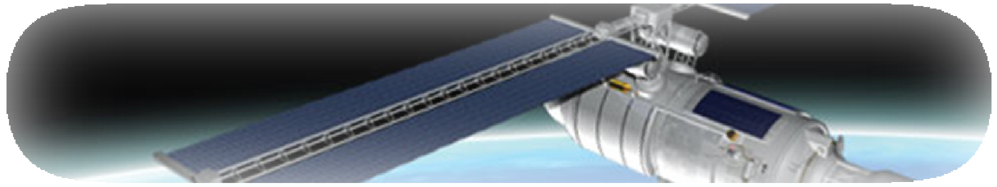
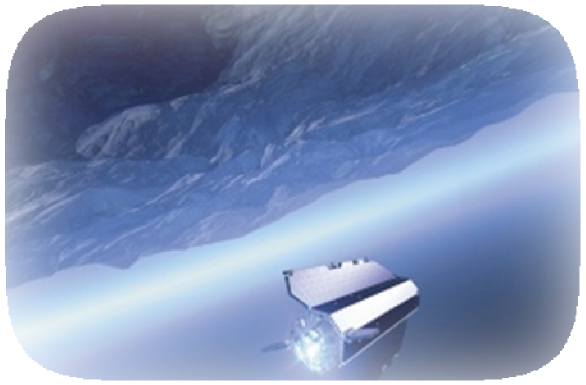


Plug'n'Play technology in SpaceWire network

A. Setkov, K. Khramenkova, L. Koblyakova

Overview

- Plug’n’Play necessity
- SpaceWire overview
- RMAP – means of realization
- SpaceWire & Plug’n’Play interaction
- Plug’n’Play algorithm
- Summary



Plug'n'Play necessity (1 / 1)

Modern Network Systems are frequently consist of a considerable quantity of devices (nodes) allocated on different distances from each other.

It's necessary to form interconnection between them and to maintain working correctness during all work-time.

Expected that when all system devices had been connected, it will be able to discover and identify network structure by special algorithm, to set logical addresses to nodes, to set routing tables and other necessary system parameters and to provide with this information all others needed devices. At regular work connection and disconnection of new network devices recognized automatically.

SpaceWire overview (1/2)

SpaceWire standard (ECSS-E-ST-50-12C)

- Is developed by international group of specialists under the aegis of European Space Research and Technology Centre (ESTEC) of European Space Agency (ESA)
- Is being widely used on many space missions by:

ESA, NASA (USA), JAXA (Japan), Russian Federal Space Agency

- assigned to make communication networks for on-board systems

SpaceWire overview (2/2)

- Provides compatibility with different types of equipment and multifunctional usage of terminal devices and subsystems.
- Transmits data and control information for on-board systems. It's high-speed standard (2-400 Mbits per second)
- SpaceWire is rapidly becoming a de facto communications standard within the space industry
- Advantages: simplicity, speed, low-power and reliability
- SpaceWire network consist of 3 types of elements: node, link, router. Nodes are connected through router and admit to exchange data packets (messages)

RMAP – means of realization (1/2)

The Remote Memory Access Protocol (**RMAP**) has been developed to support a wide range of SpaceWire applications

- Primary purpose – to configure a SpaceWire network, to control SpaceWire units and to gather data and status information from those units
- RMAP may operate alongside other communications protocols running over SpaceWire
- RMAP may be used to configure and read the status of nodes on the SpaceWire network

RMAP – means of realization (2/2)

- **For simple SpaceWire units** without an embedded processor, RMAP may be used to set application configuration registers, to read status information and to read or write data into memory in the unit
- **For intelligent SpaceWire units** RMAP can provide the basis for a wide range of communications services

SpaceWire & Plug'n'Play interaction (1/3)

SpaceWire Plug'n'Play (**SpW PnP**) is a joint effort together with NASA and other partners in the frame of the SpW PnP Working Group

Plug'n'Play technology must fulfill the following conditions:

- The Plug'n'Play technology **must provide each device in the system with information** of what it needs to route and to interpret packets received from each other device in network
- The algorithms **must not result in undetermined results or deadlock**. Plug'n'Play must allow the network to work as well as possible for as long as possible. The effect of failures should be minimized

