TotaleNodeB Common Platforms

Software Architecture Document

Version 0.1

Contents

Figures

Table

Revision History

Revision	Date	Author	Comments
0.1	Nov 15, 2012	Guru Prasad	Draft version

1 Introduction

1.1 Purpose

This document describes the software architecture for components categorized as and under Common Platforms of the Radisys' TotaleNodeB (TeNB) solution.

1.2 Scope

This document does not intended to the primary reference for the entire system software architecture of TotaleNodeB solution. However this document identifies the various software components grouped part of Common Platforms and provides the software architecture details for the same.

This document would supplement the primary ‘Radisys_TEnodeB_Software_Architecture’ document.

1.3 References

1.3.1 Abbreviations

The following table lists the abbreviation used in this document.

Table 1 Abbreviations

Acronym	Description
OAM	Operation, Administration and Maintenance
CWMP	Common WAN Management Protocol
CM	Configuration Management
FM	Fault Management
PM	Performance management
HeMS	Home eNodeB Management System
KPI	Key Performance Indicators
CL	Convergence Layer

1.3.2 Standards

1. [TR-196], “Femto Access Point Service Data Model”, Issue 2, November 2011
2. [TR-069], “CPE WAN Management Protocol”, Issue 1, Amendment 4, July 2011

1.3.3 Shared Documents

None.

1.3.4 Proprietary Documents

1. FRS Total eNodeB
2. Radisys_TEnodeb_Software_Architecture
3. [FSRS Config Management]
4. [FSRS Performance Management]
5. [FSRS Fault Management]
6. [Data Model] Radisys_Data_Model
7. 3GPP IPSec Requirements, 3GPP TS 33.234

2 System Overview

The overall system architecture is as follows. This shows how the HeNB fits into the LTE cellular network.

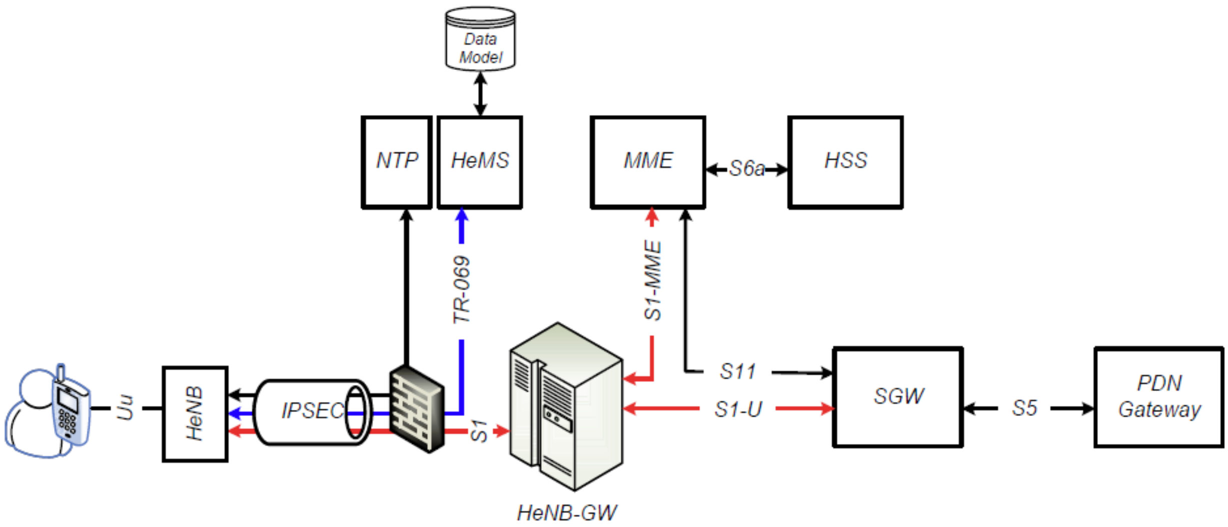

Figure 1 System Overview

2.1 Network Elements

2.1.1 Home eNodeB (HeNB)

Evolved Home NodeB (HeNB): HeNB enables the UEs to connect to the mobile operator’s network via the LTE Uu air interface. The interface to the core network has a HeNB-GW in the middle.

