test system
    - System primitives (configuration)
    - Protocol primitives
  - Output to test system
    - Traces
    - Routed protocol primitives
- **Simple exchange of drivers**

Frame Body Concept – Test Interface

Test Interface Driver List

- Name
- Address of *drv*_Init()
- Process to be notified
- Default configuration

```
const T_DRV_LIST_ENTRY DrvList[] =
{
  { NULL,          NULL,              NULL,   NULL },
  { "TIF",         TIF_Init, "TST",   NULL         },
  { "TR",          TR_Init,           NULL,   NULL },
  { "SER",         SER_Init,          NULL,   &SER_DefaultConfig },
  { NULL,          NULL,              NULL,   NULL }
};
```

Frame Body Concept – Test Interface

Primitive Converter (PCON)

⇨ exchange of primitives between platforms with different number representation (little vs. big endian)

⇨ exchange of primitives between platforms with different alignment model

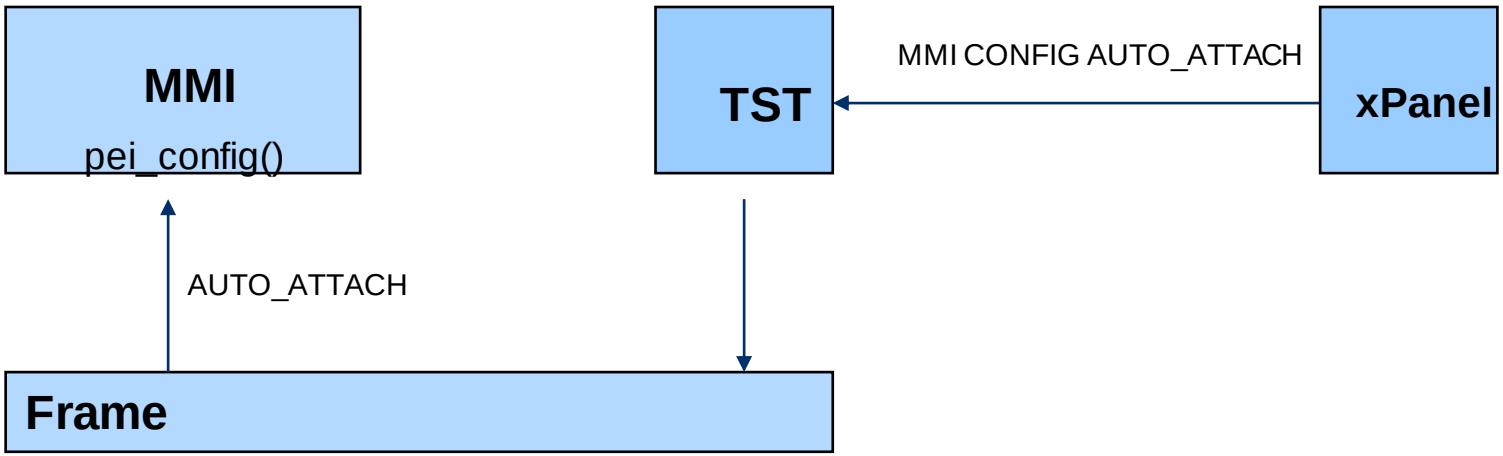
⇨ exchange of primitives containing pointers