SpaceWire & Plug’n’Play interaction (2/3)

- Operating with **only two types of devices** (a router and node), we build big arbitrarily structured network and algorithm Plug’n’Play should work correctly on it
- Technology should provide ability, in future, to add new devices. Thus, there shouldn't be any overload in presents and Plug’n’Play **discovery and configuration should work quickly and shouldn't have great influence** on performance of a network
- This technology **should be implemented with minimal cost**; device equipment should take minimal place on a integrated circuit and minimal energy consumption

SpaceWire & Plug’n’Play interaction (3/3)

The main condition is that **SpW must be unchangeable**. Plug’n’Play is above SpW

Plug’n’Play isn’t standardized. All Plug’n’Play algorithms can be divided on two main types:

- **Centralized** – a network has one device, which initiates and controls algorithm Plug’n’Play
- **Decentralized** – a network has several centers, which controls algorithm Plug’n’Play. In this case, if one or some configuration devices are out-of-work system is workable

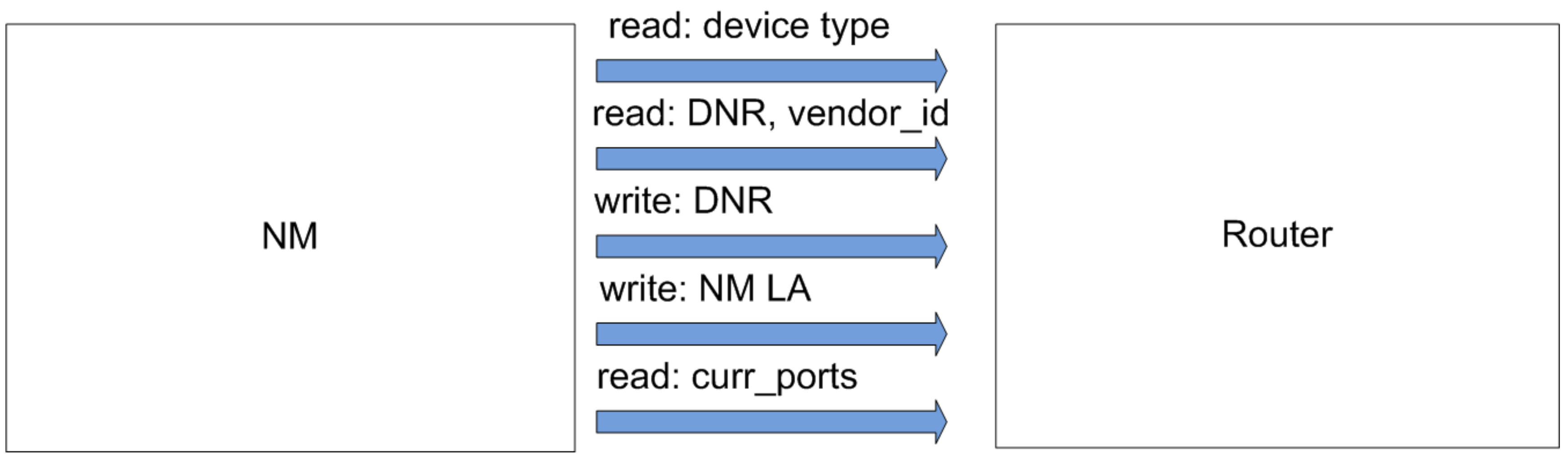
Creating SpaceWire Plug’n’Play technology, we suppose that:

- Network structure is arbitrary and unknown and can change at any time due to failures or user intervention
- New devices plugged may not be in reset status

Plug'n'Play Algorithm (1/11)