2.1.2 Security Gateway (SeGW)

It provides a secure connection from the HeNB to the operator’s core network. The SeGW is a logical node per the 3GPP specifications and it may or may not be collocated with the HeNB-GW (if one is in existence at the network). The IPSEC tunnel is terminated at the SeGW. The NTP server, HeNB-GW and HeMS are in the operator core and behind the SeGW.

2.1.3 HeNB Gateway (HeNB-GW)

It is a concentrator of S1 connections towards the MME and SGW. It enables an operator to mask the underlying HeNB RAN by concentrating their S1 connections and exposing a single S1-MME towards the MME, the SGW has a direct tunnel to the individual HeNBs.

2.1.4 HeNB Management System (HeMS)

The HMS provides the full configuration service and management of the HeNB using the TR-069 protocol. It uses the TR-196 standard data model for HeNB configuration management. HMS also provides the performance management and fault management functions for the HNBs. The HeMS service structure allows the operator to provide or receive “capability information, control information, configuration information, transport information, radio candidate lists, user / access management” in addition to managing the fault, performance and configuration metrics on the device.

2.1.5 MME

The mobility management entity hosting the following functions:

NAS signaling, NAS signaling security, Access Stratum security control, Inter core network signaling for mobility, paging the UE, Tracking area list management, PDN gateway and Serving gateway selection, MME selection for handovers, Authentication and Bearer management functions.

2.1.6 SGW (Serving Gateway)

The Serving gateway entity hosting the following functions:

The local Mobility Anchor point for inter-eNB handover, Mobility anchoring for inter-3GPP mobility, E-UTRAN idle mode downlink packet buffering, initiation of network triggered service request procedure, Lawful Interception, Packet routing and forwarding, Transport level packet marking in the uplink and the downlink, Accounting on user and QCI granularity for inter-operator charging.

2.1.7 PDN Gateway:

The network element hosting the following functions:

Per-user based packet filtering (for example, deep packet inspection), Lawful Interception, UE IP address allocation, Transport level packet marking in the downlink, UL and DL service level charging, gating and rate enforcement, DL rate enforcement based on APN-AMBR

2.1.8 NTP server

The NTP server is in the operator’s network. The NTP interface is used between the NTP client in the HeNB and the NTP server. Communication to the NTP server is normally established over a secure connection. NTP server is used to discipline the frequency of the HeNB system. NTP server is also used to adjust the HNB system clock.

2.2 HeNB interfaces to other network elements

The following are the external interfaces of the HeNB needs to provide to the other elements of the operator network.

OAM interface between the HeNB and the HeMS

The interface to the HeMS is TR-069 based and the data model to be supported is for the LTE FDD CPE. The OAM interface is secured from the HeNB to the SeGW (within the operator core). The HeMS is in-charge of configuring the HeNB and controlling its operation.

2.2.1 HeNB to SeGW secure Interface via Ethernet

Since the connection between the femto and the operator core is via the public network, securing of the IP connection is essential. Per 3GPP, the femto RAN connects to the operator core elements (such as HeMS, EPC nodes) via a security gateway. This security gateway (SeGW) terminates the IPSec connection from the femto. The femto is the other peer for the IPSec tunnel and hence is the termination point for the IPSec connection.

2.2.2 NTP interface

The NTP server is in the operator’s network. The NTP connection is normally established over a secure connection. It is used by the NTP server to synchronize the HeNB system clock and the frequency of the DAC oscillator within the accuracy of +/- 0.25 ppm. The success of the frequency synchronization depends on the NTP server performance and the network response time or delay and jitter.

2.2.3 LTE Uu interface

The LTE Uu interface has the following layers / sub-layers. PHY, MAC/RLC/PDCP and RRC.
See [2] for details.

