ITSSv6 - IPv6 ITS Station Stack
for Cooperative ITS FOTs
http://www.itssv6.eu/

ICT-2009.6: ICT for Mobility,
Environmental Sustainability
and Energy Efficiency

ITSSv6 Deliverable

ITSSv6 STREP Grant Agreement 210519
D5.1: Initial Porting and Integration Guideline

DATE	22nd May 2012
CONTRACTUAL DATE OF DELIVERY TO THE EC	M12 (31.01.2012)
ACTUAL DATE OF DELIVERY TO THE EC	M14 (31.03.2012)
EDITOR, COMPANY	András Váradi, lesswire AG
DOCUMENT CODE	ITSSv6-D.5.1-Initial_Porting_and_Integration_Guideline-v1.1
SECURITY	Public

Project Coordinated by Thierry Ernst - Mines ParisTech / Inria
E-mail: thierry.ernst@mines-paristech.fr

Document History

Release	Date	Reason of change	Status	Distribution
1.0	01/04/12	Final consolidated version	Final	Internal
1.1	01/05/12	Alignment with other deliverables	Final	EC

Contents

List of Figures

List of Tables

Executive summary

This document is meant to serve as a starting point for ITSSv6 stack integration and porting. It summarizes the work of WP5 over the first year of the ITSSv6 project. The task of WP5 is to examine and evaluate most of the available or already used hardware platforms and communication technologies. Based on this research define a minimum and recommended scenario, and suggest an ideal range of host systems. The Basic Open Platform named in this document is the starting point of such measurements. In parallel, a specific embedded target platform is also chosen to act as a primary device for field tests and to ensure portability of the code.

Though it is a public document available to a large audience, the reader should be reminded that as a deliverable, the first purpose of this document is to report to the European Commission the output of the Work Package 5 "Porting to FOTs".

Also among the goals of our work package is the deployment of the software provided by WP3 and evaluated by WP4 into specific hardware platforms. As the ITSSv6 project aims to support different wireless mediums and host systems, it is essential to provide training material, guidelines and support to any third party development group that intends to use the output of the project. This document is the first ("Initial") guide and therefore includes only a limited part of the intended final versions material and planned content. As porting is still under way by the time of this deliverable, it only deals with one STP: the embedded Laguna platform.

Due to available resources of partners the porting guideline will stay within the realm of Linux based systems. Nevertheless porting to other e.g. automotive grade operating systems is possible if the listed hardware requirements are met.

Contributors

The following ITSSv6 members have contributed to this deliverable:

- Thierry Ernst, from Mines ParisTech / Inria.
- Wilfried Lohmann, from lesswire AG.
- Roland Oberhammer, from Bluetechnix.
- András Váradi, from lesswire AG.
- José Santa, from University of Murcia.
- Antonio F. Skarmeta, from University of Murcia.
- Miguel A. Zamora, from University of Murcia.

CHAPTER 1

Introduction

1.1 The ITSSv6 project

ITSSv6 (IPv6 Station Stack for Cooperative Intelligent Transportation Systems (ITS) Field Operational Tests (FOTs) is an European Project (STREP) from the 7th Programme Framework (Grant Agreement 270519), Call 6 (ICT-2009.6: ICT for Mobility, Environmental Sustainability and Energy Efficiency) started in February 2011 and lasting until January 2014. The project is coordinated by Institut National de Recherche en Informatique et en Automatique (Inria) and gathers Universidad de Murcia (UMU),Institut Mines Telecom (IT), lesswire (LW), Magyar tudomanyos akademia szamitastechnikai es automatizalasi kutato intezet (SZTAKI), Schalk & Shalk OG (IPTE) and Bluetechnix Mechatronische Systeme GmbH (BT) (Bluetechnix in short).

The objective of the ITSSv6 project is to deliver an optimized IPv6 implementation of ETSI / ISO ITS station reference architecture. ITSSv6 builds on existing standards from ETSI, ISO and IETF and IPv6 software available from the CVIS and GeoNet projects. The IPv6 lTS station stack provided by ITSSv6 supports at least 802.11p and 2G/3G media types and is configured differently according to the role played by the ITS station (roadside, vehicle, central). The tasks of ITSSv6 are to:

- Gather third party users into a common User Forum to collect user's requirements;
- Enhance existing IPv6-related ITS station standards and specification of missing features;
- Implement IPv6-related ITS station standards;
- Validate the implementation and assess its performance;
- Port the IPv6 ITS station stack to selected third party platforms;
- Support third parties in the use of the IPv6 ITS station stack.

Publishable material of the ITSSv6 project (public deliverables, newsletters, presentations made at conferences or workshops, information about events) are made available on the project's web site (http://www.itssv6.eu) as it becomes available.

1.2 Purpose of this deliverable

This deliverable describes the basic open platform and reports the software analysis of the IPv6 ITS Station stack so that the IPv6 ITS Station stack could be ported easily to various specific target platforms.

It includes the conclusions made during tasks T5.1: Definition of the target platform(s) and T5.2: Software analysis, furthermore describes and assists porting as a result of the work performed in T5.3 Porting to target platform(s).

Taking into account needs from prospective third parties users (FOTs and projects) identified in T6.1, an intel-based basic open platform (BOP) convenient to the ITSSv6 partners (hardware, Linux OS version, price, etc.) is defined in order to ease portability and to benefit to a wide number of third party users (MS52). More specific target platform(s) (STP) are defined according to requirements expressed by selected third parties identified in T6.1. This work is reported in D5.1 and revised in D5.4.

All modules composing each version of the IPv6 ITS Station stack developed in T3.3 are investigated in order to identify system dependent parts. A clear definition of the requirements on the used libraries and interfaces is produced. The software is restructured to separate the system dependent parts from the system independent parts. Constant feedback is provided to WP3 in order to avoid late and unfeasible modifications (MS53 & MS56). This work is reported in D5.1 and revised in D5.4.

Porting to STP(s) is performed in relation with selected third parties once the validation of the IPv6 ITS Station stack on the BOP is concluded in T4.2. To that end the system definition of the target platforms is investigated. Critical and system dependent parts of the IPv6 ITS Station stack are adapted to the target platforms. The task is performed in close relation with WP3 should there be any necessary bug fix. A detailed report of the porting exercise and a systematic description of all problems encountered is produced. The guideline will be useful for porting to other target systems. The package and the guideline are released at least twice (MS54 & MS57) and presented as D5.1, D5.2, D5.4 & D5.5. Non technical details of this activity are reported in D5.7.

1.3 Objectives and contributions

The main target of this deliverable to serve as a guide for developers who intend to build upon the ITSSv6 Software stack. In order to do so, one must understand the basic concept and structure of the projects outcome both from the software, as from hardware side. This document intends to serve as such guideline to help understand and modify our research outcome for a specific purpose and role by porting it to a different system. Bluetechnix and Lesswire specialises in the modification of the code and the determination of its hardware requirement. This work is supported by SZTAKI who is responsible for the main software base.

1.4 Structure of the document

The present document is structured as follows:

- Chapter 1 introduces the ITSSv6 project for the reader;
- Chapter 2 gives a quick overview of the system elements;
- Chapter 3 deals with milestone MS52: Basic open Platform definition and its conclusion;

- Chapter 4 gives a detailed explanation on the research done for WP5 on targeted radio and host systems;
- Chapter 5 displays the milestone MS53: Initial Software analysis;
- Chapter 6 consists of the initial porting guideline;
- Appendix C provides the list of acronyms;
- References are provided at the end of this document.

CHAPTER 2

System Introduction and Requirements

This chapter provides a short system overview including the related documents for easy use of this Porting Guideline. Detailed descriptions of each element are published in the following chapters.

2.1 The ITSSv6 Software package

ITSSv6 builds on the base of existing standards from ETSI, ISO and IETF and IPv6 software available from CVIS and GeoNet projects. Its main objective is to deliver an optimized IPv6 implementation of ETSI/ISO's ITS station reference architecture.

This software package is presented in Deliverable D5.2 "Initial IPv6 ITS Station Stack Package" and includes a limited number of features chosen for the first release. As part of the methodology of the project, features were classifed into three classes: short term, medium term and long term. The current system integrates the short term features. Features are also categorized by target platform and role. Each target platform can assume the role of any ITS Station node, or they can operate as combination of nodes such as Host/Router combinations (i.e., Vehicle ITS Station nodes need different software components than Central ITS Station nodes). The base system includes features for all roles, but as porting effort resources are usually limited, only the required elements should be chosen for integration. Table 2.1 lists available features and roles.