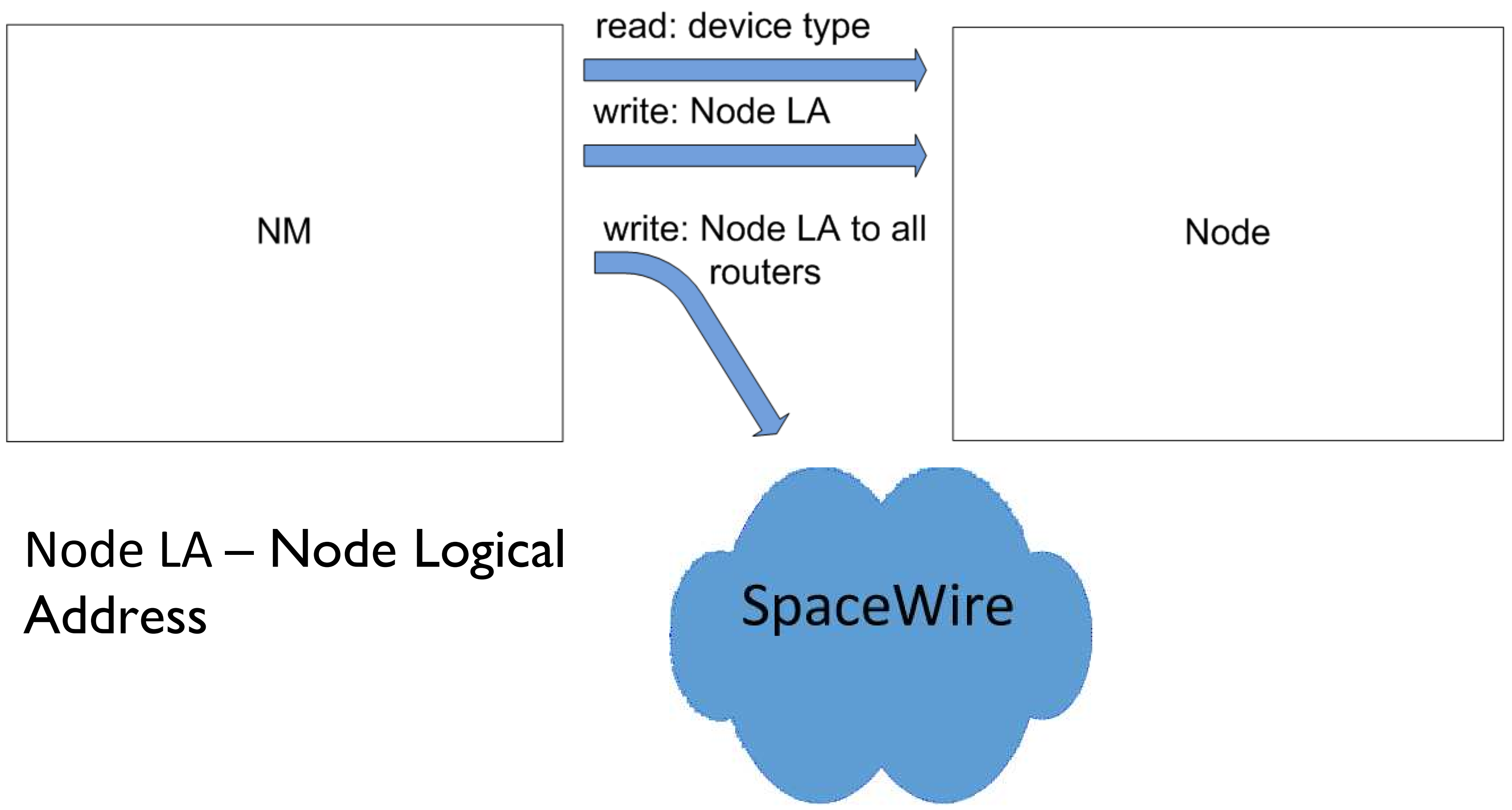
- Network discovery shall be executed by an intelligent node, called **Network Manager**, which detects plug/unplug events of any SpaceWire link, device or subnet and uniquely identifies all devices in the network
- NM can be implemented as PC, node or other SpW device
- RMAP protocol is using to configure remote devices

Plug'n'Play Algorithm (2/11)