2.2.4 HeNB to HeNB-GW (MME/SGW) interface

The S1 interface conceptually is split into S1-C (also referred to as S1-MME), a reference to the control plane and S1-U, a reference to the user plane. The S1-C and S1-U interfaces are the same from the HeNB perspective regardless of whether there is a HeNB-GW on route to the core network. The S1 interface carries the S1AP protocol and e-GTP protocol. S1 is secured from the HeNB to the SeGW (within the operator core)

2.3 HeNB system level functions/phases

System level tasks and states/phases are identified as follows. Considering these as internal requirements, HeNB needs to handle these tasks to fulfill the required system functionality.

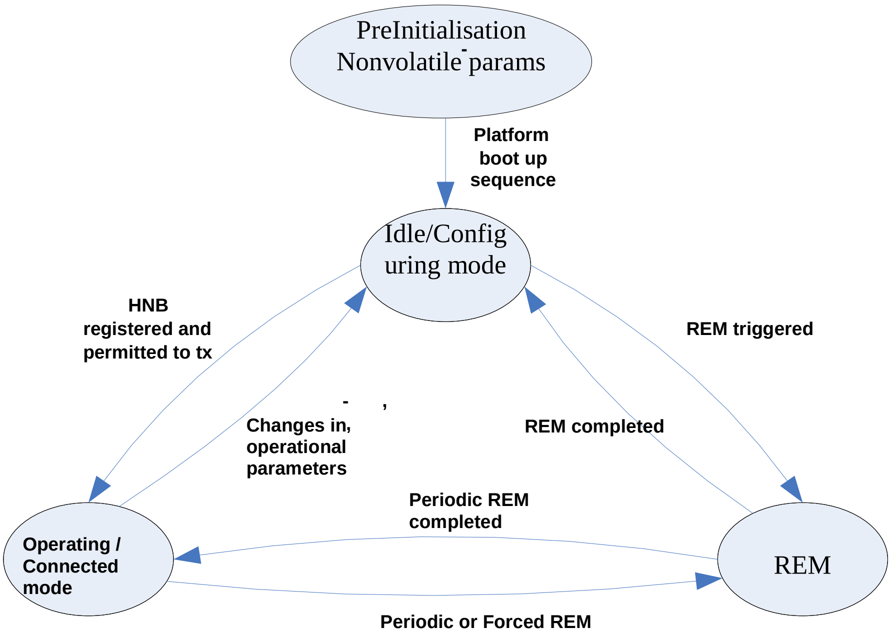

Figure 2 HeNB system level functions/phases

3 HeNB architecture

This section introduces the Common Platform functional blocks within Radisys HeMS system and the interfaces between them.

3.1 Functional block diagram

The following figure gives an overview of Radisys HeNB solution.

