International Journal of Advances in Telecommunications Electrotechnics, Signals and Systems

a publication of the International Science and Engineering Society

Vol. 4, No. 2
2015

ISSN: 1805-5443

www.ijates.org

I J A T E S²

International Journal of Advances in Telecommunications Electrotechnics, Signals and Systems

a publication of the International Science and Engineering Society

Vol. 4, No. 2, 2015 ISSN: 1805-5443

Aims and Scope

The International Journal of Advances in Telecommunications, Electronics, Signals and Systems (IJATES²) is an all-electronic international scientific journal with the aim to bring the most recent and unpublished research and development results in the area of electronics to the scientific and technical societies, and is supported by the ISES (International Science and Engineering Society, o.s.). The journal's scope covers all the aspects of telecommunication, signal processing, theory and design of circuits and systems for electronics.

The IJATES² is ready to publish experimental and theoretical full papers and letters submitted by prospective authors. Paper submitted for publication must be written in English and must follow a prescribed format. All papers are subjected to a critical peer-review prior to publication.

The IJATES² is an open access journal which means that all content is freely available without charge to the user or his/her institution. Users are allowed to read, download, copy, distribute, print, search, or link to the full texts of the articles in this journal without asking prior permission from the publisher or the author. This journal provides immediate open access to its content on the principle that making research freely available to the public supports a greater global exchange of knowledge.

www.ijates.org

Achieving Performance Speed-up in FPGA Based Bit-Parallel Multipliers using Embedded Primitive and Macro support

B. Khurshid and R. N. Mir

***Abstract*— Modern Field Programmable Gate Arrays (FPGA) are fast moving into the consumer market and their domain has expanded from prototype designing to low and medium volume productions. FPGAs are proving to be an attractive replacement for Application Specific Integrated Circuits (ASIC) primarily because of the low Non-recurring Engineering (NRE) costs associated with FPGA platforms. This has prompted FPGA vendors to improve the capacity and flexibility of the underlying primitive fabric and include specialized macro support and intellectual property (IP) cores in their offerings. However, most of the work related to FPGA implementations does not take full advantage of these offerings. This is primarily because designers rely mainly on the technology-independent optimization to enhance the performance of the system and completely neglect the speed-up that is achievable using these embedded primitives and macro support. In this paper, we consider the technology-dependent optimization of fixed-point bit-parallel multipliers by carrying out their implementations using embedded primitives and macro support that are inherent in modern day FPGAs. Our implementation targets three different FPGA families viz. Spartan-6, Virtex-4 and Virtex-5. The implementation results indicate that a considerable speed up in performance is achievable using these embedded FPGA resources.**

***Keywords*— Fixed point arithmetic, FPGA primitives, VHDL, Instantiation based coding, Look-up table.**

I. Introduction

The multiplier circuit is one of the fundamental components used in digital signal processing (DSP) [1] [2] [3] [4]. The field of DSP has always been driven by the advancements in scaled very-large-scale-integration (VLSI) technologies. The goal of digital design is to maximize the performance while keeping the cost down [5]. In the context of general digital design, performance is measured in terms of the amount of hardware circuitry and resources required; the speed of execution (throughput and clock rate); and the amount of power dissipated. There is always an application-driven tradeoff between these parameters. It is, therefore, desirable to have an efficient realization of these circuits for use in different DSP systems [6] [7].

DSP algorithms have traditionally been implemented using general purpose processors or DSP processors. However, with current trend moving back towards hardware intensive processing it becomes important for the designers to give a serious thought to the underlying implementation platform [8]. Applications demanding an increased performance mainly use application integrated circuits (ASIC) or structural ASICs [2]. The main attraction with ASICs is that the architecture can be developed specifically to meet the performance requirement. However, the non-recurring engineering (NRE) costs associated with ASICs have cornered their use only to high-volume markets. Field programmable gate arrays (FPGAs) provide an alternate approach to ASICs. They avoid the high NRE costs by giving the user the flexibility to configure the device in field [4], [9]. Some other advantages include large-scale integration [4], [10], lower energy requirements [11], [12] availability of several on-board intellectual property (IP) cores [13] etc.

Design for FPGAs differs dramatically from general VLSI design [14]. The design process proceeds through phases like design entry, synthesis, translation, mapping and place & route (PAR). Design entry is the only manual phase in the entire design flow. Therefore, using FPGAs as an implementation platform requires programming of the desired functionality using some hardware descriptive language (HDL), as it is the most widely used design entry method [15]. The rest of the design process is automated and there is a strong computer aided design (CAD) support for synthesis and implementation. However, sophisticated CAD tools are often not good enough to meet some design constraint if an arbitrary coding style is used [16]. A popular guideline that has been followed for writing functional synthesizable HDL codes is the RTL guideline, where RTL stands for register transfer level, signifying that data transfer should occur through registers only. These guidelines adhere to synchronous design practices and signify the regulation of data flow, and how data is being processed [17] rather than what part of the FPGA fabric processes the data. In effect, such codes are purely inferential and strongly rely on the software environment that distributes the logic as per the design goal. Thus, in order to effectively use embedded primitive and macro resources the design entry needs to be modified.

There has been subsequent work regarding implementation of multipliers on FPGAs [17-33]. These mainly focus on modifying the multiplier architecture to achieve performance improvement. However, there has been very limited effort in improving the performance by using embedded FPGA resources [34-36]. In this paper we carry

Manuscript received March 31, 2015. Received in revised form May 5, 2015.

B. Khurshid is with the National Institute of Technology, Srinagar, J & K, India (e-mail: burhan_07phd12@nitsri.net).

R. N. Mir is with the National Institute of Technology, Srinagar, J & K, India (e-mail: naaz310@nitsri.net).

doi: 10.11601/ijates.v4i2.115

out technology dependent optimizations of fixed-point multipliers by modifying the coding strategy at the design entry phase. This is achieved by writing functional and synthesizable codes that involve direct primitive and macro instantiations. This requires detailed information about the FPGA target family that is being used and the primitives that are supported. In our study different primitives have been used and system functionality has been distributed in a way that utilizes these components with perfect mappings rather than writing a functional code and allowing the synthesizer to distribute the logic through inferences. The study focuses on Spartan-6, Virtex-4 and Virtex-5 families. Detailed analysis is carried out and it is concluded that by using primitive instantiations a subsequent improvement in performance can be achieved. This is achieved without having to alter the data-time relation of the algorithm under consideration. The only tradeoff is that the design entry gets complicated.

The rest of the paper is as follows. Section II briefly discusses the fixed-point bit-parallel multipliers that have been considered in this work. Section III lists the primitives that have been used in this work. A brief description about each primitive is provided. Section IV carries out the actual synthesis and implementation. Conclusions are drawn in section V and references are listed at last.