2.2 Software requirements

The IPv6 lTS station stack provided by ITSSv6 supports multiple development and embedded platforms. It also implements multiple media interfaces such as 802.11p or 2G/3G and is configured differently according to the role played by the ITS station (roadside, vehicle, central).

Class-1 Feature	Vehicle ITS Station		Roadside ITS Station		Central ITS Station	Central or Roadside ITS Station
	Host	Host/Router	Host	Host/Router	Host/Router	Border Router
Mobility management						
Mobility support (NEMO, MCoA)		X			X	
Routing						
IPv6 over GeoNetworking		X		X		
Internal Network Prefix Discovery (INPD)		X		X		
Multicast						
MLDv2		X		X		
Security and authentication						
IPSec		X			X	
IKEv2		X			X	
IPv4-IPv6 transition						
Softwire		X		X		X
OpenVPN		X		X		X
DNS64/NAT64					X	X
Media						
2G/3G		X		X		
802.11a/b/g/n	X	X	X	X	X	X
802.11p		X		X		

Table 2.1: D3.1's table of features implemented in ITS Stations

CHAPTER 3

Basic Open Platform (BOP) Definition

The following chapter is the output of MS52 ITSSv6 Basic Open Platform Definition. It defines the main characteristics of the required host hardware and interfaces, taking into account the needs of prospective third parties users (FOTs and projects) identified in T6.1.

3.1 Core elements

Defining the base of the ITSSv6 stack's underlying hardware is the most important milestone, therefore the first to be discussed. As the target software feature set and performance is obviously not reached during the beginning of the project, the target systems must have a relatively high performance. This is a key factor to assure that future software services have sufficient computing performance at their disposal.

3.1.1 Intel-based platform

The intentional use of the x86 platforms is for system and application development. For this purpose we introduce a recommendation in table 3.1. Please note that as this platform is intended for coding, overachieving these values is advised and will result in lower compilation times.

Platform component	Minimum requirement	Recommended requirements	Remarks
Processor architecture	x86 compatible		other x86 compatible non-Intel platforms may also be considered
Clock rate	800Mhz		
RAM memory	1024 MB		
Hard Disc or Solid State Disc memory	20 GB		
Communication interfaces	As defined in table 3.3		
Sensor interfaces	As defined in table 3.4		

Table 3.1: Basic Open Platform Requirements for x86

3.1.2 Embedded platform

Some of the target systems may be intel-based, but most of the devices running ITS stacks will be embedded platforms serving as Mobile Routers or Roadside units. As the computing performance of real life ITS stations used in European FOTs mostly does not reach the power of today's x86 processors, a second BOP is defined in table 3.2 for embedded devices. The ARM controller family is widespread in ITS systems and is therefore selected by ITSSv6 project as embedded BOP.

The ITSSv6 project intends to support and apply embedded platforms to demonstrate its awareness of real life environments.

Platform component	Minimum requirement	Recommended requirements	Remarks
ARM architecture	ARMv5 (ARM9)	ARMv6	
Clock rate / MIPS	400Mhz / 400 MIPS	800 MIPS	
RAM	64 MB	128 MB	
Flash memory	16 MB	32 MB	
Communication interfaces	As defined in table 3.3		
Sensor interfaces	As defined in table 3.4		

Table 3.2: Basic Open Platform for embedded ITS system on ARM platform

3.2 Communication interfaces

The Basic Open Platforms system core must be extended with internal or external wireless mediums to enable communication. ITSSv6 aims to support different technologies, especially for European ETSI standards but also for ISO and US norms. 802.11p has a primary importance due to its dedicated role. Table 3.3 displays these recommendations.

Extension slots (like mini PCI) should also be considered, but will not be listed below as their requirement depend on the (listed) function one needs to achieve through its interface.

Interface type	**Mandatory**	**Optional**	**Remarks**
802.11 p (ETSI ITS G5)	X		Dual channel support
802.11 a/b/g/n (2.4 & 5 GHz)	X		
Cellular 2G/3G	X		4G/LTE for future use
Satellite			out of scope
Infrared		X	
Microwave			out of scope
60GHz Millimeter-wave			out of scope
Ethernet 10 Mbps	X		10Mbps or better
CAN		X	
USB 2.0		X	USB high speed or better
SD card slot		X	

Table 3.3: Communication interfaces for BOP

3.3 Sensor interface

The following table (3.4) displays all suggested sensor interfaces that enable the platform to fulfill the role of an ITS station.

Sensor Interface	**Mandatory**	**Optional**	**Remarks**
GPS	X		Or higher accuracy, 1 Hz or higher update rate
GLONASS		X	
Gyroscope		X	
Accelerometer		X	
Real time clock		X	

Table 3.4: Sensor interfaces for BOP

3.4 Conclusion

To summerize the decisions of this chapter the definition of the basic Open platform is established.

The core elements of the hardware base for an ITSSv6 station BOP should have a high volume of reserved computing potential as future software elements may require further resources. For this reason a fairly powerfull setup was advised both for the Intel-based and for the embedded platform with clock rates over 400/800 MHz with relatively high RAM compared to other optimised HW platforms serving similar purpose.

Communication interfaces :

The platform should be equipped with multiple wireless communication systems. Two mayor technologies were chosen based on their properties and incidence: mobile telecommunication (3G/4G) and Wireless LAN (802.11 including both classic WiFi and 802.11p). The ITS architecture defines at least two types of link between different ITS entities. A reliable and quick interface is needed between and across vehicle and roadside equipment. 802.11p is recommended and prioritazed for this purpose. The distance of this connectivity is relatively short compared to the second type: One link must also be present between the central station and the previously mentioned nodes. 3G/4G or ethernet is presfered for this purpose.

Considering 3G and WiFi the technology, there should be little restriction as the differences between vendors and types does not affect our targets. On the other hand, the prematurity of the 802.11p technology and its primary role in equipment acceptance by the FOTs makes this a critical issue. The differences and properties of mayor radios can be found in a later chapter.

Sensors interfaces :

As the platform is primary meant to serve as a router, only the GPS sensor was labelled mandatory. Depending on the usage of the target platform, further intarfaces and sensors may become required.

Software base :

Software and Operating system recommendation was based on previous projects (CVIS, GeoNet). For deetails on the software related decisions please refer to project deliverable D3.1: "Intermediate Implementation Guideline" chapter "Core Software".

CHAPTER 4

Targeted Projects and Platforms

The porting of the software to specific target platforms and the integration of necessary communication interfaces (802.11p radios) is an essential part of the ITSSv6 project. The following chapter introduces all important embedded hardware systems encountered during development. A comparison table about the investigated equipment can be found in Appendix B for pointing out the specific differences between these embedded systems.

4.1 Specific Target Platforms

4.1.1 Laguna from Commsignia

Laguna is a multipurpose all-in-one (host+router+modem) solution consisting of LGN-00-11 base unit, 2x 802.11p radios.

- Cavium CNS3420 600MHz Dual Core ARM11 (∼1400 DMIPS)
- 128 MB DDR2-400 SDRAM
- 16 MB Flash
- 4 MB SPI-Flash
- Integrated GPS receiver
- Built-in vehicle power-supply with smart sleep mode
- Optional 3G support
- 2x 802.11p (ETSI G5) radios with driver included. (UNEX or ITRI mini-PCI).
- Open SDK
- Recommended for both roadside and vehicle use

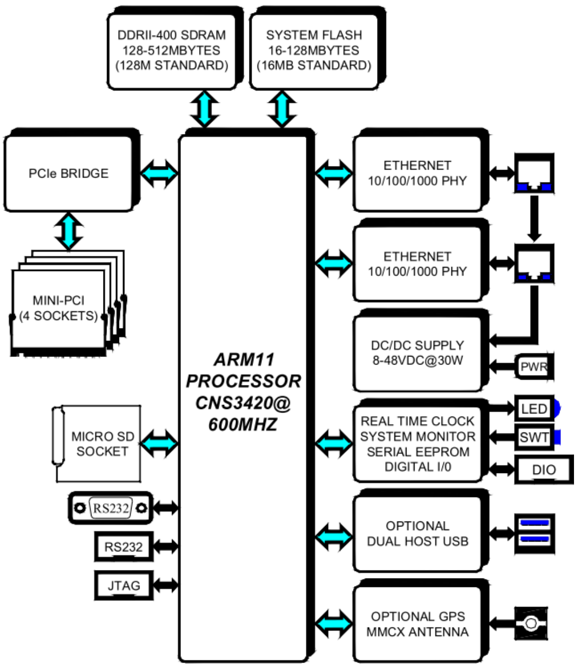

Figure 4.1: Functional overview of the Laguna platform

4.1.2 CM-i.MX53 from Bluetechnix

CM-i.MX53 is Bluetechnix's chosen internal STP to serve as their host system. Its key features are:

- Freescale Application Processor i.MX53, 800 MHz ARM Cortex-A8 (>∼2000 DMIPS)
- 1 GB DDR2-SDRAM
- 2 GB NAND-Flash
- 4 MB SPI-Flash

Figure 4.2 shows the platforms functional description. Several extender Boards exists for this platform, especially sensor interfaces like Gyro and acceleration sensor as well as GPS. As porting the ITSSv6 software to CM-i.MX53 is still ongoing there are no performance test result available to date.