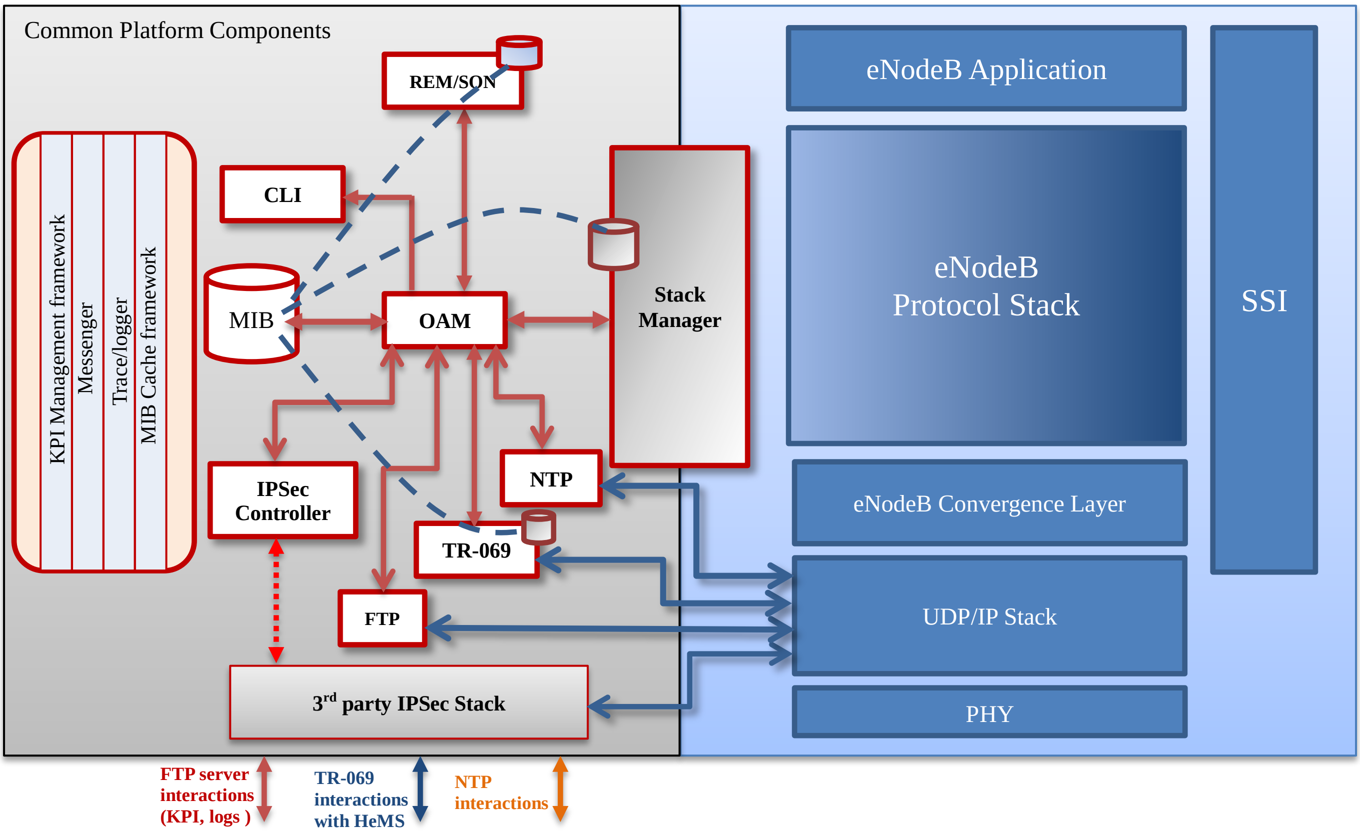

Figure 3 Functional Block Diagram

The HeNB system consists of '*Trillium core stack components'* and *'Common Platform components*'.

The *Trillium core stack components* comprises of

- Trillium Call Control Applications and FSM
- 3GPP compliant Trillium eNodeB Protocol stack
- Trillium Silicon-Specific Convergence Layer
- Third Party provided UDP/IP Stack and PHY

Common Platform components comprises of

- Common Framework and Infrastructure components
- OAM FSM and control module
- TR-069 client module
- Command Line Interface module
- IPSec Controller module
- NTP client
- Management Information Base module (Data Model repository)
- REM Controller and SON module.

3.1.1 Radisys HeNB core stack components

Refer Radisys Total eNodeB Software Architecture document [2].

3.1.2 Radisys Common Platform components

The Common Platform components provide essential functionalities beyond the core stack components functionalities. These do not directly contribute for the S1 or the Uu interfaces however they provide HeNB specific functionalities and additional standard interfaces primarily for managing the HeNB, providing secure transport.

These components include:

3.1.3 Common Framework and Infrastructure components

These components provide the basic building blocks for the platform components. In a way these components are essential for running the other functional blocks. It includes the following:

Messenger - Provides a mechanism for inter process communication between components (system applications) like OAM, TR-069, CLI and so on. The messenger is designed to provide UDP Post-Office like messaging service. Each of the platform application components are required to register with the messaging entity and each application is assigned specific ports to listen.