II. BIT-PARALLEL MULTIPLIERS

In parallel multipliers number of partial products to be added is the main parameter that determines the performance of the multiplier. Bit-parallel multipliers process one whole word of the input sample each clock cycle and are ideal for high-speed applications. The multiplication process is carried out as shown in figure 1. In this paper three different bit-parallel multipliers are considered viz. Parallel ripple-carry array multipliers; Parallel carry-save array multipliers and Baugh-Wooley multipliers. The details of these multipliers could be found in [5]. The operands in each case are assumed to be in fixed-point 2's complement representation. Such a representation ensures a correct final result even if there is an intermediate overflow [5].

III. FPGA PRIMITIVES

Primitives are the components that make an FPGA. The exact nature of a primitive may vary from family to family. In this section we briefly describe the primitives that are used in this work. These belong to the Spartan-6, Virtex-4 and Virtes-5 families.

A. *BUFG* [38]

This design element is a high-fan-out global clock buffer that connects signals to the global routing resources for low skew distribution of the signal. BUFGs are typically used on clock nets as well other high fan-out nets like sets/resets and clock enables. The primitive is supported by all the three families under consideration.

B. *FDSE* [38]

FDSE is a single D-type flip-flop with clock enable and synchronous set. The synchronous set input, when high, overrides the clock enable input and sets the output high during the low-to-high clock transition. The data is loaded into the flip-flop when set is low and clock enable is high during the low-to-high clock transition. The primitive is supported by all the three families under consideration.

C. *LUT4_L* [38]

This design element is a 4-bit look-up table (LUT) with a local output that is used to connect to another output within the same configurable logic block (CLB). The primitive is supported by all the three families under consideration.

D. *LUT6_2* [38]

This design element is a 6-input, 2-output LUT that can implement any two 5-input logic functions with shared inputs, or implement a 6-input logic function and a 5-input logic function with shared inputs and shared logic values. The primitive is not supported by the Virtex-4 logic family.

E. *CARRY4* [38]

This primitive represents the fast carry logic for a slice. The carry chain consists of a series of four multiplexers and four XOR gates that connect to the other LUTs in the slice via dedicated routes to form more complex functions. The fast carry logic is useful for building arithmetic functions like adders, counters, subtractors etc. The primitive is not supported by the Virtex-4 logic family.

F. *MULT_AND* [38]

MULT_AND is an AND component used exclusively for building fast and smaller multipliers. The primitive is only supported by the Virtex-4 logic family.

G. *MUXCY_L* [38]

This primitive is a 2-to-1 multiplexer for carry logic and is used to implement a 1-bit high-speed carry propagate function. The primitive is only supported by the Virtex-4 logic family.

H. *XORCY* [38]

XORCY is a special XOR element with general output that generates faster and smaller arithmetic functions. The primitive is only supported by the Virtex-4 logic family.

I. *DSP48* [38]

This design element is a versatile, scalable, hard IP block that allows for the creation of compact, high-speed, arithmetic-intensive operations, such as those seen for many DSP algorithms. Some of the functions capable within the block include multiplication, addition, subtraction, accumulation, shifting, logical operations, and pattern detection. The primitive is supported by all the three families under consideration.

							A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
							B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	
							A_7B_0	A_6B_0	A_5B_0	A_4B_0	A_3B_0	A_2B_0	A_1B_0	A_0B_0	
						A_7B_1	A_6B_1	A_5B_1	A_4B_1	A_3B_1	A_2B_1	A_1B_1	A_0B_1		
					A_7B_2	A_6B_2	A_5B_2	A_4B_2	A_3B_2	A_2B_2	A_1B_2	A_0B_2			
				A_7B_3	A_6B_3	A_5B_3	A_4B_3	A_3B_3	A_2B_3	A_1B_3	A_0B_3				
			A_7B_4	A_6B_4	A_5B_4	A_4B_4	A_3B_4	A_2B_4	A_1B_4	A_0B_4					
		A_7B_5	A_6B_5	A_5B_5	A_4B_5	A_3B_5	A_2B_5	A_1B_5	A_0B_5						
	A_7B_6	A_6B_6	A_5B_6	A_4B_6	A_3B_6	A_2B_6	A_1B_6	A_0B_6							
A_7B_7	A_6B_7	A_5B_7	A_4B_7	A_3B_7	A_2B_7	A_1B_7	A_0B_7								
P_{15}	P_{14}	P_{13}	P_{12}	P_{11}	P_{10}	P_9	P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

Figure 1 Tabular form for Parallel Array multiplication

IV. SYNTHESIS AND IMPLEMENTATION

A. Methodology

The implementation in this work targets three different FPGA families viz. Spartan-6, Virtex-4 and Virtex-5. Only LX series has been considered as it is apt for general logic applications. The implementation is carried out for an input operand length varying from 4 to 32 bits. The parameters considered are resource utilization, timing and dynamic power dissipation. Resource utilization is considered in terms of on chip FPGA components used. Timing refers to the clock speed of a design and is limited by the setup time of the input/output registers, propagation and routing delays associated with the critical path, clock to output time associated with the flip flops and the skew between the launch (input) register and the capture (output) register. Timing analysis is done to provide information about the speed/throughput of the system. Dynamic power dissipation is related to charging and discharging of node capacitances along the different switching elements. To ensure a fair comparison, similar test benches have been used for all the implemented designs i.e. the input statistics remain the same in each case. The initial design entry is done using VHDL. The coding strategy is based on instantiation of different primitives listed in section III. However, for comparison we have also followed the conventional inferential approach. The constraints relating to the period and offsets are duly provided and a complete timing closure is ensured. The design synthesis, mapping and translation are carried out in Xilinx ISE 12.1 and the simulator database is then analyzed for on-chip resources, throughput and timing metrics. Power metrics are obtained using Xpower analyzer.

B. Experimental results

As mentioned earlier, for each implementation a traditional inferential coding strategy is followed. Synthesis based on this coding strategy utilizes the FPGA resources as general logic elements. This will serve as a standard against which other implementations will be compared. Metrics associated with the instantiation of various primitives are named as per the primitives used. Tables 1, 2 and 3 give a comparison of the on chip resources utilized by different primitives for an input word-length of 16 bits. The architectures considered are the bit-parallel RCA, CSA and BW multipliers. The target device is XC6SLX16 from Spartan-6.