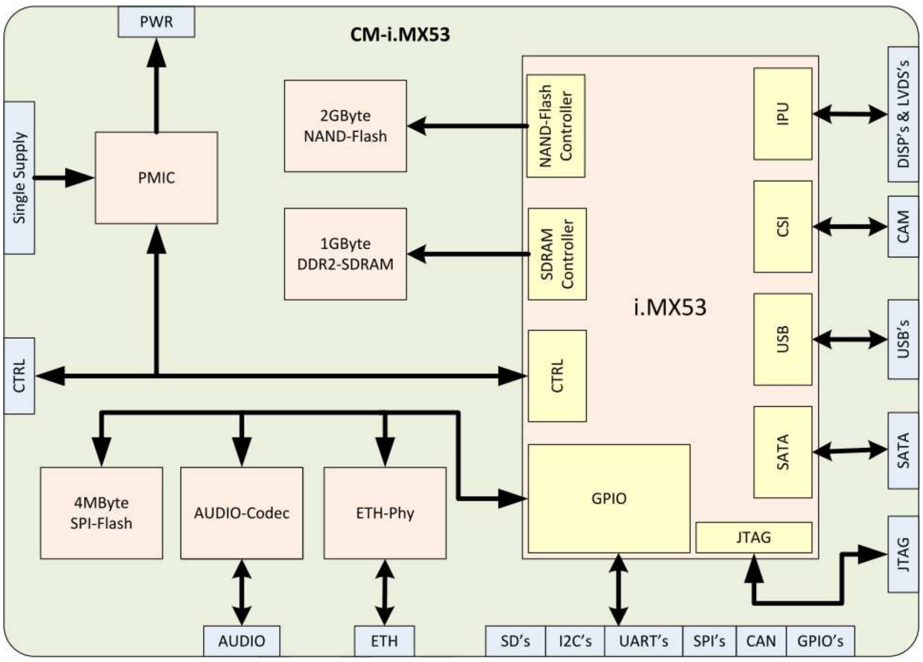

Figure 4.2: Functional overview of CM-i.MX53

4.1.3 Wi2U from lesswire

- Atmel ARM9 400MHz (∼400 MIPS)
- 64 MB DDR2-SDRAM
- 256 MB NAND-Flash

Figure 4.3 shows the platforms functional description. As porting the ITSSv6 software to the Wi2U is still ongoing there are no performance test result available to date.

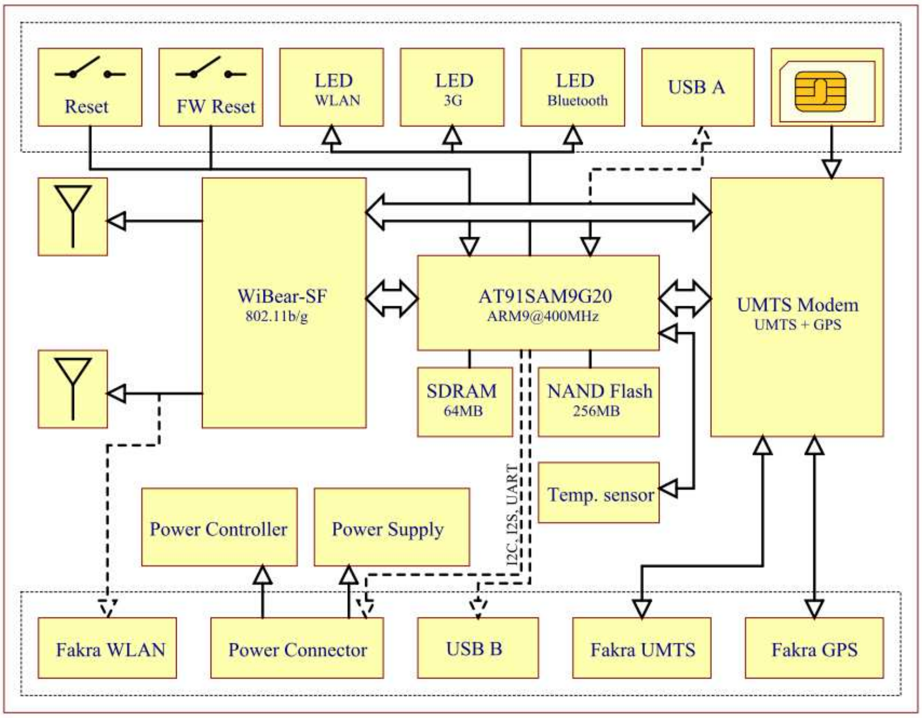

Figure 4.3: Functional overview of the Wi2U

4.1.4 VTC 6200 from NEXCOM

- Intel Atom D510 Dual Core 1.6GHz processor
- up to 2 GB DDR2-SDRAM
- SATA 2.5" HDD
- Equipped with 3G, 11p and 11n

4.1.5 Denso WSU

Denso is one of the main contributors to European FOTs and therefore is one of the platforms in focus.

- 400MHz MPC5200B PowerPC
- 128MB DDR SDRAM
- 64MB Flash ROM
- Equipped with 11p and external GPS

4.1.6 ITRI IWCU 2.0/4.0

- MPC8377 667 MHz PowerPC
- 256MB SDRAM
- 64MB Flash ROM
- Equipped with 11p and GPS

4.1.7 Cohda Wireless MK2,MK3 (full)

Cohda is one of the main contributors to European FOTs and therefore is one of the platforms in focus. Investigation targeted two versions of the hardware, the MK2 and the newest MK3. It should be noted that a simTD version also exists, please find it in among the transceivers in section 4.2.1.

A detailed description about the hardware can be found in Appendix A.

4.2 802.11p transceivers

The ITSSv6 project aims to extend its experience on car to car wireless communication. Interoperability and reliable throughput are essential prerequisites for being invited to European FOTs. Our commitment to apply the ETSI G5 standard in our environment results in a narrow selection of available devices.

4.2.1 Cohda Wireless MK2/MK3 - simTD version

The simTD edition of the MK2/MK3 is stripped-down version of the original devices. It does not include the host processor, but consists of a single or dual channel CohdaMobility IEEE 802.11p radio and a high speed USB 2.0 interface. UDP over USB is also present in some variants of the product. (On MK3 No GPS, ethernet or serial is present)

4.2.2 Marvell

Marvell Semiconductors shows interest in 11p technology, but will wait until the standardization process comes to an end.

4.2.3 Industrial Technology Research Institute (ITRI)

ITRI's Mini-PCI 802.11p Network Interface Card stands out from the market with its property of being Industrial-grade specifically designed for vehicular environments. It supports the frequency band from 5.86 GHz to 5.92 GHz. The ultra-wide range of the operating temperature -40°C∼+85°C allows it to work well in harsh vehicular environment. The ITRI Mini- PCI 802.11p NIC fully complies to the driver framework of the Linux kernel 2.6.32.45 or later, allowing users to control/use it using standard Linux APIs and quickly integrate it into their target systems. Optional software features, such as IEEE 1609, SAE J2735 Message Codec, ETSI TS 102 636-4 (GeoNetworking Part 4), ETSI TS 102 636-5 (GeoNetworking Part 5), ETSI TS 102 637-2 (CAM), ETST TS 102 637-3 (DENM), are available. From -97.3 dBm (with 3Mbps, 10MHz band, and 25C) to (with 27Mbps, 10 MHz band, and 25C)

4.2.4 Atheros AR5414

Unex DCMA-86P2 is a mini-PCI radio module for IEEE 802.11p based DSRC (WAVE) and ETSI G5 communication. The module strictly follows the ASTM E2213 specification for information exchange for various V2X applications.