⇨ can be called in TIF driver when primitive is sent or received

Frame Body Concept – Test Interface

Dynamic Entity Configuration

Send strings transparently through TST and the frame to entity

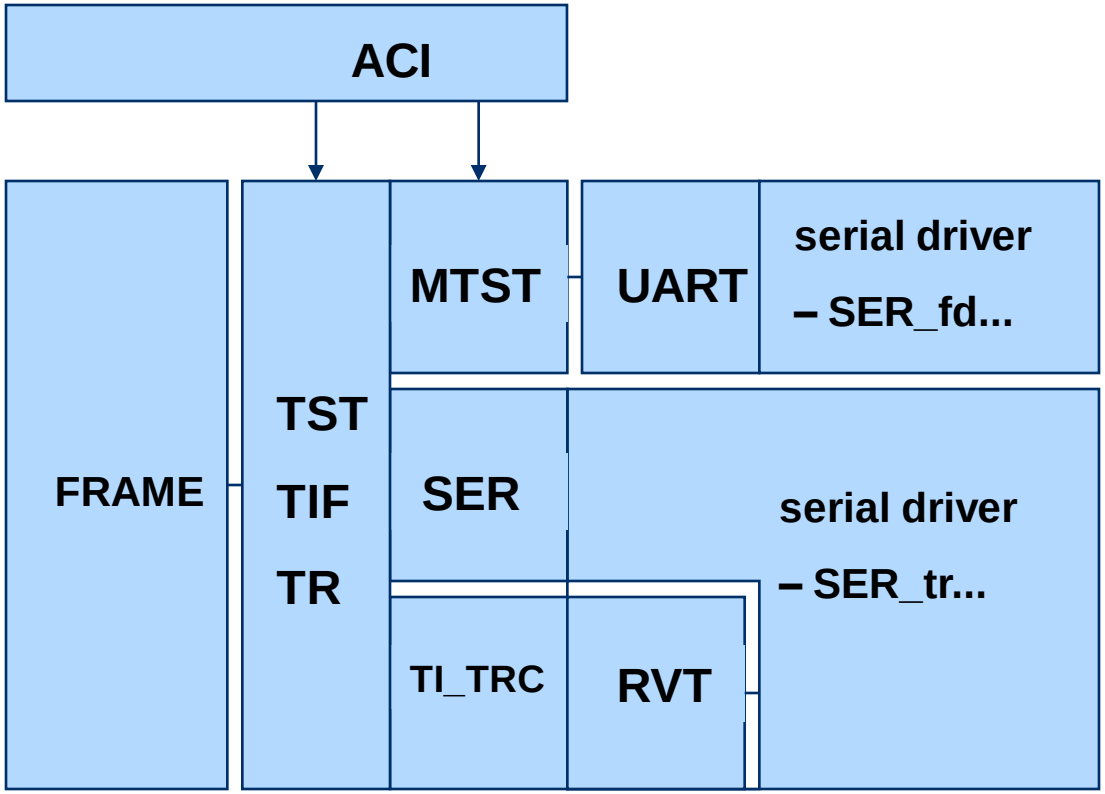


Used for

- ⇨ **Entity configuration** GMM CONFIG MULTISLOT_CLASS=<8>
- ⇨ **AT commands** MMI CONFIG AT+CFUN=1
- ⇨ **Timer configuration** L2R CONFIG TIMER_SET TUP_SND 1000

Frame Body Concept – Test Interface

Test Interface – Target Driver Configuration



Frame Body Concept

- Overview
- Frame Functionality & Interfaces
- Test Interface
- **System Setup**
- Layer 1 Interface

Frame Body Concept – System Setup

Configuration Files

- xxxcomp.c
  - Task list
- xxxdrv.c
  - Test interface driver list
- xxxinit.c
  - Application_Initialize() function
- xxxconst.h
  - Number and size of partitions
  - Number of tasks, timers, semaphores

⇨No frame recompilation when configuration files are changed

Frame Body Concept – System Setup

Provided Frame Libraries

frame	task management, communication, memory, timers, Semaphores, error handling
tif	test interface, drivers
misc	common services

Extensions

_na7	nucleus arm7
_npc	nucleus MNT (PC Simulation)
_wn	WIN32
_db	debug
_ps	partition pool supervision
_pc	pcon
_ir	running in internal RAM
_fl	running in flash

Examples:	frame_na7_db_fl.lib	(target)
	frame_npc_db_ps.lib	(PC simulation)