It is observed that by instantiating primitives and macro blocks there is a subsequent reduction in the on-chip resources being utilized by a particular structure. This is achieved without having to modify any architectural details. The most area efficient structure is obtained with LUT6_2 primitive because of its ability to implement both sum and carry in a single LUT. LUT4-L uses two different 4 input LUTs to implement the sum and the carry parts in each processing cell of the array. The CARRY4 and DSP48 primitives provide fast carry logic for each row. Their inclusion prominently will affect the timing properties of the structure. However, there is still some reduction in the slices being utilized when compared to the basic structure generated through inferential coding style. Further analysis is carried out for different multiplier structures for varying word-lengths and different target families. The metrics obtained from the synthesizer database are then plotted as a function of operand word-lengths and are presented in figures 2, 3 and 4. For simplicity we have considered only the occupied slices in each case. Virtex-4 family does not support the LUT6_2 primitive and hence does not appear in the plot. Further the fast carry logic in this family is implemented using a combination of MULT_AND and MUXCY_L primitives.

TABLE 1
RESOURCE UTILIZATION FOR RCA MULTIPLIER ON SPARTAN-6

On-chip resource	**Inferential coding style**	**LUT4_L**	**LUT6_2**	**CARRY4**	**DSP48***
Registers	32	32	32	32	32
LUTs	943	359	**286**	385	168
Slices	361	99	**82**	143	133

* DSP48 uses additional resources in the form of 16 DSP48A1 blocks

TABLE 2
RESOURCE UTILIZATION FOR CSA MULTIPLIER ON SPARTAN-6

On-chip resource	**Inferential coding style**	**LUT4_L**	**LUT6_2**	**CARRY4**	**DSP48***
Registers	32	32	32	32	32
LUTs	686	420	**343**	385	525
Slices	197	101	**92**	127	163

* DSP48 uses additional resources in the form of 16 DSP48A1 blocks

TABLE 3
RESOURCE UTILIZATION FOR BW MULTIPLIER ON SPARTAN-6

On-chip resource	**Inferential coding style**	**LUT4_L**	**LUT6_2**	**CARRY4**	**DSP48***
Registers	32	32	32	32	32
LUTs	583	492	**373**	407	500
Slices	206	128	**109**	134	174

* DSP48 uses additional resources in the form of 16 DSP48A1 blocks

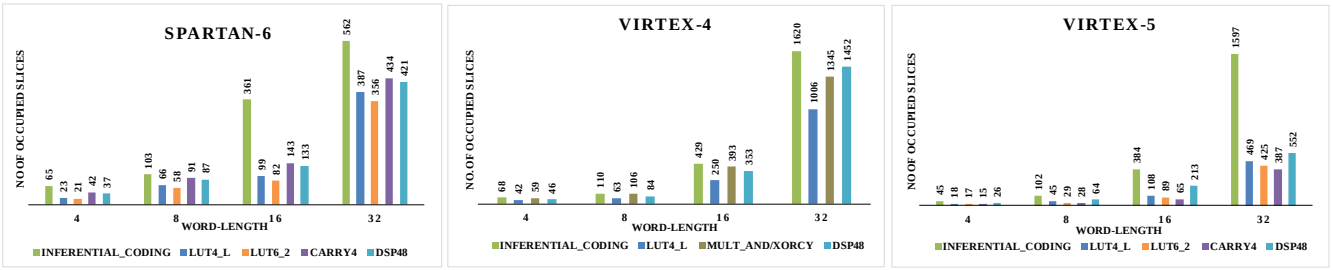


Figure 2 Resource utilization for RCA multiplier on different FPGA families

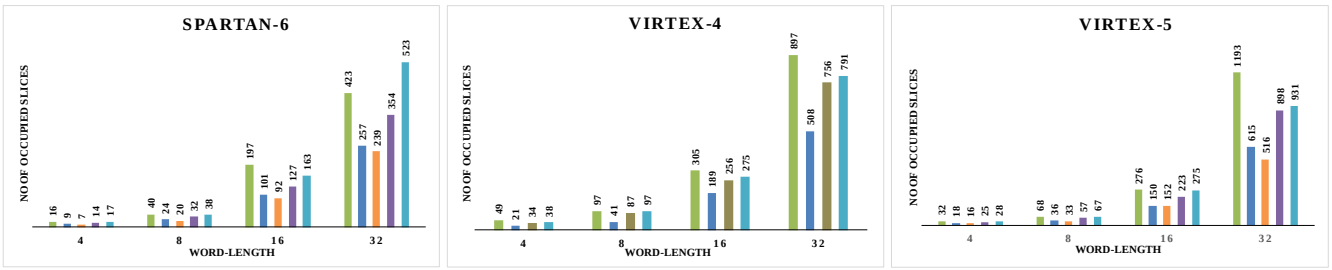


Figure 3 Resource utilization for CSA multiplier on different FPGA families

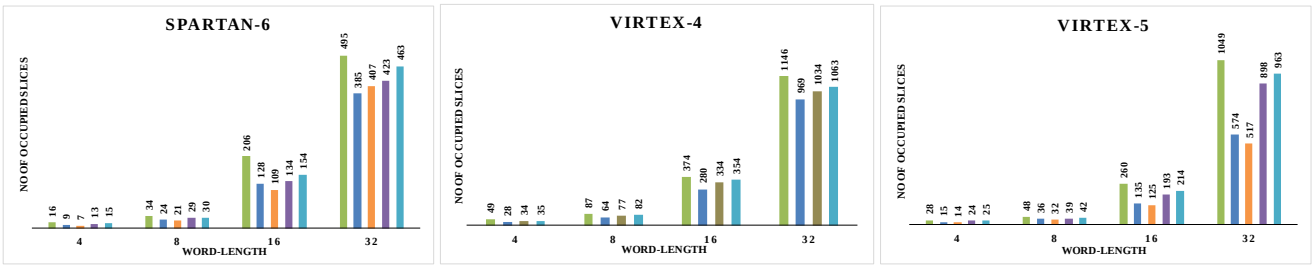


Figure 4 Resource utilization for BW multiplier on different FPGA families

It is observed that in each case there is a substantial reduction in the area when the structures are generated through instantiation of different primitive components. Also different primitives give different area performances depending upon the logic they implement. If area is the parameter of interest LUT6_2 gives the best performance.

The use of primitives LUT4_L and LUT6_2 although reduces the overall logic being used but the logic associated with the critical path of the structure is increased. This is indicated by the increase in the number of logic levels in the critical path. As a result the logic delay and the associated route delay increases. However, the fast carry logic associated with the CARRY4 primitive makes the addition process really fast resulting in reduced route delays. For Virtex-4 devices the fast carry logic is implemented using a combination of MULT_AND, MUXCY_L and XORCY primitives. The use of CARRY4 logic enhances the speed only in case of RCA multipliers as the critical path is limited by the rippling of the generated carry in each cell. However, with CSA and BW multipliers there is no rippling of the carry in the main structure. The only part of the multiplier that is enhanced using the fast carry logic is the vector merging adder (VMA). Tables 4, 5 and 6 provide a comparison of the maximum achievable clock rates post implementation for a word length of 16 bits. The target family is Spartan-6. The structures generated through instantiation of different primitives tend to have better timing closures in terms of the relationship between an external clock pad and its associated data-in or data-out pad. This is indicated by the offset-in and offset-out metrics from the timing database of the synthesizer. The values are included in the tables and are indicative of the fact that with primitive instantiations better timing behavior is achieved.