Chip:	AR5414A-B2B (industrial grade)
Bandwidth control:	40/20/10/5 MHz
Operation temperature:	-40 ˜+85C
Power level to pass Class C spectrum mask:	20 dBm
ESD/Surge cap:	Up to 14KV, or higher
Minimum controllable power:	0.0˜2.5dBm
Sensitivity:	From -97.3 dBm (with 3Mbps, 10MHz band, and 25C) to -78.8 dBm (with 27Mbps, 10 MHz band, and 25C)
Modulation:	OFDM with BPSK, QPSK, 16-QAM, and 64-QAM
Freq. offset over OP. temp. & board to board variation:	within +/- 10ppm
OSC source:	40MHz TCXO
RX BPFs pass band:	5.855˜.925GHz
5.9 Ghz band support in HW:	Yes
On board Power Amplifier:	2pcs LX5530
Interface:	32-bit mini-PCI type III-A @ 3.3 VDC

Software driver

Controllable parameters:

- Bandwidth
- Transmit power
- Center frequency/Operating channel
- Priority of the transmission (QoS)
- ACK policy
- Bitrate

Status parameters:

- Center frequency
- Operating channel
- Transmit power
- Rx/Tx statistics (data, errors, etc.)
- RSSI (provided by driver API)
- Sensitivity (provided by driver API)

4.2.5 Other vendors - Please populate any item from the following list if input is available - All

- Broadcom, Ralink
- Siemens Wien (maybe IPTE?)
- Hitachi (maybe IPTE?)
- Renesas
- NEC
- Q-Free (maybe IPTE?)
- Kapsch (maybe IPTE?)
- Comsis (maybe IT?)

4.3 Targeted projects

4.3.1 FOTsis

FOTsis (European Field Operational Test on Safe, Intelligent and Sustainable Road Operation) is a large-scale field testing of the road infrastructure management systems needed for the operation of seven close-to-market cooperative I2V, V2I & I2I technologies (the FOTsis Services), in order to assess in detail both 1) their effectiveness and 2) their potential for a full-scale deployment in European roads. Specifically, FOTsis will test the road infrastructure's capability to incorporate the new cooperative systems technology at 9 Test Sites in four European Test-Communities (Spain, Portugal, Germany and Greece), providing the following services:

- Emergency Management
- Safety Incident Management
- Intelligent Congestion Control
- Dynamic Route Planning
- Special Vehicle Tracking
- Advanced Enforcement
- Infrastructure Safety Assessment

4.3.2 SCORE@F

SCORE@F is a French Field Operationnal Test for Road Cooperative System, in collaboration with DRIVE C2X and CO-DRIVE projects. It started in September 2010 for a duration of 30 months.

The purpose of the project is to significantly improve traffic flow and road safety by democratizing the use of communications technology infrastructure to vehicle and vehicle to vehicle. This will be achieved by validating the technological choices suited for cost-optimized service and also validating the economic feasibility of a deployment scenario.

Its expected results are:

- Quantification of the benefits in terms of Traffic and Safety
- Identification by all stakeholders of a viable deployment scenario
- Validation or evolution proposals of "Car to Infrastructure"ETSI standards and Applications
- Test and development of qualification tests to ensure interoperability
- Choice of technical architectures and detailed assessment of deployment costs associated with of a "package" of services

CHAPTER 5

Initial Software Analysis

An initial software analysis regarding the performance of the initial stack in real life situations and a static code analysis has been done. This chapter describes both of them. The results of the dynamic code analysis shows the performance of the ITSSv6 stack in terms of throughput. The static code analysis was done doing a static code inspection regarding portability and resource usage.

5.1 Static software analysis

The following section provides information about static code analysis regarding portability, resource usage and performance issues of the ITSSv6 stack in order to use it on different platforms even with small RAM memory and less performance. There are several points which will be taken into account which have an impact on performance of the stack on platforms with limited resources.

Not only hardware platform issues will be taken into account but also regarding the portability to other operating systems. The code inspection was done manually. At that time no automated code inspection tool has been used.

5.1.1 General hardware dependencies

The entire source code of the ITSSv6 stack doesn't have any hardware dependent code and is ready to be ported to many different platforms. All hardware dependent code is limited to the lower level driver implementations for the physical interfaces.

5.1.2 Floating point operations

Floating point operations may result in a decreasing performance on platforms with CPUs without floating point units. In that case the floating point operations will be emulated by implementations in software floating point libraries.

In the ITTSv6 stack there is only a small usage of floating point operations mainly for statistical calculations therefore they should not have an impact on platforms without floating point units

5.1.3 Endianness

Platforms with little and big endianness can be found on the market. Therefore care has been taken to eliminate dependency on the endianness wherever it was possible.

Mainly elementary types have been used for variables. They will be translated according to the endianness of the CPU by the compiler of the platform.

For accessing single bytes within variables declared using integrated types, special translation macros (WORDS_BIGENDIAN) have been used to eliminate the dependency of machine endianess.

5.1.4 Memory operations

Extensive usage of memory copying functions like 'memcpy' functions should be avoided if it is possible. Accessing the external memory buffers can several decrease the performance of a code. Currently no avoidable 'memcpy' function calls have been found. This point will be under close investigation during the whole code development.

5.1.5 Coding style

The source code has been investigated according to the /Documentation/CodingStyle document for the linux kernel.

5.1.6 MISRA

MISRA compatibility code generation is not a basic requirement and has therefore not been checked yet.

5.1.7 Global variables

If large arrays are declared as global variables they may lead to a greater memory footprint of the code, especially if they are not zero initialized. Therefore declarations of such large amount of memory in form of global array declarations have to be avoided where it is possible. They should be allocated on heap. Of course this is not the case if lookup tables have to be implemented.

Global variables are used within the code but there are no critical declarations of large arrays.

Usage of static keyword :

> Global variables should be declared as static to generate better assembler code on embedded platform RISC processors or DSP. In that case surrounding synchronization code can be avoided.
>
> In the stack implementation global variables are always declared as static.

5.1.8 Conditional Branches

Within loops : Conditional jumps leads to empty pipelines on CPUs without code prediction. Therefore they should be avoided. The code has been analyzed regarding code constructs like the following. Be aware that this is only pseudo code:

```
x=5;
```

```
for (...) {
        if (x == 5)
         ...
}
```

No such avoidable conditional jumps have been found in the code.

Probability based conditional statements :

On conditional branches either using the if statement or a switch case construct, the most probably expression should be the first. This because every "if" statement creates a conditional jump which on processors without code prediction empties the pipeline which leads to lower execution performance.

There are some of such expression present which could be optimized regarding probability based conditional execution. This issue will taken into account later on in the project. At the current state of the development there are to few information about the execution probability of different branches in real life conditions.

switch vs if statement :

In some cases it may be of advantage to gate the switch case statement with a preselecting 'if' statement avoiding the need to pass all the conditions before running into the 'default' statement. The reason is the same as mentioned above. In that case you avoid additional conditional branches which could decrease the performance of the code. The following example shows such an optimized piece of code:

```
if (x < 4) {
        switch (x) : {
            x = 1:
            break;
            x = 2:
            break;
            x = 3:
            break;
            default:
        }
}
```

Boolean expression vs nested if :

Instead of using nested if statements with one boolean expression each, a combination of boolean expressions should be used. This because each of the if statements creates a conditional branch with the already mentioned flushing of the pipeline which leads to a possible decrease of the performance. The boolean expression instead can be solved by using processor register based boolean operations. After solving the boolean operation just one conditional branch is needed. The following example shows the difference between a nested if statement and the appropriate single statement including one boolean expression.

```
if (x == 2) {
        if (y == 4) {
        ...
}
```

From a performance perspective it would be better to code it into:

```
if (x == 2 && y == 4) {
        ...
}
```

The ITSSv6 stack is allmost free of such performance limiting conditional branch constructs.

5.1.9 Compiler optimization

Compiler optimization can severely increase performance of the software but it is highly platform dependent. On some CPUs especially DSP based processors the special features of the DSP architecture will only be used if compiler optimization has been enabled. The compiler often uses special precompiler switches to control the optimization process.

In the software of the ITSSv6 stack currently no special precompiler switches for optimization have been used. Therefore special advantages on certain architectures can't be used. In that way the platform independency can be granted. On special hardware architectures it may be an advantage regarding the performance, if the user of the stack adds special precompiler commands for compiler optimization but this should be completely under the responsibility of the user.