Frame Body Concept – System Setup

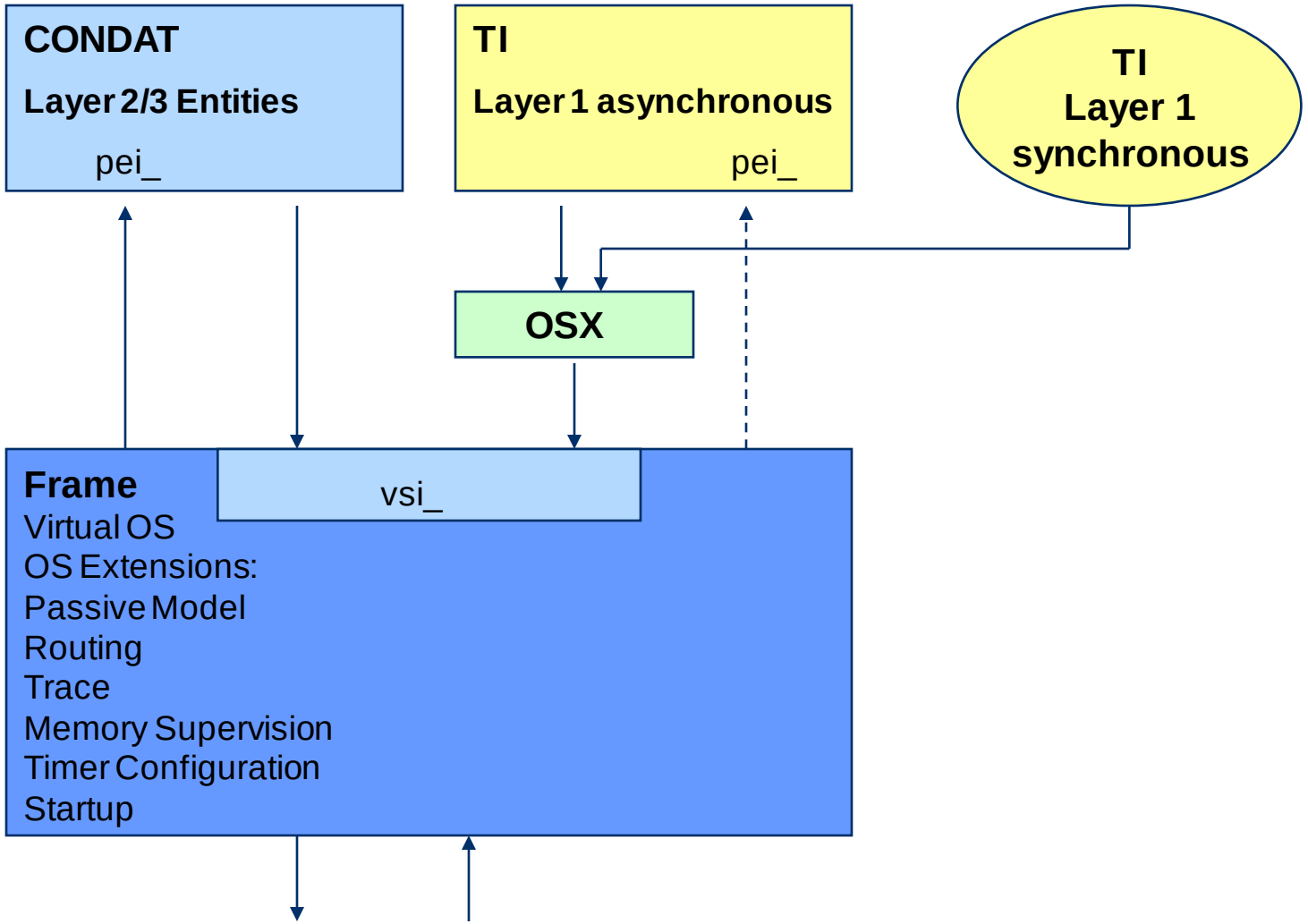
Required Resources

Library	RAM [kB]	ROM[kB]
frame_na7_db_fl.lib	1.3	15.7
frame_na7_db_ir.lib	0.0	7.8
misc_na7_db_fl.lib	0.1	1.0
misc_na7_db_ir.lib	0.0	0.2
tif_na7_db_fl.lib	0.6	7.0
tif_na7_db_ir.lib	0.0	0.8
osx_na7_db.lib	0.1	0.9