TABLE 4
TIMING ANALYSIS FOR RCA MULTIPLIER ON SPARTAN-6

Timing Parameter	**Inferential coding style**	**LUT4_L**	**LUT6_2**	**CARRY4**	**DSP48**
Maximum frequency (MHz)	30.67	23.186	23.89	38.7	**144.38**
Minimum available offset-in (ns)	5.112	5.018	2.568	**2.118**	2.543
Minimum available offset-out (ns)	11.727	8.488	10.047	6.782	**2.765**

TABLE 5
TIMING ANALYSIS FOR CSA MULTIPLIER ON SPARTAN-6

Timing Parameter	**Inferential coding style**	**LUT4_L**	**LUT6_2**	**CARRY4**	**DSP48**
Maximum frequency (MHz)	50.182	29.3	27.42	53.987	**167.87**
Minimum available offset-in (ns)	6.017	5.245	3.28	2.87	**2.521**
Minimum available offset-out (ns)	11.851	9.335	8.813	10.474	**4.78**

TABLE 6
TIMING ANALYSIS FOR BW MULTIPLIER ON SPARTAN-6

Timing Parameter	Inferential coding style	LUT4_L	LUT6_2	CARRY4	DSP48
Maximum frequency (MHz)	50.117	29.36	29.216	52.987	**165.43**
Minimum available offset-in (ns)	6.346	5.154	3.052	2.66	**2.543**
Minimum available offset-out (ns)	10.095	9.517	8.531	9.987	**4.87**

The results also indicate that CSA and BW multipliers have higher operating frequencies when compared to the RCA multiplier structures. Further analysis is carried out by plotting the maximum achievable speed against the operand word lengths for different structures and for different target families. The results are shown in figures 5, 6 and 7. Again for simplicity only the maximum achievable speeds have been considered.

It is observed from the plots that the use of fast carry logic results in faster execution and thus higher clock frequencies are achievable. The effect is more prominent in RCA multiplier as the carry rippling is completely eliminated.

Finally dynamic power dissipation for different structures is considered. Because an FPGA is programmable, it is only natural to look into minimizing the power dissipated. The dynamic power dissipation in a CMOS circuit is a function of the input voltage (V^2), the clock frequency (f_{clk}), the switching activity (α), the total capacitance seen by a particular node (C_L) and the number of elements used (σ). The analysis was done for a constant supply voltage and at maximum operating frequency for each structure. To ensure a reasonable comparison the test vectors provided during post route simulation were selected to represent the worst case scenario for data coming into the multiplier block. Same test bench was used for all the synthesized structures. The design node activity from the simulator database along with the power constraint file (PCF) was used for power analysis in the Xpower analyzer tool. Table 7 shows the power dissipated in various resources for RCA multiplier for operand length of 16 bits. The targeted device is Spartan-6. Tables 8 and 9 show the same metrics for CSA and BW structures.

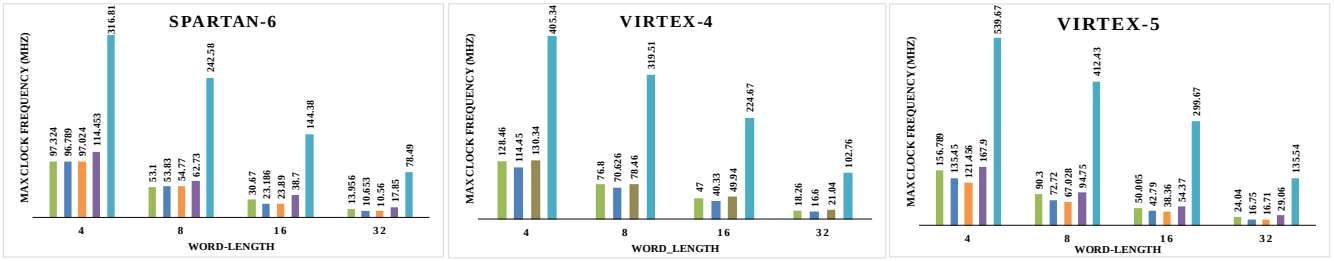


Figure 5 Maximum clock frequency comparisons for RCA multiplier on different FPGA families

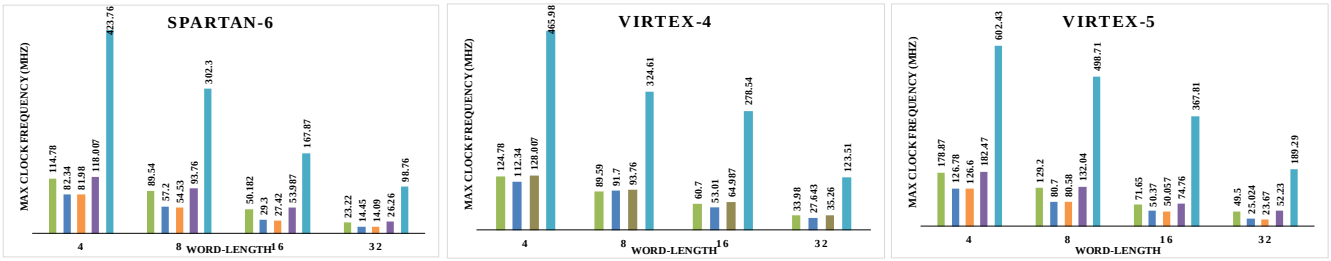


Figure 6 Maximum clock frequency comparisons for CSA multiplier on different FPGA families

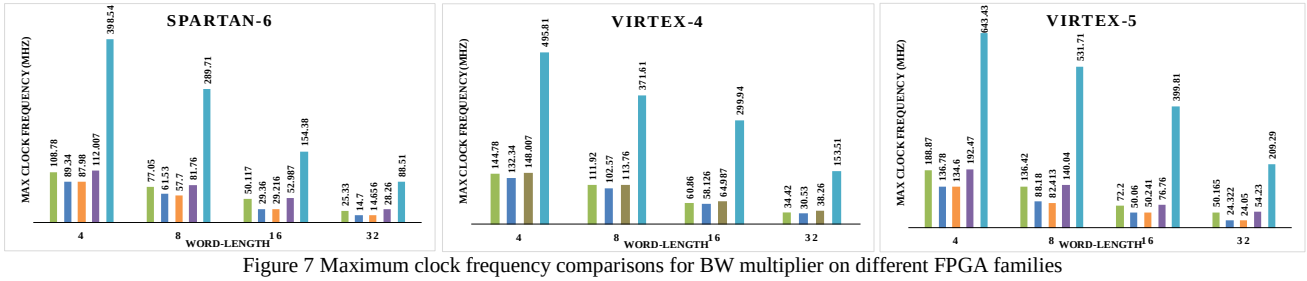


Figure 7 Maximum clock frequency comparisons for BW multiplier on different FPGA families