5.1.10 Inline declaration

Inline declaration where often and consequently used within the code.

5.1.11 Function calls

Recursive calls :
On platforms with small memory size, recursive calls could lead to a memory problem because of the amount of needed stack size. There are some recursive calls within the code but no critical issues will be expected at that time.

Parameters :
The number of parameters within a function call could have an impact on the needed stack size. Usually up to three parameters will be handled by using processor registers. Additional parameters will be passed using the stack. Therefore passing big size arrays to a function is not recommended for using it on platforms with a small amount of available memory.

No critical sections have been found regarding this issue.

5.1.12 Stack usage within functions

Variable declared within functions will be allocated on the stack. Therefore for small memory equipped platforms it is necessary to keep the amount of needed stack memory for the variable small. Therefore from a memory requirement perspective it is not recommended to declare big sized arrays within functions.

The source code of the ITSSv6 stack doesn't include critical declarations of such type. The size of the declared arrays within functions is almost below 250 bytes.

5.1.13 Data Alignment

Data alignment can be an issue on small performance RISC processors. They have a dedicated interface to the volatile memory device. To get the highest performance it is important to use the maximum data size of the interface. On modern low performance CPUs this is at least 16 bytes. On medium to high performance embedded processors even 32 bytes. Therefore it is important to use larger variable types to get the best performance of the code regarding execution time. Therefore it is suggested to use the 'int' type for common variables. From almost all the compilers the int type will be translated to the maximum width of the memory interface, in terms of number of bits. This point is also important for loop control variables. The C standard states that loop control variables which are not declared as integer should be extended to an integer. This creates additional computation effort.

The code inspection regarding this topic shows a consequent usage of the 'int' type. Almost all counting variables and return values are declared as 'int'. This issue doesn't affect pointer declarations which will always be adapted by the compiler to the address scheme of the used processor. Therefore they haven't been taken into account.

5.1.14 Operating system dependencies

This section discusses the dependency on operating system resources. The complete stack is tightly interwoven with the linux kernel. Several linux based modules like D-BUS will be used by the stack. Interoperability with other operating systems therefore may be an issue for future projects but is not on the focus of the current project. Nevertheless for kernel calls regarding thread and mutex operations the posix interface has been used. This guarantees interoperability with several operating systems as posix is a common and well documented standard interface to access kernel resources.

5.2 Validation of the Executing Software

This section describes the operation of the ITSSv6 communication stack while running a basic set of tests and analyzing results from the network performance point of view, which is the most interesting for the project. A more detailed experimental assessment of the software platform can be found in ITSSv6 deliverable D4.1 [ITSSv6-D4.1].

5.2.1 Testing scenario

The scenario considered comprises a reference deployment for testing basic network mobility, where nodes have been distributed at UMU laboratories. The overall picture of the scenario is showed in Figure 5.1.

As can be seen, the Home Central ITS Station is placed indoors, in one of the laboratories of UMU. Two network domains are considered, the one provided by the Home Central ITS-S (home domain), through a 6o4 tunnel over 3G, and a visited domain provided by a Visited Roadside ITS-S. Since these tests have been performed indoors, the Roadside ITS-S has been located in a laboratory where the Vehicle ITS-S is also placed. The 802.11b support has been physically provided by a common 802.11g access point. A physical embedded computer

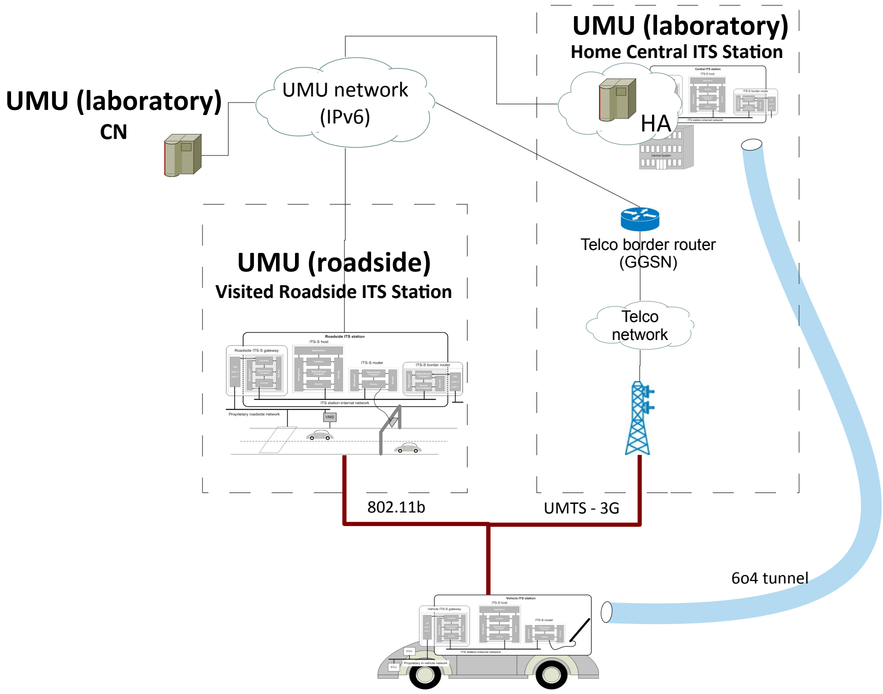

Figure 5.1: Testing scenario for dynamic evaluation of the distribution

installed together with the access point integrates the required router capabilities of the Roadside ITS Station. The Vehicle ITS-S is maintained in the laboratory, where the 3G coverage is good, and network changes are induced by disabling the 802.11g access point.

Finally, a correspondent node (CN), i.e. a common networked host, has been placed in another laboratory to provide an edge for evaluating data connections under mobility. An embedded computer which provides ITS-S router capabilities (mobile router) has been used to execute the ITSSv6 software modules, while a common laptop with Linux is used as Vehicle ITSS-S Host. More details about the hardware equipments used can be found in ITSSv6 deliverable D4.1 [ITSSv6-D4.1].

Software tools used to assess the performance of the network have been Iperf, to create TCP and UDP data flows, and the Linux command Ping6, to generate ICMPv6 messages.

5.2.2 Performance analysis

The previous scenario has been used to analyze the network performance under handoff conditions. Figure 5.2 shows the results obtained when a TCP flow is maintaining between the Vehicle ITS-S Host and the CN in the above described scenario. First of all, it can be observed the variable results, due to, on one hand, the inherent wireless environment, and, on the other hand, the behavior of the TCP slow start algorithm for adapting to the medium. Moreover, it is noticeable that 802.11b (WiFi) results are quite better than UMTS (3G) values, due to the better bandwidth achievable in 802.11 network connections when a near access point is available. Nevertheless, what is important here is to observe how the communication stack correctly performs network changes at both physical/MAC and network layers without loosing connectivity, maintaining data rates of around 3 Mbps when 802.11g is used and 1 Mbps when UMTS is selected.