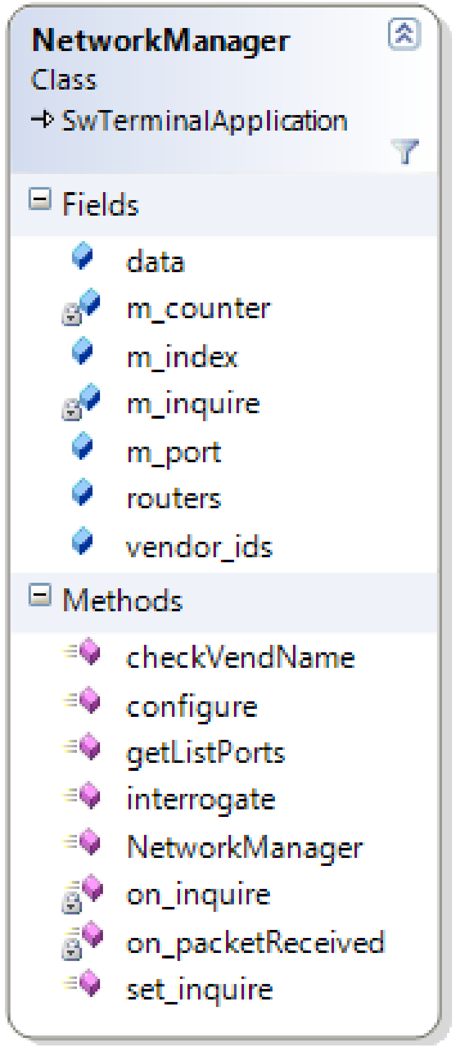
device_type – router or node
vendor_id – device identifier (implemented by vendor)
DNR – Discovery Nerwork Register – indication, that device has been already configured
NM LA – NM Logical Address
curr_ports – list of active ports

Plug'n'Play Algorithm (3/11)



Node LA – Node Logical Address

Plug'n'Play Algorithm (4/11)



NetworkManager – NM Application

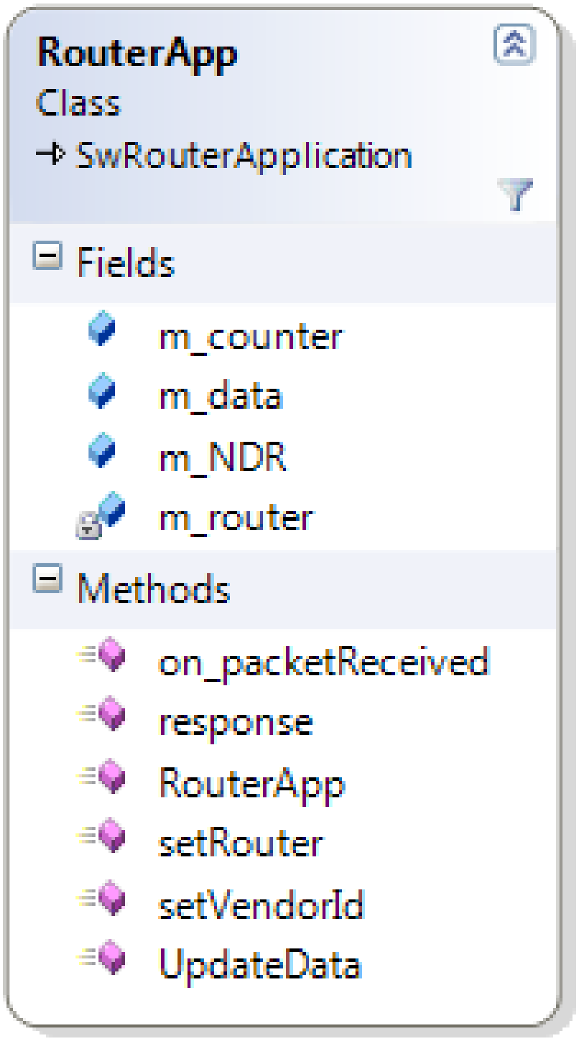
configure – generates and sends configuration packets

interrogate – generates and sends interrogation packets

on_inquire – interrupt handler, starts network discovery

on_packetReceived – interrupt handler, packet received

Plug'n'Play Algorithm (5/11)