TABLE 7
POWER DISSIPATION FOR RCA MULTIPLIER ON SPARTAN-6

FPGA resource	Power dissipated (mW)				
	Inferential coding style	LUT4_L	LUT6_2	CARRY4	DSP48
Clock	0.54	**0.47**	**0.47**	0.64	2.8
Logic	2.34	1.64	1.78	1.4	**0.87**
Signals	4.41	1	**0.88**	1.16	1.23
I/Os	6.01	5.16	6.39	4.63	**2.54**
Total	13.3	8.27	9.52	7.83	**7.44**

TABLE 8
POWER DISSIPATION FOR CSA MULTIPLIER ON SPARTAN-6

FPGA resource	Power dissipated (mW)				
	Inferential coding style	LUT4_L	LUT6_2	CARRY4	DSP48
Clock	1.24	**0.7**	0.81	1.24	3.25
Logic	3.14	2.18	2.27	1.14	**0.69**
Signals	2.66	1.41	**1.2**	1.66	1.63
I/Os	5.44	5.35	5.12	4.44	**2.31**
Total	12.48	9.64	9.4	8.48	**7.88**

TABLE 9
POWER DISSIPATION FOR BW MULTIPLIER ON SPARTAN-6

FPGA resource	Power dissipated (mW)				
	Inferential coding style	LUT4_L	LUT6_2	CARRY4	DSP48
Clock	1.22	**0.63**	0.88	1.24	3.43
Logic	2.71	2.25	2.02	1.14	**0.67**
Signals	3.29	1.39	**1.11**	1.96	1.97
I/Os	4.99	6	5.75	4.42	**2.36**
Total	12.21	10.27	9.76	8.76	**8.43**

The power dissipated in the clocking resources varies with the clock activity (clock frequency) as provided in the PCF. Since each structure is operated at its maximum operating frequency, the power dissipated by the clock varies accordingly and has a maximum value for the multiplier based on CARRY4 and DSP48 primitives. However, the capacitance $C_{L,}$ which needs to be driven at each toggling node, varies with the type, fan-out, and capacitance of the logic and routing resources used in the design. The use of primitives through instantiations has a soothing effect on the fan-out of the non-clocking nets. This is indicated in table 10 where the average fan-out of non-clocking nets for different multipliers using different primitives has been enlisted for a 16-bit operand word-length. In addition, there is a reduction in the number of elements (σ) being utilized by different multiplier structures when designed using direct instantiation of primitives. Thus, the power dissipated in the logic is reduced and has a minimum value for CARRY4 and DSP48 primitives. The reduction in the power dissipation in the signals and I/Os is indicative of the fact that primitive instantiation also tends to relax the signal transition rates for the duration of operation.

Further analysis is carried out by plotting the total dynamic power dissipation as a function of input word-length for different multiplier structures and for different FPGA families. The results are shown in figures 8, 9 and 10.

TABLE 10
AVERAGE FAN-OUT OF NON-CLOCKING NETS FOR DIFFERENT MULTIPLIERS ON SPARTAN-6

Multiplier design	Average fan-out of non-clock nets				
	Inferential coding style	LUT4_L	LUT6_2	CARRY4	DSP48
RCA	5.52	2.94	2.32	**1.98**	--
CSA	4.75	2.71	2.22	**1.78**	--
BW	4.78	2.65	2.12	**1.66**	--

For DSP systems it is more appropriate to quantify the power efficiency through energy analysis [39]. This gives idea about the power requirements of a design at a lower level. Three energy related parameters are defined for different multiplier designs. These include Energy per operation (EOP), which is the average amount of energy required to compute one operation; Energy throughput (ET) which is the energy dissipated for every output bit processed and Energy density (ED) which is the energy dissipated per FPGA slice. Tables 11, 12 and 13 provide these metrics for different designs. The input operand length in 16 bits and the target device is from Spartan-6. In each case the critical path delay is taken as the approximate time to complete one operation. Further analysis is carried out by plotting the energy metrics as a function of operand word length for different multipliers. The plots appear in figures 11, 12 and 13. The target device in each case is XC6SLX16 from Spartan-6.

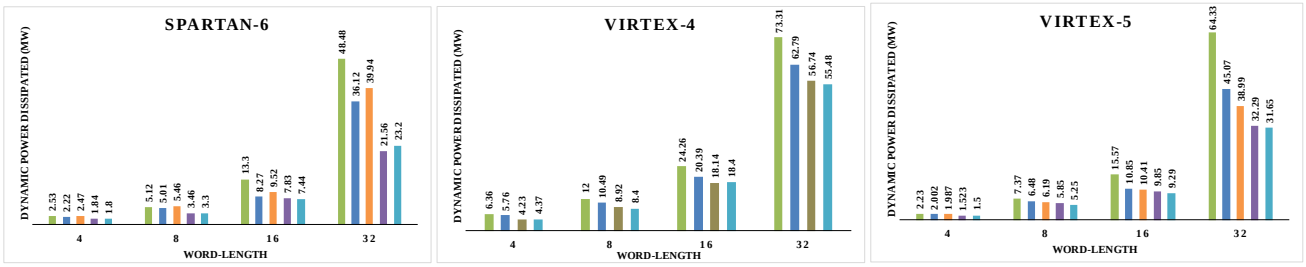


Figure 8 Dynamic Power dissipation comparisons for RCA multiplier on different FPGA families

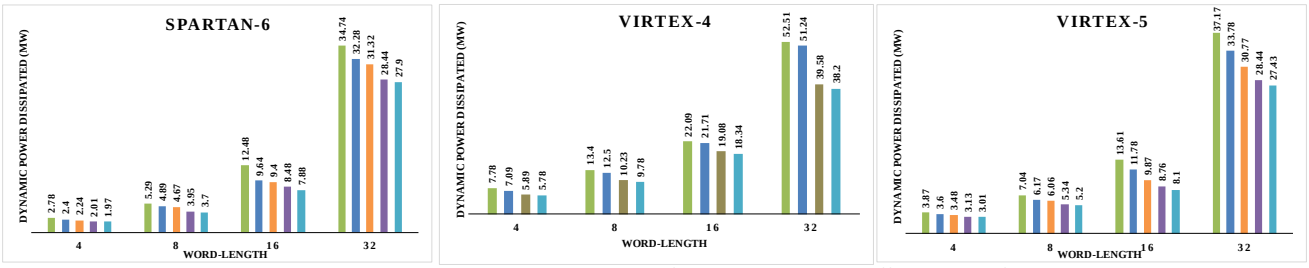


Figure 9 Dynamic Power dissipation comparisons for CSA multiplier on different FPGA families