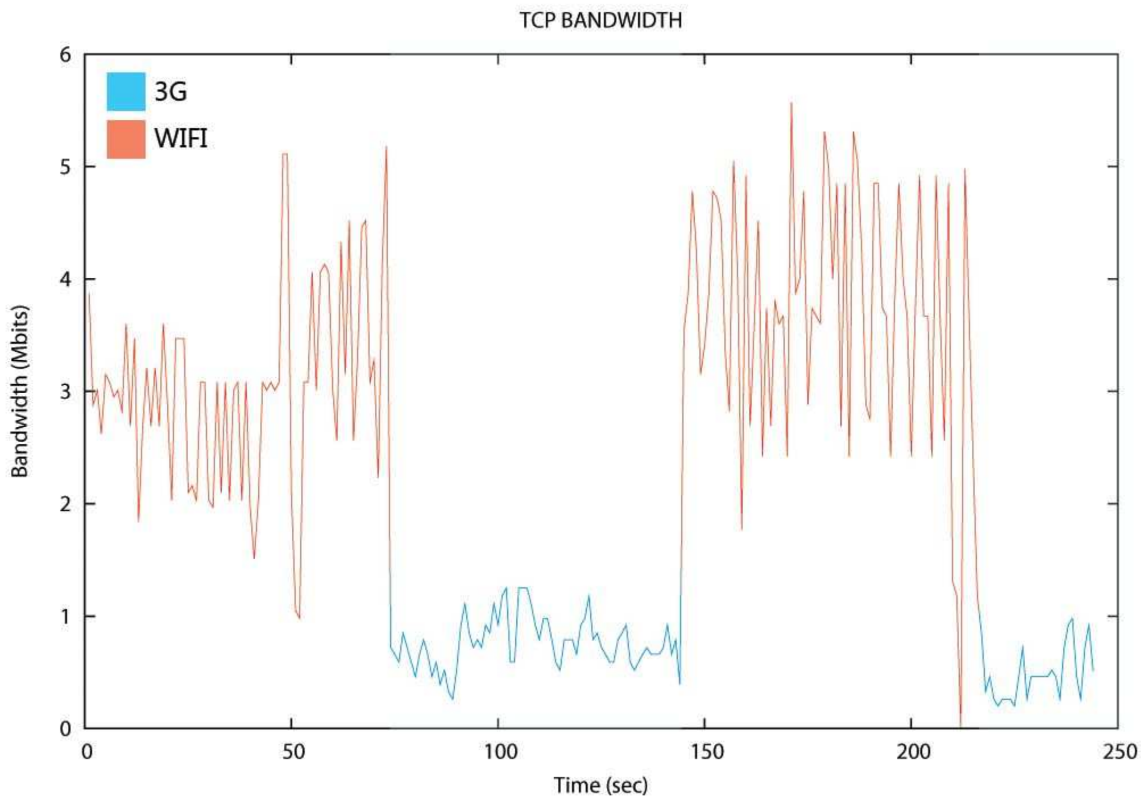

Figure 5.2: Bandwidth of the communication stack under handoff and TCP traffic

Figure 5.3 and Figure 5.4 shows the performance of the system when several handoffs occur and a variable UDP flow is sent from the CN to the Vehicle ITS-S Host. From time 0 sec to time 200 sec a UDP flow of 10 Kbps is sent. As can be seen in the graphs, the bandwidth is almost ideal and only one packet is lost when the 802.11g link is used. In the first handoff from UMTS to 802.11g, no packets are lost, while in the second handoff one packet is lost. When the data flow is changed for 100 Kbps, the UMTS networks operates quite well, neither loosing packets nor varying the required bandwidth. However, when the handoff to 802.11g occurs, a uncommon jitter is evident between times 235 and 345 sec. We explain these results by the shared usage of the access point with other users. The UMTS network is able to maintain a constant bit rate due to the tests were carried out at the end of the day in the non populated area of the Campus of Espinardo (University of Murcia). Finally, when the UDP data rate is modified up to 5 Mbps at time 400 sec, the network presents great loss rates, but it is noticeable how the maximum allowable bandwidth is reached for the UMTS case (around 2.5 Mbps), with a constant loss rate, while the 802.11g values show that during a period of ten seconds no packets are lost. This last fact shows that a higher bandwidth and a higher packet delivery ratio would be achievable by the 802.11g link. Finally, it is noticeable again how the network handoffs are correctly supported my the NEMO module of the communication stack.

A last analysis of results is illustrated in Figure 5.5, which shows the round-trip delay time (RTT) obtained sending ICMP traffic during the first TCP test. In this way, it is possible to

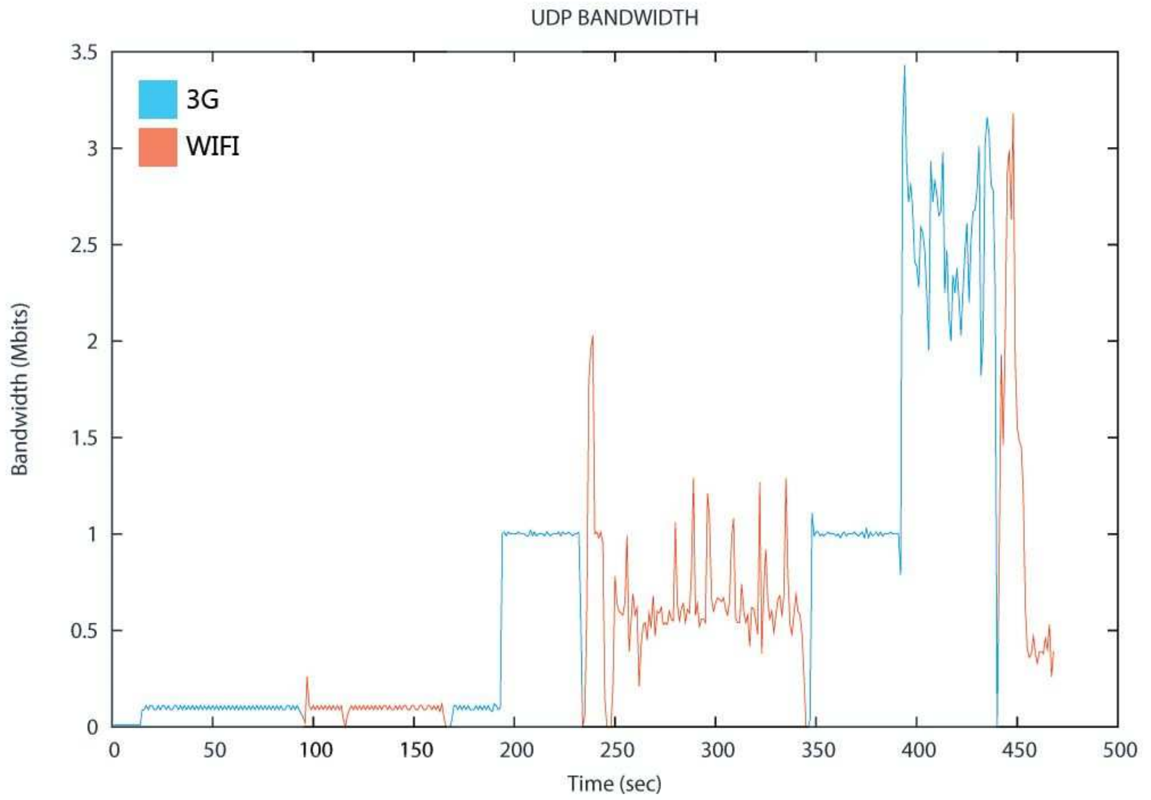

Figure 5.3: Bandwidth of the communication stack under handoff and variable UDP traffic

investigate the packet delay under stressing situations. As can be noted in the graph, high variabilities in delay have been obtained. When the TCP protocol decides to increase the transmission window, the RTT decreases. This is evident for the case of 802.11g at time 50s, where the TCP transmission decreases an a quite small delay is observed (around 10 ms). At time 105 sec there is an equivalent example for the UMTS case. It is also important to note that the minimum RTT delay is obtained when the 802.11g technology is used (around 10 ms), which is smaller that the lower values obtained in the UMTS case, which are about 50 ms, as it was expected.

5.3 Future analysis

As the initial software test was done on just a basic set of features, a more complex and widespread analysis is still to be made in the future. The final software stack will run on target devices, a full run-time code analysis is needed on an embedde BOP in order to validate the platforms computing power and to map its limits. This will enable FOTs and other developers to be able to calculate needed resources when deciding for a target platform.

Basic performance test include :