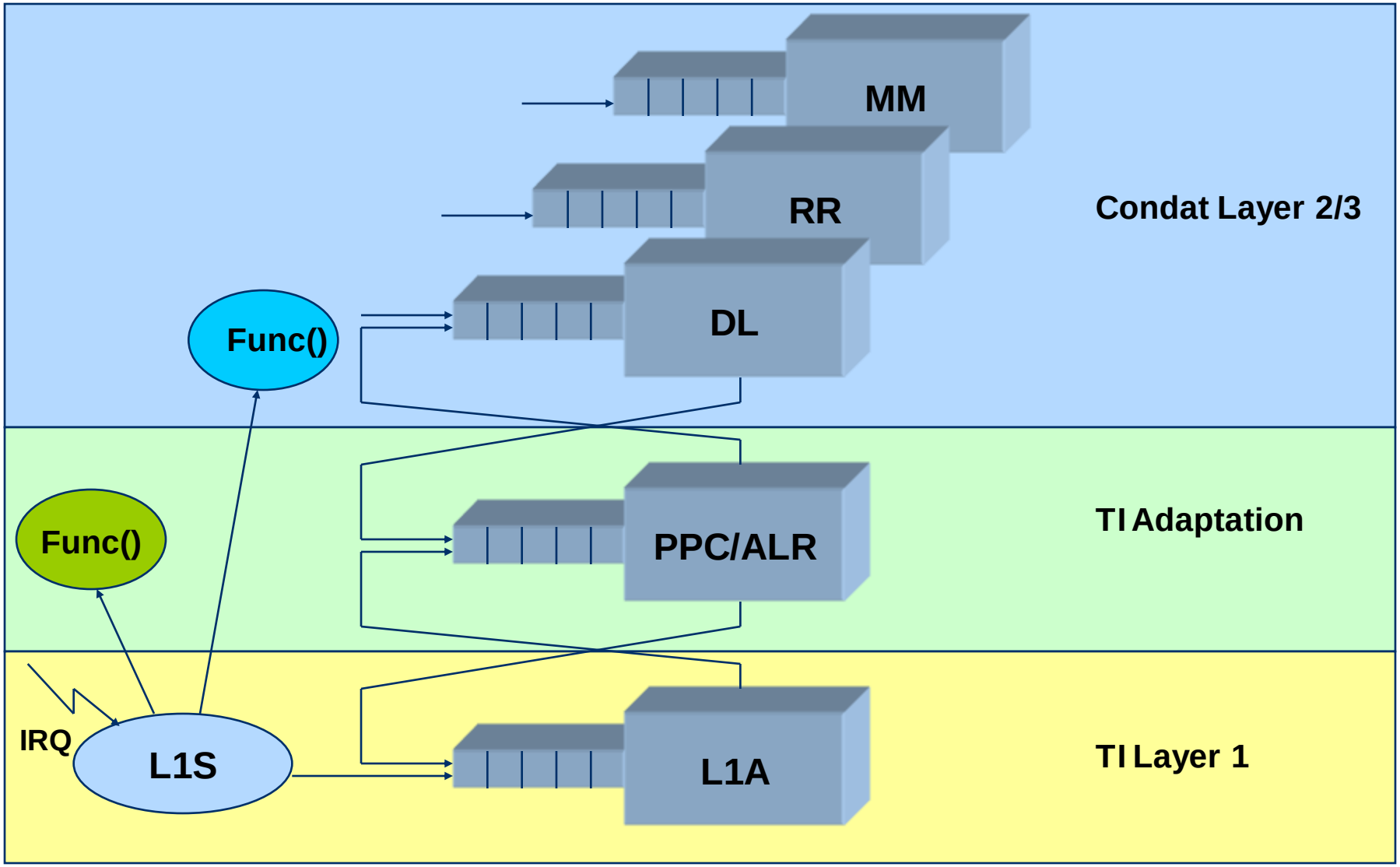
Frame Body Concept

- Overview
- Frame Functionality & Interfaces
- Test Interface
- System Setup
- **Layer 1 Interface**