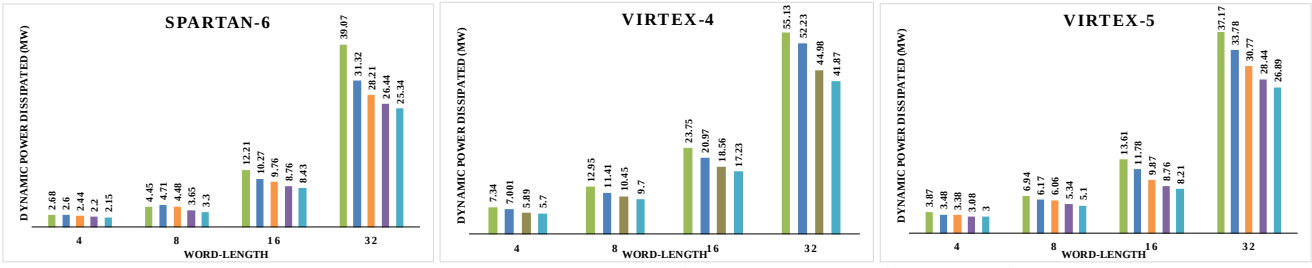


Figure 10 Dynamic Power dissipation comparisons for BW multiplier on different FPGA families

The plots clearly reveal that the structures based on primitive instantiations have high power efficiency. The energy requirement is minimum for the structures based on CARRY4 and DSP48 primitives. Also, note that the effect is more prominent for RCA multipliers as the entire structure is synthesized using the CARRY4 primitive, where as in the CSA and BW multipliers only the VMA part is based on the fast carry logic.

TABLE 12
ENERGY ANALYSIS FOR RCA MULTIPLIER ON SPARTAN-6

Energy parameter	Inferential coding style	LUT4_L	LUT6_2	CARRY4	DSP48*
EOP (pJ)	433.64	356.68	398.49	202.32	**51.530**
ET (pJ/bit)	27.103	22.29	24.90	12.645	**3.220**
ED (pJ/slice)	1.2012	3.602	4.859	1.4148	**0.387**

TABLE 12
ENERGY ANALYSIS FOR CSA MULTIPLIER ON SPARTAN-6

Energy parameter	Inferential coding style	LUT4_L	LUT6_2	CARRY4	DSP48*
EOP (pJ)	248.694	329.01	342.8	157.07	**46.94**
ET (pJ/bit)	15.543	20.56	21.42	9.81	**2.93**
ED (pJ/slice)	1.2624	3.25	3.72	1.23	**0.28**

TABLE 13
ENERGY ANALYSIS FOR BW MULTIPLIER ON SPARTAN-6

Energy parameter	Inferential coding style	LUT4_L	LUT6_2	CARRY4	DSP48*
EOP (pJ)	243.62	349.79	334.06	165.33	**54.60**
ET (pJ/bit)	15.22	21.86	20.87	10.33	**3.41**
ED (pJ/slice)	1.18	2.73	3.06	1.23	**0.35**