- CPU usage
- memory requirements

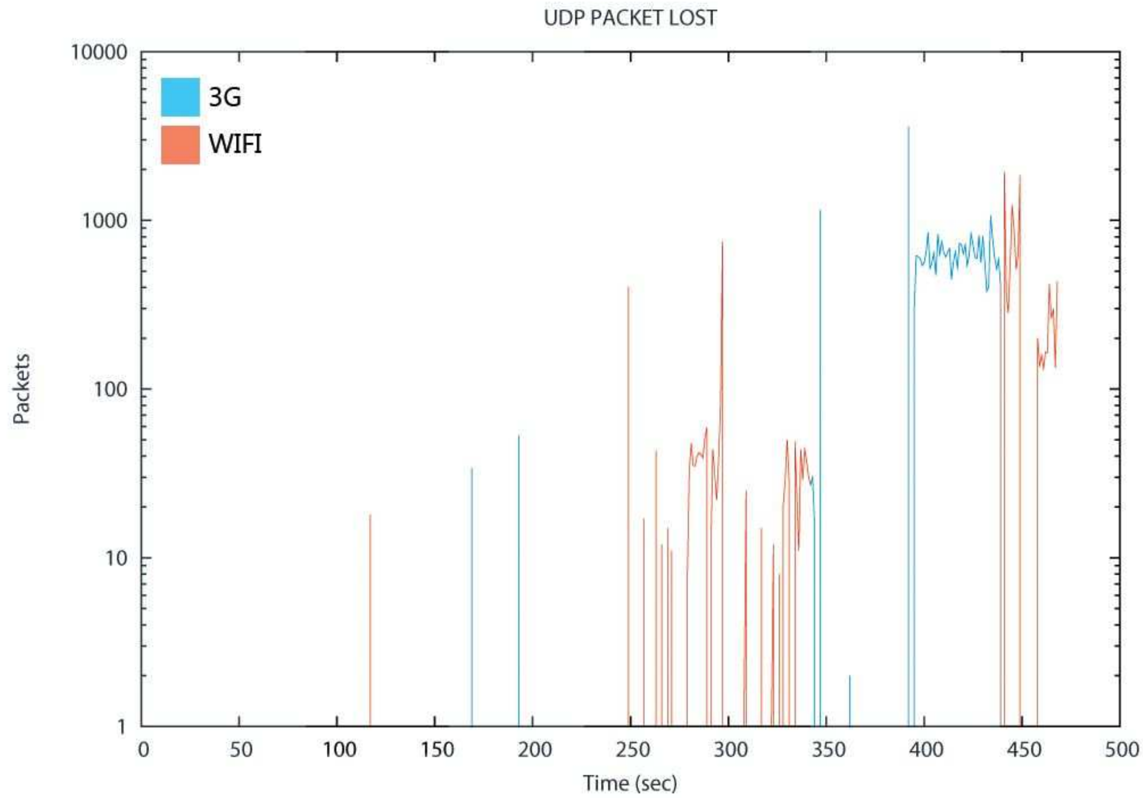

Figure 5.4: Packet delivery ratio of the communication stack under handoff and variable UDP traffic

Extended performance test include :

- CPU usage with individual packages on/off
- memory requirements with individual packages on/off
- periferial performance
- hard disk usage

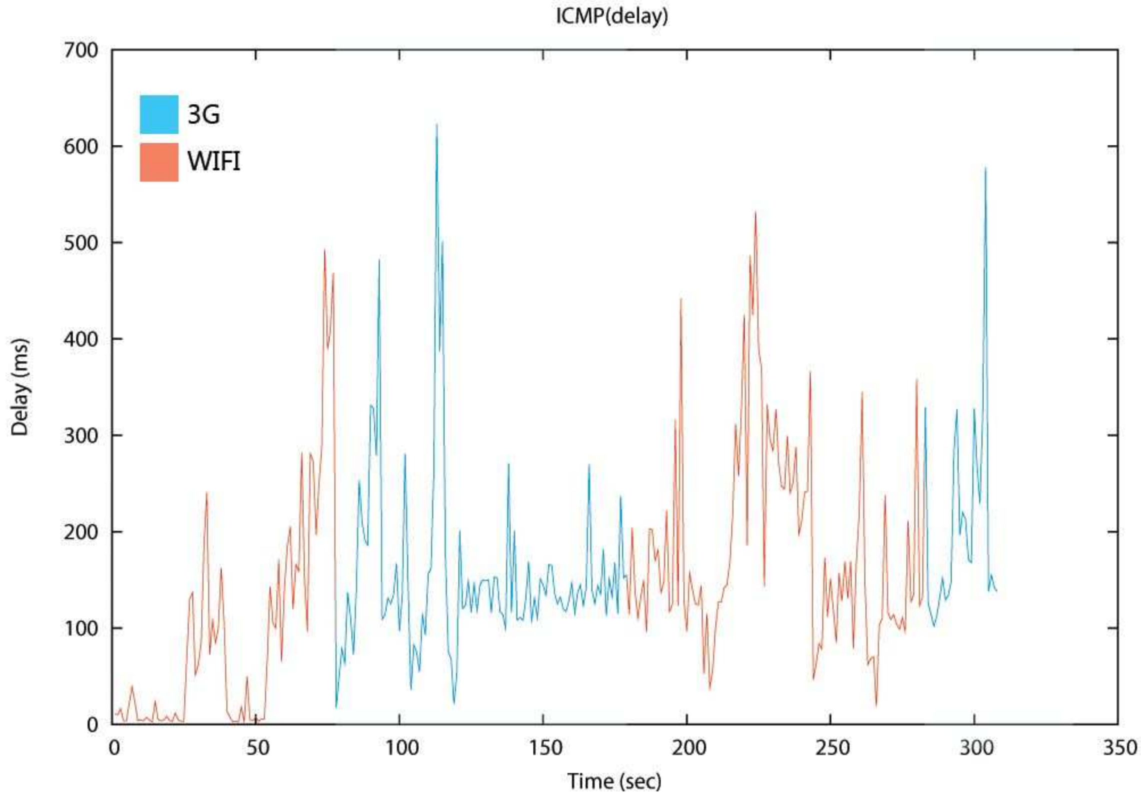

Figure 5.5: Round trip delay time obtained under handoff and TCP traffic

CHAPTER 6

Porting Guideline

6.1 Porting the ITSSv6 Linux a Specific Target Platforms

The development environment introduced in D3.1 is able to produce binaries to multiple target platforms, therefore it is the best candidate for porting system. As written in the Development Environment chapter in the D3.1 document, once software components are in place, the target platform has to be specified with the following parameters

- Target Architecture toolchain: A toolchain has to be added for the given target platform which is able to cross-compile binaries.
- Target Kernel: The operating system kernel is the most important step in the porting process. The kernel configuration has to be adjusted to match hardware requirements of the target embedded platform. Missing drivers need to be implemented. Compression is to be used to reduce final size.
- Initialization process, flash layout: As the operating system needs different initialization methods when used in embedded platform, some configuration is needed to enhance boot time and. Special filesystems has to be added to optimize the system for flash memory storage devices.

Once the above configuration and implementation steps are in place, the development environment is able to produce firmware images that can be run on the target platform. The ideal porting process is when the same version of development tools, kernel version, etc. can be used for each target platform. This step heavily depends on the SDK and documentation provided for the platform. Some closed source solutions might not guarantee full portability, and in such cases custom porting strategies are needed.

Currently only the building of pre configured binaries for specific platforms (x86 and Laguna) are supported. Additional options, supported targets and a general guidence for porting the core components will be added to this porting guide.

APPENDIX A

Cohda Wireless MK2,MK3 (full) in detail

Unlike most of the 11p radio vendors, Cohda decided not to base their design on one of the known IC vendors (like Atheros), but has the 802.11p PHY/MAC implemented in an FPGA (MK2), the MK3 on the other hand has the 802.11p PHY/MAC baseband processing performed in an automotive grade software defined radio chip from NXP. The two types are otherwise functionally and mechanically identical.

System components of the MK3

- ARM11 533MHz
- 64MB LPDDR-133 SDRAM (up to 256MB)
- 512MB NAND-Flash ???
- Single or Dual channel CohdaMobility IEEE 802.11p radio
- IEEE 1609 networking system, including security functions.
- Advanced GPS positioning system
- on-board, dead-reckoning sensors (gyroscope)
- 2 CAN bus interfaces - high speed and low speed
- Ethernet (IPv4/IPv6 networking), USB 2.0 high-speed, serial
- External VGA touch-screen, audio in/out.

APPENDIX B

Comparison of embedded platformsl

B.1 Comparison of embedded platforms

Table B.1 gives an overview of the main platforms.