KPI management framework – Provides mechanisms for building a list of supported Key Performance Indicators. It also provides ways to distribute the KPI collection across the system components. The complex tasks of KPI value collation/aggregation from various collectors is simplified by Kpi-Manager and Kpi-Collector functionalities.

Trace/logger – These set of modules and macros provides common interface for adding traceability and debugging-ability for the components. This module provides various functionalities such as file-logging, udp-port logging, log file-rotation, trace-level, and trace-categories and abstracts the complexity from other modules.

Mib-Cache framework – This framework helps keep a consistent configuration data across the system components. The user applications have the option to subscribe to a subset of parameters in the main Management Information Base (MIB or DataModel repository). This framework updates the subscribers when the appropriate parameters change. This framework uses the Messenger interface as communication base.

Start-up scripts - A set of scripts takes care of system startup, secure tunnel setup indication and so on.

3.1.4 IPSec stack

HeNB interface towards the operator Core Network (HeMS, HeNB-GW, NTP server and so on) are over the wired, broadband network which is non-secure. IPSec stack provides secure IPSec connection (tunnel mode) as needed for securely exchanging control plane, user plane, TR-069 OAM messaging and NTP messages between HeNB and the core network infrastructure.

IPSec related modules include the

- 'IPSec-stack' - for managing the secure tunnel,
- Total eNodeB IPSec controller – IPSec stack configuration, handle tunnel creation, teardown, connection/disconnection indication, certificate management and so on.

3.1.5 OAM

This component is mainly responsible for initializing and configuring the HeNB. This module is the central controller within the HeNB and drives the top level state machine and tasks carried out in the HeNB. This module interfaces and controls various Common Platform components like TR-069, CLI, REM, IPSec Controller, NTP Client, FTP client controller.

OAM handles and choreographs many tasks including:

- During Start-up from power-on to eNodeB operational.
- Handle Admin Lock/Unlock – manage eNodeB radio
- REM scan – interfaces with REM module, handles boot-up scan, periodic scan, and forced scan requests.
- Interfaces with Frequency Sync module.
- Interfaces with IPSec Controller for configuration, creating/deleting tunnel.
- Manages IPSec digital certificates.
- Manages KPI periodic polling from KPI-Collectors
- Manages alarms within in the system

OAM interfaces with the HeNB Core Stack via the 'StackManager module'. This interface is used to initialize Core Stack components. This interface is also used to control the eNodeB operational state.

3.1.6 TR-069

This component provides the TR-069 protocol support interfacing the HeMS. This interface is a secure connection and helps exchange configuration and control information between HeMS and HeNB. Various data-models are supported by the TR-069 stack and is mapped by the OAM component to the local configuration parameters and stored locally within HeNB.

The local storage of configuration and control parameters are designated as MIB (Management Information Base. This local storage (MIB) however is proprietary and not related to other standardized protocols/technology.

3.1.7 Command Line Interface (CLI)

HeNB CLI function is for use by the field test engineers and for lab debugging purpose. The CLI runs in the HeNB on the serial port, SSH and on the telnet. The serial port is not accessible from outside of the HeNB enclosure. Once the HeNB UART connector is connected with a serial interface console utility in a PC [for example, putty or hyper terminal in the windows] or a user logs in to the HeNB using telnet, the user gets a Linux bash shell prompt. The CLI in the HeNB is used to do the following (based on the main HeNB feature availability):

- HeNB configuration updates
- HeNB process or resource status query
- HeNB diagnostics tests
- HeNB software upgrade/downgrade
- HeNB alarm and fault monitoring
- HeNB KPI statistics query

The CLI interface also provides a help manual that lists the available commands for each component.

3.1.8 Logging

Various software components generate events which need to be logged

Logging/Trace is the logging/debug feature that writes trace information to local files and console. Optionally it allows remote trace functionality using UDP port.