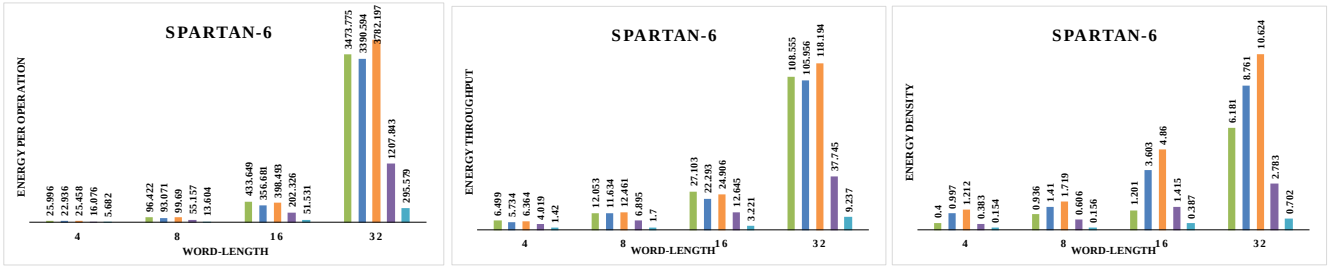


Figure 11 Energy analyses for RCA multiplier on Spartan-6 FPGA family

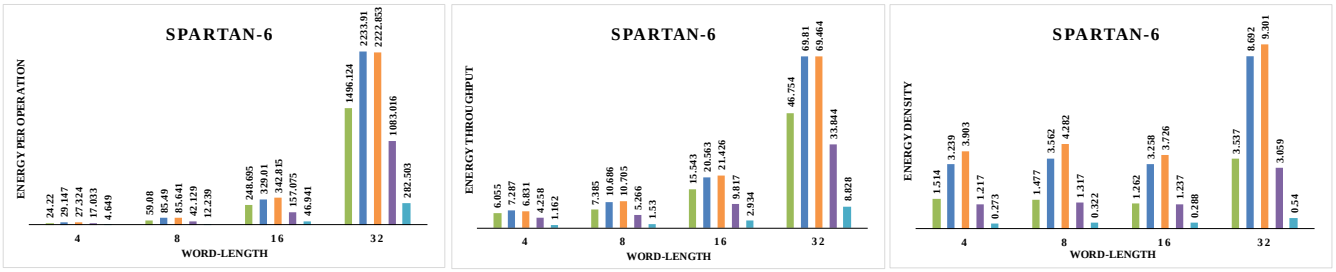


Figure 12 Energy analyses for CSA multiplier on Spartan-6 FPGA family

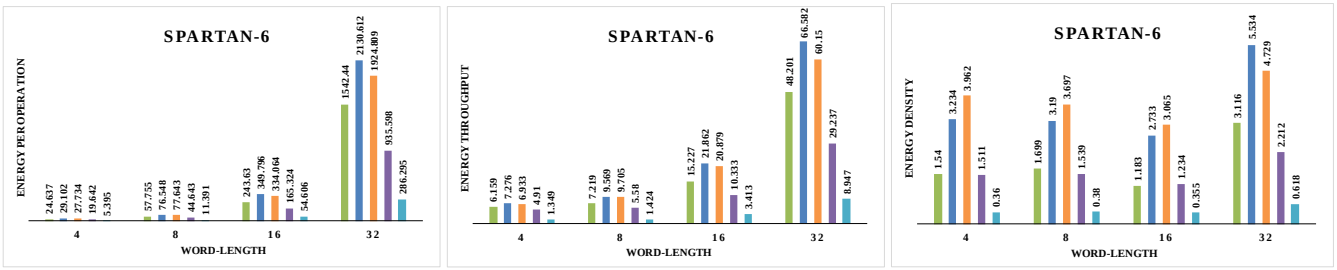


Figure 13 Energy analyses for BW multiplier on Spartan-6 FPGA family

V. CONCLUSIONS AND FUTURE SCOPE

This paper implemented the bit-parallel fixed-point multipliers in three different structures. The hardware implementations presented in this paper were based on the use of various in built primitives and macro blocks inherent to modern FPGAs. The analysis and the experimental results carried out in this paper clearly indicate that a considerable improvement in performance is indeed achievable by using these primitives. Further the design entry used in this paper was based on instantiations rather than inferences. By using a coding strategy based on instantiations the on-chip FPGA components can be used in a manner that fully utilizes their potential. Also a judicious choice of primitives will ensure that a particular performance parameter is enhanced as may be required by any particular application. This paper deliberately ruled out any architectural modification that may be carried out at the top level of the design. The idea was to present a clear cut analysis that will provide an insight about the performance speed-up that may be achieved by utilizing the huge primitive support provided by FPGA families. Currently the authors are working on achieving a performance speed-up by using a combination of architectural modifications and embedded primitives in FPGAs.

REFERENCES

[1]. G. L. Narayan and B. Venkataramani, " Optimization Techniques for FPGA based Wave Pipelined DSP Blocks," IEEE Transc.Very Large Scale Integr. (VLSI) syst., vol. 13, No. 7, pp. 783-792, July 2005.

[2]. M. A. Ashour and H. I. Saleh, "An FPGA Implementation guide for some different types of Serial-Parallel Multiplier Structures," Microelectronics Journal, vol. 31, pp. 161-168, 2000.
[3]. K. Compton, S. Hauck, "Reconfigurable Computing: A survey of Systems and Software," ACM Computing Surveys, vol. 34, No. 2, pp. 171-210, June 2002.
[4]. R. Tessier, W. Burleson, "Reconfigurable Computing and Digital Signal Processing: Past, Present and Future," Programmable Digital Signal Processors, Yu Wen Hue d, Marcel Dekker, pp. 147-186, 2002.
[5]. Keshab K. Parhi, "VLSI Digital Signal Processing Systems Design and Implementation," Wiley, 1999.
[6]. S. Shanthala and S. Y. Kulkarni, "VLSI Design and Implementation of Low Power MAC Unit with Block Enabling Technique," European Journal of Scientific Research, ISSN 1450-216X, vol. 30, No. 4, pp. 620-630, 2009.
[7]. K. H. Chen, Y. H. Chen and Y. S. Chu, "A Versatile Multimedia Functional Unit Design using the Spurious Power Suppression Technique," in Proc. IEEE Asian Solid-State Circuits conf., 2006, pp. 111-114.
[8]. Roger Woods, John McAllister, Gaye Lightbody and Ying Yi, "FPGA-based Implementation of Signal Processing Systems," Wiley, 2008.
[9]. Z. Guo, W. Najjar, F. Vahid and K. Vissers, "A Quantitative Analysis of the Speed up Factors of FPGAs over Processors," in Proc. Int. Symp. on FPGAs, ACM Press, 2004.
[10]. K. Underwood "FPGAs vs. CPUs: Trends in Peak Floating-Point Performance," in Proc. Int. Symp. on FPGAs, ACM Press, 2001.
[11]. G. Stitt, F. Vahid and S. Nematbakhsh, "Energy Savings and Speed ups from Partitioning Critical Software Loops to Hardware in Embedded systems," ACM Transc. Embedded Comput. Systems, vol. 3, pp. 218-232, 2004.
[12]. R. Tessier and W. Burleson, "Reconfigurable Computing for DSP: A Survey," Journal of VLSI Signal Processing, vol. 28, pp. 7-27, 2001, Kluwer Academic Publisher.
[13]. T. J. Todman, G. A. Constantinides, S. J. E. Wilton, O. Mencer, W. Luk and P. Y. K. Cheung, "Reconfigurable Computing: Architecture and Design Methods," in IEEE Proc. Comput. Digit. Tech., vol. 152, No. 2, March 2005.
[14]. K. S. Hemmert and K. D. Underwood, "Fast, Efficient Floating-Point Adders and Multipliers for FPGAs," ACM Transactions on Reconfigurable Technology and Systems, vol. 3, No. 3, Article 11, September 2010.
[15]. G. Quan, J. P. Davis, S. Devarkal and D. A. Buell, "High-Level Synthesis for Large Bit-Width Multipliers on FPGAs: A Case Study," ACM 2005.
[16]. Steve Kilts, "Advanced FPGA Design Architecture, Implementation, and Optimization," Wiley 2007.
[17]. Seetharaman Ramachandran "Digital VLSI Systems Design: A Design Manual for Implementation of Projects on FPGAs and ASICs using Verilog," Springer, 2011.
[18]. M. Shand, P. Bertin, and J. Vuillemin, "Hardware Speedups in Long Integer Multiplication," Computer Architecture News, vol. 19, No. 1, pp. 106–114, 1991.
[19]. L. Louca, T. A. Cook, and W. H. Johnson, "Implementation of IEEE Single Precision Floating Point Addition and Multiplication on FPGAs," in ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Monterey, CA, pp. 107–116, Feb. 1996.
[20]. F. de Dinchin and V. Lef`evre, "Constant Multipliers for FPGAs," in Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, H.R. Arabnia (Ed.), CSREA Press, vol. I, pp. 167–173, June 2000.
[21]. T. Courtney, R. Turner, and R. Woods, "Multiplexer Based Reconfiguration for Virtex Multipliers," in Field-Programmable Logic and Applications. Proceedings of the 9th International Workshop, FPL 2000, pp. 749–758, 2000.
[22]. T. Courtney, R. Turner, and R. Woods, "An Investigation of Reconfigurable Multipliers for use in adaptive Signal Processing," in Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines (FCCM '00), IEEE Computer Society Press, pp. 341–343, April 2000.
[23]. A. F. Tenca, M. D. Ercegovac, and M. E. Louie, "Fast On-Line Multiplication Units Using LSA Organization," in Proceedings of the International Society of Optical Engineering (SPIE). Visual Communications and Image Processing. Real-Time Signal Processing, vol. 3807, pp. 74–83, 1999.
[24]. C. Wallace, "A Suggestion for a Fast Multiplier," IEEE Transactions on Electronic Computers, 13:14–17, 1964.
[25]. Z. Wang and W. C. Miller, "A new Design Technique for Column Compression Multipliers," IEEE Transactions on Computers, vol. 44:962–970, 2005.
[26]. F. Cheng and M. Theobald, "Design of Synchronous Variable Latency Pipelined Multipliers.," IEEE Transaction on Computers, vol. 49: 659-672,2005.
[27]. Z. Huang, "High Level Optimization Techniques for Low Power Multiplier Design" Ph.D. Thesis, University of California, los angels, 2003.
[28]. C. H. Chang and R. K. Satzoda, "A Low Error and High Performance Multiplexer-Based Truncated Multiplier," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 18, No. 12, December 2010.
[29]. S. S. Kidambi, F. E. Guibaly and A. Antoniou, "Area-Efficient multipliers for Digital Signal Processing Applications," IEEE Transactions on Circuits and Systems –II: Analog & Digital Signal Processing, Vol. 43, No. 2, February 1996.
[30]. J. E. Stine and O. M. Duverne, "Variations on Truncated Multiplication," Proceedings of the Euromicro Symposium on Digital System Design, 2003.
[31]. Y. M. Motey and T. G. Panse, "Hardware Implementation of Truncated Multiplier Based on Multiplexer Using FPGA," International conference on Communication and Signal Processing, April 3-5, 2013.
[32]. H. Park and E. E. Swartzlander, "Truncated Multiplications for the Negative Two's Complement Number System," 49th IEEE International Midwest Symposium on Circuits and Systems, San Juan, August 6-9, 2006.
[33]. J. Valls and E. Boemo, "Efficient FPGA Implementation of Two's Complement Digit-Serial/Parallel Multipliers," IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, Vol. 50, No. 6, June 2003.
[34]. G. Zhou, L. Li and H. Michalik, "Area optimization of bit parallel finite field multipliers with fast carry logic on FPGAS," International Conference on Field Programmable Logic and Applications, 2008.
[35]. S. Gao, D. A. Khalili and N. Chabini, "Efficient Scheme for Implementing Large Size Signed Multipliers UsingMultigranular Embedded DSP Blocks in FPGAs," International Journal of Reconfigurable Computing Vol. 2009, Article ID 145130, Hindawi Publishing Corporation.
[36]. C. Ingemarsson, P. Kallstrom and O. Gustafsson, "Using DSP block pre-adders in pipeline SDF FFT implementations in contemporary FPGAs," 22nd International Conference on Field Programmable Logic and Applications, August 2012.
[37]. C. R. Baugh and B. Wooley, "A two's Complement Parallel Array Multiplication Algorithm," IEEE Trans. On Computers, vol. C-22, No. 12. Pp. 1045-1047, Dec. 1973.
[38]. http://www.xilinx.com
[39]. P. K. Meher, S. Chanderasekaran and A. Amira, "FPGA Realization of FIR Filters by Efficient and Flexible Systolization using Distributed Arithmetic," IEEE Transactions on Signal Processing, vol. 56, No. 7, July 2008.