Parameter	Laguna	Cohda MKx full	CM-i.MX53	Wi2U
Basic parameters				
CPU Vendor	Cavium		Freescale	Atmel
CPU Core	v6 Dual ARM11	v6 ARM11	v7 ARM C.-A8	v5 ARM9
CPU Freq	2 * 600 MHz	533MHz	1GHz	400MHz
MIPS	$\sim$1400DMIPS		$>$2000DMIPS	
SDRAM	128 MB DDR2-400 (max512MB)	64MB LPDDR-133 (max256MB)	1GB DDR2-400	64MB
NAND Flash	16MB (max 128)	512MB	2GB	256MB
ext.Flash	4MB SPI	no (via microSD)	4MB SPI	no (via USB)
SDK	open			partner
Available interfaces				
11p Core	ITRI or Atheros AR5411 via Mini-PCI	MK1:FPGA /MK2:NXP MARS SoC	EXT?	EXT
11p "Type"	G5/M5/WAVE	WAVE		
GPS	A-GPS	A-GPS	A-GPS	A-GPS
USB	2 * USB 2.0	USB 1.2 and 2.0 High-speed	2 * USB2.0 High-speed	2 * USB2.0
ETH	2 * 10/100/1000 Base-Tx	10/100 Base-Tx	10/100 Base-Tx	10/100 Base-Tx
CAN	NO (via USB)	2 (high&low speed)	yes	NO (via USB)
3G	EXT	NO (via USB)	NO (via USB)	yes
Temp	-40°C - +85°C	-40°C - +85°C	-40°C - +85°C	-40°C - +85°C
Dimensions (mm)	100 x 152 x 30	100 x 170 x 40		

Table B.1: Target Platforms comparison table

APPENDIX C

List of acronyms

2G 2nd Generation mobile telecommunications
3G 3rd Generation mobile telecommunications
A-GPS Assisted GPS
AnaVANET ANAlyzer for Vehicular Adhoc NETworks
AP Access Point
AR access router
ASN.1 Abstract Syntax Notation One
AU Application Unit
BA Binding Acknowledgement
BC Binding Cache
BE Binding Error
BID Binding Identification number
BOP Basic Open Platform
BU Binding Update
BUL Binding Update List
BR border router
BT Bluetechnix Mechatronische Systeme GmbH
C2C-CC Car-to-Car Communication Consortium
C2CNet Car-to-Car Network
CALM Communications Access for Land Mobiles
CAM Co-operative Awareness Messages
CCU Communication and Control Unit
CDMA Code Division Multiple Access
CE Correspondent Entity
CEN European Committee for Standardization
CI Communication Interface
C-ITS Cooperative Intelligent Transportation Systems
C-ITSS central ITS station
CIMAE Communication Interface Management Adaptation Entity
CN Correspondent Node
CoA Care-of Address
CoDrive Co-Pilote pour une Route Intelligente et des Véhicules Communicants
Coopers Co-operative Systems for Intelligent Road Safety
CoT Care-of Test
CoTI Care-of Test Init
CR central router
CVIS Cooperative Vehicle-Infrastructure Systems
DAD Duplicated Address Detection
DENM Decentralized Environmental Notification Messages
DHAAD Dynamic Home Agent Address Discovery
DHCP Dynamic Host Configuration Protocol
DMIPS Dhrystone MIPS, Million instructions per second
DNS Domain Name System
DoT U.S. Department of Transportation
DriveC2X Connecting vehicles for safe, comfortable and green driving on European roads
DSRC Dedicated Short Range Communications

EC European Commission
ETSI European Telecommunications Standards Institute
FlowID Flow Identifier
FM Frequency Modulation
FMIPv6 Fast Handovers for Mobile IPv6
FOT Field Operational Test
FOTsis Field Operational Test on Safe, Intelligent and Sustainable Road Operation
FP6 Sixth Framework Programme
FP7 Seventh Framework Programme
GLONASS Global Navigation Satellite System
GeoNet IPv6 GeoNetworking
GPRS General Packet Radio Service
GPS Global Positioning System
GPSR Greedy Perimeter Stateless Routing
GSM Global System for Mobile communications
HA home agent
HIP Host Identity Protocol
HMIPv6 Hierarchical Mobile IPv6
HNA Host and Network Association
HoA Home Address
HoT Home Test
HoTI Home Test Init
HSPA High Speed Packet Access
I2V Infrastructure-to-Vehicle
ICMPv6 Internet Control Message Protocol version 6
ICT Information Communication Technologies
IEEE Institute of Electrical and Electronics Engineers
IETF Internet Engineering Task Force
IME Interface Management Entity
INP Internal Network Prefix
INPA Internal Network Prefix Advertisement
Inria Institut National de Recherche en Informatique et en Automatique
INPD IPv6 Internal Network Prefix Discovery
IINP ITS Station Internal Network Prefix
IP Internet Protocol
IPFR IP Filter Rule
IPsec Internet Protocol security
IPTE Schalk & Shalk OG
IPv6 Internet Protocol version 6
ISO International Organization for Standardization
ITS Intelligent Transportation Systems
ITSSP ITS Station Protocol
ITSSPD ITS Station Protocol Daemon
ITS-S ITS station
ITSSv6 web page http://www.itssv6.eu
IT Institut Mines Telecom
ITU International Telecommunication Union
L2 Layer 2
L2TP Layer-2 Tunneling Protocol
L3 Layer 3
LAN Local Area Network
LDM Local Dynamic Map
LLC Logical Link Control
LTE Long Term Evolution
LS Location Service
LT Location Table
LW lesswire
MAC Medium Access Control
MADM Multiple Attribute Decision Making
MAN Metropolitan Area Network
MANET Mobile Ad-hoc Network
MAP Mobility Anchor Point
MCoA Multiple Care-of Addresses Registration
MIB Management Information Base
MIPS Million instructions per second
MLME Medium Access Control (MAC) Layer Management Entity
MN Mobile Node
MNN mobile network node
MNP Mobile Network Prefix
MNPP Mobile Network Prefix Provisioning

MobiSeND Mobile Secure Neighbor Discovery
MR mobile router
MTU Maximum Transmission Unit
NA Neighbor Advertisement
NAT Network Address Translation
NDP Neighbor Discovery Protocol
NEMO Network Mobility
NemoBS Network Mobility Basic Support
NEPL Network Mobility (NEMO) Platform for Linux
NMEA National Marine Electronics Association
NS Neighbor Solicitation
NTP Network Time Protocol
OASIS Operation of Safe, Intelligent and Sustainable Highways
OBU On-Board Unit
OLSR Optimized Link State Routing
OSI Open Systems Interconnection
OSPF Open Shortest Path First
PAN Personal Area Network
PathID Path Identifier
PDR Packet Delivery Ratio
PFBU Peer Flow Binding Update
PHY Physical
P-ITSS personal ITS station
PM Person-Month
PLME PHY Layer Management Entity
PMIPv6 Proxy Mobile IPv6
PPP Point-to-Point Protocol
PR personal router
PRESERVE Preparing Secure Vehicle-to-X Communication Systems
QoS Quality of Service
R2C Roadside ITS station to Central ITS station
RA Router Advertisement
RADVD Router Advertisement Daemon
RDS Radio Data System
RFC Request for Comments
RIPng Routing Information Protocol
R-ITSS roadside ITS station
RO Route Optimization
RPDB Routing Policy Database
RS Router Solicitation
RSSI Received Signal Strength Indication
RSU Road Side Unit
RR roadside router
RTT Round-Trip Time
SafeSpot Cooperative vehicles and road infrastructure for road safety
SAP Service Access Point
SAT ITS station access technologies layer
SAW Simple Additive Weighing
SCORE@F Système COopératif Routier Expérimental Français
SCTP Stream Control Transmission Protocol
SeVeCom Secure Vehicular Communication
SF ITS station facilities layer
SHIM6 Level 3 Multihoming Shim Protocol for IPv6
SHIM6 Site Multihoming by IPv6 Intermediation
SLAAC Stateless Address Auto-Configuration
SME ITS station management entity
SMIv2 Structure of Management Information Version 2
SNMP Simple Network Management Protocol
SNT ITS station networking & transport layer
SPI Security Parameter Index
SSE ITS station security entity
STP Specific Target Platform
SZTAKI Magyar tudomanyos akademia szamitastechnikai es automatizalasi kutato intezet
SZT SZTAKI
TC204 WG16 Technical Committee 204 Working Group 16
TCP Transmission Control Protocol
UDP User Datagram Protocol
UMU Universidad de Murcia
UMTS Universal Mobile Telecommunications System

UTC Coordinated Universal Time
V2C Vehicle ITS station to Central ITS station
V2I Vehicle-to-Infrastructure
V2L Vehicle ITS station to legacy system
V2P Vehicle ITS station to Personal ITS station
V2R Vehicle ITS station to Roadside ITS station
V2V Vehicle ITS station to Vehicle ITS station
VANET Vehicular Ad-hoc Network
V-ITSS vehicle ITS station
VR vehicle router
VCI Virtual Communication Interface
WAVE Wireless Access in Vehicular Environments
WG Working Group
WGS-84 World Geodetic System 84
WIMAX Worldwide Interoperability for Microwave Access
WLAN Wireless Local Area Network
WSMP Wireless Access in Vehicular Environments (WAVE) Short Message Protocol
XML Extensible Markup Language

Bibliography

[ITSSv6-D4.1] ITSSv6 Members. ITSSv6 STREP No.210519 D4.1 Initial Validation & Evaluation Results. *ITSSv6 Deliverable*, 2012. ITSSv6-D4.1-v1.0. pages 29, 30