Frame Body Concept - Layer 1 Interface

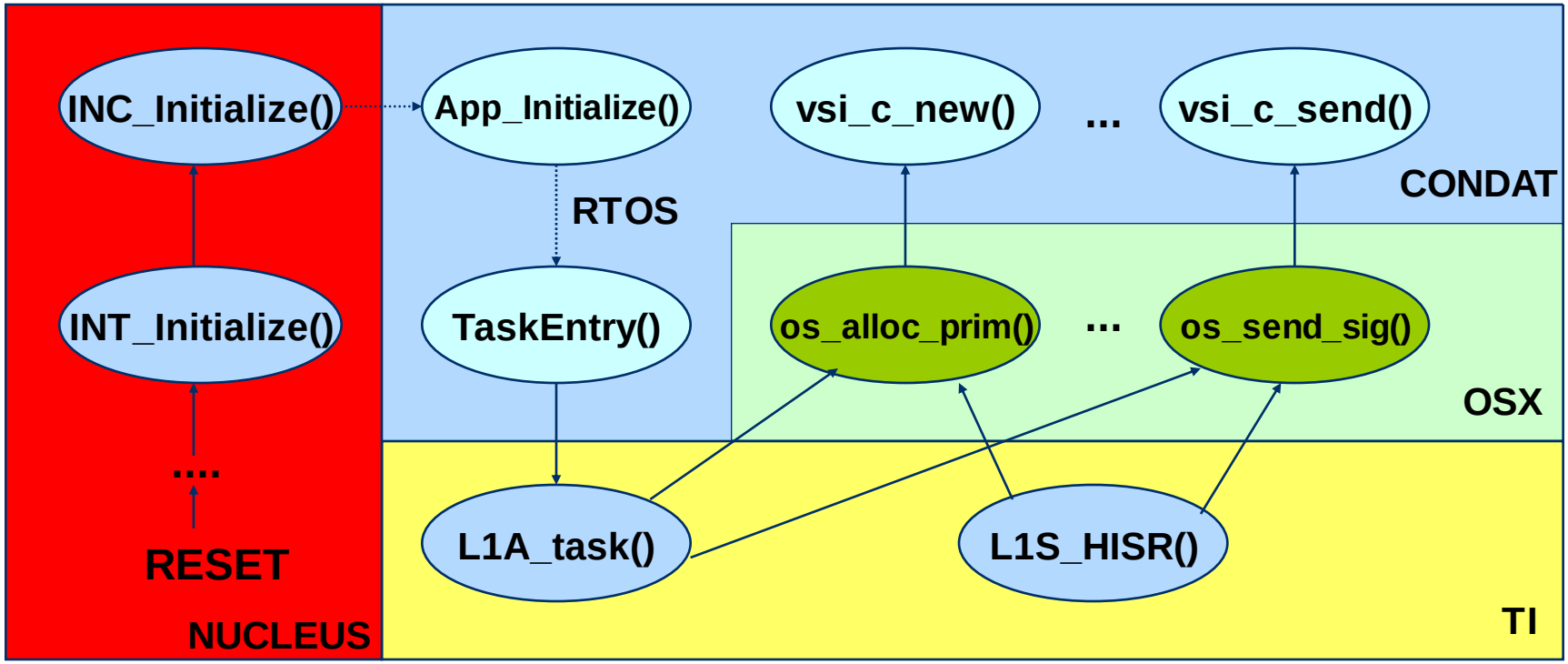


Frame Body Concept - Layer 1 Interface



Frame Body Concept - Layer 1 Interface

Startup and L1 Frame Access



Frame Body Concept - Documentation

- Frame Users Guide 8434.100
- VSI/PEI Interface Description 8415.033
- OS Interface Description 8415.036

END