Trace supports much functionality like:-

- Each application specifies a specific local file to write traces to.
- Generally trace files are located at a specific location.
- Trace files have standard file format, which makes it easy to identify date-time, application which created the trace file
- Log/trace files are rotated after they reach a pre-defined size (this is however configurable via the MIB)
- Trace macros are called by 'C' functions as well.

3.1.9 Stack Manager

Stack Manager Module (SM) is responsible for configuration and management of various HeNB Core-Stack components including the protocol layers of the eNodeB Protocol Stack. This module interfaces with OAM component using Messenger framework. SM receives HeNB configuration requests from OAM and applies them on the various Core-Stack components and protocol layers. This makes SM an important component in enabling complete HeNB Management and following lists the main functionalities of the same:

- Configuration/re-configuration of all the eNodeB layers
- Controlling eNodeB protocol layers
- Handling of faults generated by various HeNB components and protocol layers
- Manage KPI-Collector object(s) and Handling KPI pegging requests (and any additional processing required).

The interfaces to Stack Manager and Core-Stack components including protocol layers are well defined (refer [2] for more details).

3.1.10 REM/SON component

Radio Environment Measurement (REM) is needed for purposes like Location Verification, Neighbor List (NL) Configuration and Parameter value selection (Self Configuration). REM component includes the REM controller application and REM CL. REM controller interacts with

OAM component using Messaging framework for necessary configuration parameters and handles REM scan requests from OAM. REM Convergence Layer is the interface to PHY; it abstracts the PHY specifics from the controller.

REM scan is expected to be executed at the following timings:

- Very first (“out-of-the-box”) initialization of HeNB.
- Subsequent initialization (that is, reboot/reset)
- At periodic interval during the normal operation

Once the REM scan requests are completed it is indicated to OAM and the scan results made available. HeMS is updated about the scan results via OAM, TR-069 interface.

3.1.11 SON (Self Organizing Networks)

HeNB needs to support Self Configuration and Dynamic Optimization. These requirements arise from the fact that LTE Small Cells significantly increases the number of transmitting nodes in the network and it also means that the network operator has less control of these nodes. Many configurable parameters like physical cell ID, location ID and so on needs to be derived based on the surrounding radio environment. SON working along with REM controller aims to provide basic and advanced features including self-configuration, self-optimization and self-healing.

3.1.12 NTP/Frequency Synchronization

NTP provides an alternative to OTA (over the air) synchronization with the macro cellular network. This method is based on the timestamps on the received packets from the NTP server and the success depends on the network and NTP server performance. NTP packets are secured over IPSec tunnel.

NTP Common Platform module is responsible for interacting with the NTP server and implements NTP based frequency and system time synchronization algorithms.

OAM component is responsible for determining if OTA synchronization or NTP based synchronization needs to be employed for system synchronization. NTP client component interfaces with OAM using messaging interface.

3.2 Interfaces between functional blocks

3.2.1 Interfaces to MIB

The MIB is a central repository for the HeNB configuration and provides mechanism for defining and storing the configuration parameters and related meta-data. MIB also provides the initial (default or non-volatile) values that are needed for the setup and initialization of HeNB components including the Core-Stack components. Values in the MIB are set either by

- HeMS via a secure TR-069 interface or
- by the field test engineers and developer for debugging purpose via CLI or
- by the default or non-volatile values stored/available within the system

The HeNB platform has designed so that the MIB could potentially be configured from the HMS via a different messaging protocol to TR-069 if required. Within the HeNB system the TR-069 component translates parameters specific to HeNB to the standard TR-196 data model in a TR-069 messaging structure.

3.2.2 Interface between OAM and HeNB core stack components

OAM being the central Configuration management module runs the top level state machine for HeNB, interacts with HeMS via the TR-069 component and configures HeNB components including Core-Stack Components via Stack Manager.