RouterApp
Class
→ SwRouterApplication

Fields
- m_counter
- m_data
- m_NDR
- m_router

Methods
- on_packetReceived
- response
- RouterApp
- setRouter
- setVendorId
- UpdateData

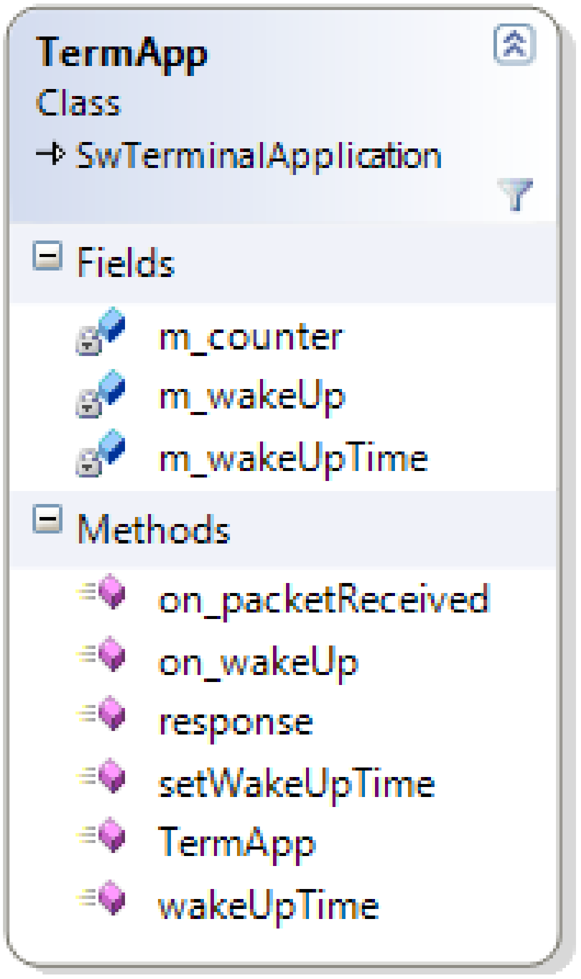
RouterApp – Router Application

on_packetReceived – interrupt handler, packet received

response – generates and sends response packets

UpdateData – updates program-accessible memory

Plug'n'Play Algorithm (6/11)


TermApp
Class
→ SwTerminalApplication

Fields
- m_counter
- m_wakeUp
- m_wakeUpTime

Methods
- on_packetReceived
- on_wakeUp
- response
- setWakeUpTime
- TermApp
- wakeUpTime