SM by virtue of using the Messenger interface has a Message Queue (a queue which holds an instance of a class to be specific). OAM and SM are separate applications (processes) and communicate using Messenger interface which implements the transport mechanism.

SM has access to a local MIB-Cache which provides faster access to configuration parameters and gets notified when any of the subscribed parameters get updated.

SM handles this parameter/attribute update indication and performs appropriate action to pass on the change in configuration to the TeNB Application. TeNB Application is responsible to configure/reconfigure the components and protocols using well defined interfaces.

3.2.3 Interfaces to REM controller

OAM directs REM specific configuration parameters (which are either default/non-volatile or pushed by HeMS or via CLI) to REM controller via the MIB-Cache both during REM initialization and any dynamic updates. OAM also requests REM scan using Messenger interface. REM controller updates scan results to the MIB before indicating the scan status to OAM via the Messenger interface. Message types for Scan request, Scan response are defined and identified by Message Serialization ID.

3.2.4 Interfaces to NTP client component

OAM component has a sub entity called 'FreqSyncEntity' which is designated to interact with NTP client application using Messaging interface. FreqSyncEntity also implements a state machine. As OAM drives the state machine for the HeNB and interacts with REM controller, OAM decides based on the REM scan results whether NTP based frequency synchronization to be enabled or if OTA synchronization is sufficient. Also based on the RF tuning results OAM initiates control to adjust the system oscillator.

3.2.5 Interfaces towards Hardware

The following Common Platform Components need interfaces to the Hardware/PHY.

REM CL - REM controller is dependent on REM CL (hardware specific convergence layer) to scan all macro and other Femto cells on intra-RAT as well as inter-RAT (WCDMA). REM CL directly interacts with the Hardware APIs to perform the above functionalities. It uses the vendor specified APIs, message formats and library to communicate with PHY.

FreqSyncEntity - This component needs to discipline the system frequency and clock. For this purpose it uses the PHY APIs or ioctl/system call.

IPSec HW Acceleration – IPSec components include:

Internet Key Exchange (IKE) protocol for setting up the IPSec connection including initial authentication and negotiation of security associations with IKE peer, Security Association Database, IPSec Stack.

Acceleration is implemented for IKE as well as for ESP encryption. In both cases IPSec components need to interact with the Hardware accelerator chipset via drivers and crypto APIs provided by the hardware vendor.

4 Software

This section outlines the details of the software architecture for the various components discussed in the previous sections.

MindSpeed platform has Lower Arm and Upper Arm each having multiple cores.

4.1 Lower Arm

LowerArm runs 4GMX operating system and has total four cores.

Core-Stack layers like PDCP, RLC, MAC, and Convergence Layer run on Lower Arm.

4.2 Upper Arm

Upper Arm runs Linux operating system and has two cores.

Core-Stack layers like TeNB Application, S1AP, RRC, eGTP, SCTP, TUCL and so on run on Upper Arm.

Additionally Common Platform components are also run on Upper Arm.

4.2.1 Memory model

Common Platform components use the Linux user process model for memory management which uses separate memory space for each user space process. If any of the application components create threads then each of them would share the same memory space as that of the parent. No shared memory is used between the processes.

Stack Manager is an exception

4.2.2 Process Thread Details

Most of the Common Platform Components run as individual processes in the Linux User Space.

All of the processes run as single threads except Stack Manager.

Stack Manager runs in two separate threads. The main task identified as Entity ‘ENTSM’ is a TAPA task which creates an independent system thread (detached) for the SM-Messenger (SMM).

The sole task of the SMM is to initialize the application, register Message handler(s), initialize KPI-Collector, subscribe to MIB parameters, and register KPIs and KPI groups with the KPI Manger.

Once the initializations are done SMM processes the received events forever.

4.3 Linux kernel space modules

IPSec kernel module, IPSec HW accelerator driver.

4.4 Linux User Space modules

All Common Platform components run in user space including IPSec IKE application component.

radisys

www.radisys.com