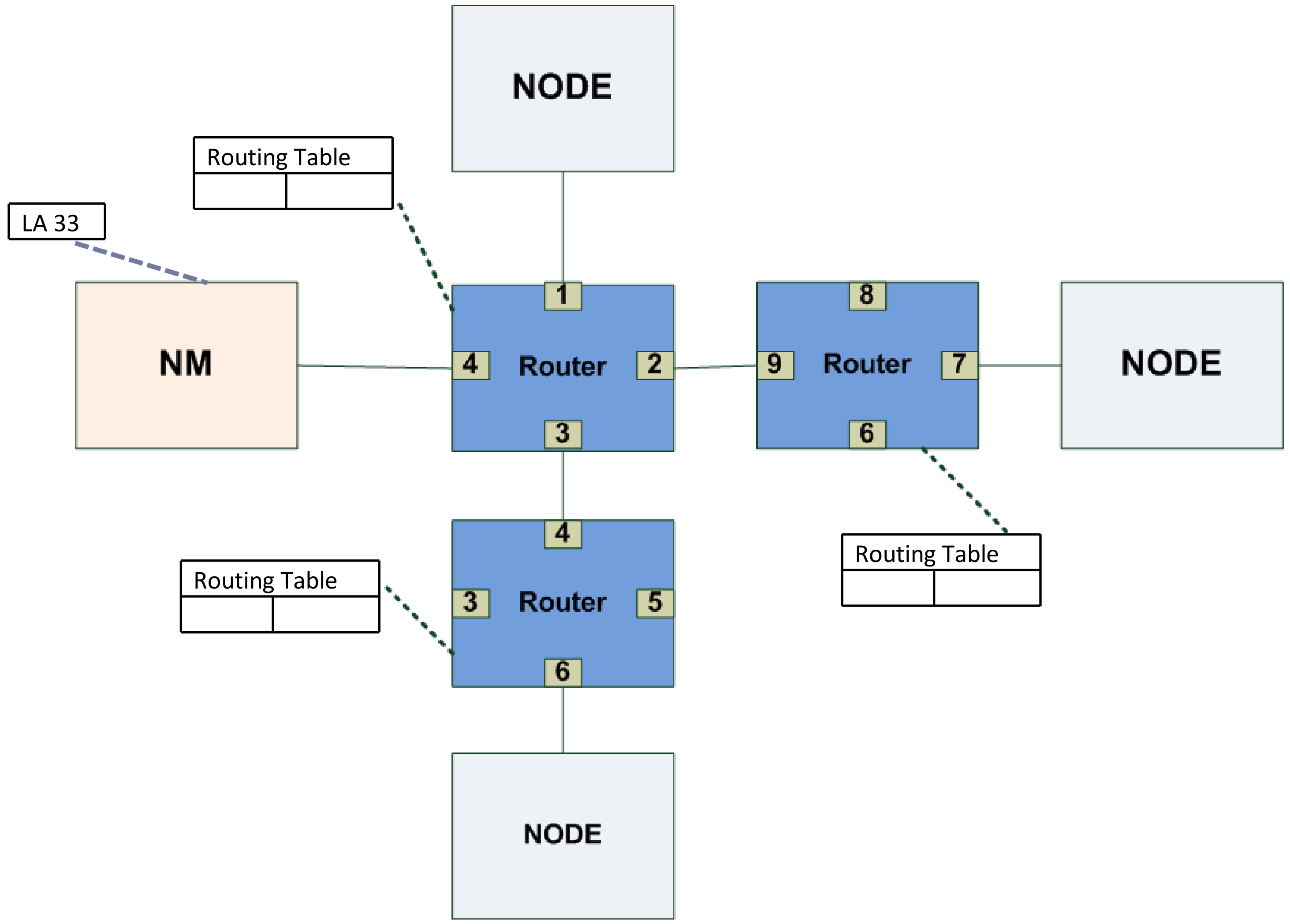
TermApp – Terminal Application

on_packetReceived – interrupt handler, packet received

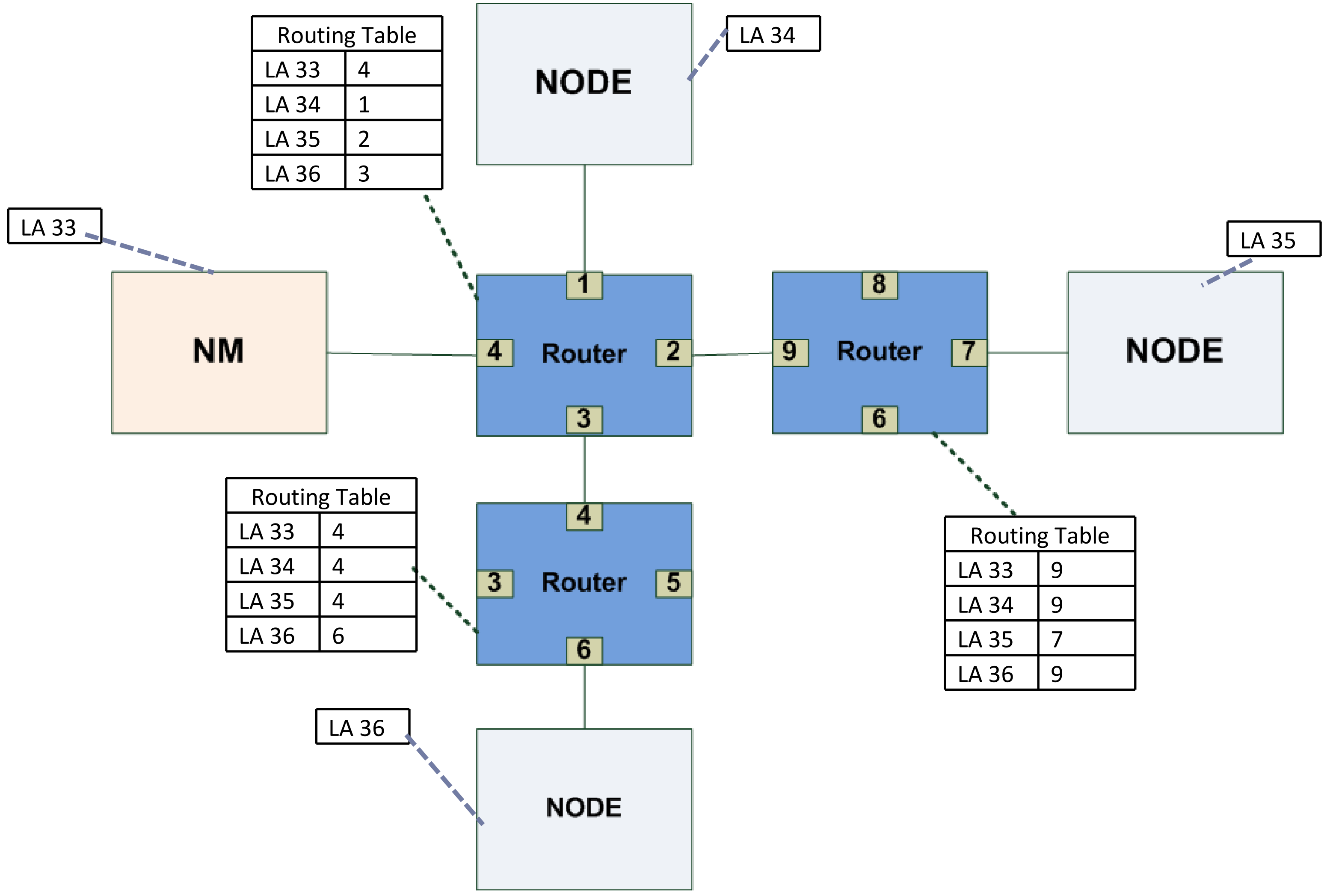
response – generates and sends response packets

On_wakeUp – on this command terminal «connects» to network

Plug'n'Play Algorithm (7/11)



Plug'n'Play Algorithm (8/11)



Routing Table	
LA 33	4
LA 34	1
LA 35	2
LA 36	3

Routing Table	
LA 33	4
LA 34	4
LA 35	4
LA 36	6

Routing Table	
LA 33	9
LA 34	9
LA 35	7
LA 36	9

Plug'n'Play Algorithm (9/11)

Requirements and limitations putted on our implementation:

- Devices should support sending-receiving RMAP packets, path and logical addressing
- Number of nodes in network is limited by Routing Table size
- Node can be connected only to one router
- Nodes should be set up in such way to be able to send response packet when it received an interrogation

Plug’n’Play Algorithm (10/11)

Usage of memory: each router uses Router table (hardware implementation), Notification List, Network Discovery Register (1 byte) (software implementation)

In NM allocated memory for storing router path addresses

Following variants are foreseen:

- Detaching device group
- Attaching device group
- Cycles

Plug’n’Play Algorithm (11/11)

Main aspects:

- Eliminates all manual configuration
- Only one NM is used in this algorithm
- Regionally-logical addressing is not used
- Router shouldn’t be programmed specially
- Using RMAP (there are no need to use new protocols)
- Users can manually configure any device just using RMAP commands

Shortcoming: when NM is out-of-working, devices can’t be uniquely discovered and configured.

Summary

As a result: we have configured network consisting of nodes and routers and using one NM

NM admits to identify attached/detached devices in automatic mode, without any human intervention. Thus, we can configure any network, which applied to our demands and limitations. at the expense of simultaneously sending packets to all active ports device identification and setting performed quickly

Current working:
Algorithm is debugging on modeling system NetSim