AUTHOR PROFILES

B. Khurshid received the B.E. degree in Electronics and Communications Engineering from the Kashmir University, India, in 2008, the M.Tech degree in Communications and IT from National Institute of Technology, Srinagar, India in 2011. Currently he is pursuing his PhD in System design in the department of Computer Science and Engineering, NIT, Srinagar. His research interests include Reconfigurable architectures, Platform oriented solutions for arithmetic and DSP algorithms, Architectural and technology dependent optimizations targeted for FPGA platforms, etc. He has many publications in the related field and is a student member of IEEE. He is also a lifetime member of IETE.

R. N. Mir received B.E. (Hons) in Electrical Engineering from University of Kashmir (India) in 1985, M.E. in Computer Science & Engineering from IISc Bangalore (India) in 1990 and Ph D from University of Kashmir, (India) in 2005. She is currently a Professor in the department of Computer Science & Engineering at NIT Srinagar, India. She is the co-author of many scientific publications in international journals and conferences. Her current research interests include reconfigurable computing, security and routing in wireless ad-hoc networks and sensor networks

Advanced Measurements of the Aggregation Capability of the MPT Network Layer Multipath Communication Library

Gábor Lencse, Ákos Kovács

***Abstract*—The MPT network layer multipath communication library is a novel solution for several problems including IPv6 transition, reliable data transmission using TCP, real-time transmission using UDP and also wireless network layer routing problems. MPT can provide an IPv4 or an IPv6 tunnel over one or more IPv4 or IPv6 communication channels. MPT can also aggregate the capacity of multiple physical channels. In this paper, the channel aggregation capability of the MPT library is measured up to twelve 100Mbps speed channels. Different scenarios are used: both IPv4 and IPv6 are used as the underlying and also as the encapsulated protocols and also both UDP and TCP are used as transport protocols. In addition, measurements are taken with both 32-bit and 64-bit version of the MPT library. In all cases, the number of the physical channels is increased from 1 to 12 and the aggregated throughput is measured.**

***Keywords*—channel capacity aggregation, network layer multipath communication, performance analysis, TCP/IP protocol stack, tunneling**

I. Introduction

Multipath communication is a hot research topic today. There were different solutions invented: the multipath technology can be used in different layers (link layer, network layer, transport layer) see our little survey in the next section. Now, we focus on the MPT network layer multipath communication library [1], which one was developed at the *Faculty of Informatics*, *University of Debrecen*, Debrecen, Hungary. It can be freely downloaded for 32-bit and 64-bit Linux operating systems as well for Raspberry Pi from [2]. It makes possible to aggregate the transmission capacity of multiple interfaces of a device. Its performance, especially its channel aggregation capability for two channels was analyzed in [3] and for four channels in [4] using serial links with the speed of a few megabits per second.

We measured the channel aggregation capability of the MPT network layer multipath communication library using significantly increased number of physical channels and transmission speed compared to the earlier test of other researchers [3] and [4]. Our preliminary results concerning the 32-bit version of the MPT library measured by the industrial standard `iperf` tool using the UDP transport layer protocol were published in our conference paper [5], which one is now extended with the HTTP measurements (using TCP) and with the testing of the 64-bit version of the MPT library.

The remainder of this paper is organized as follows. First, the different multipath solutions are surveyed in a nutshell. Second, a brief introduction is given to the MPT network layer multipath communication library. Third, our test environment is described. Fourth, our experiments are described, the results of our high number of measurements are presented and discussed. Fifth, the directions of our future research are outlined. Finally, our conclusion is given.

Manuscript received February 26, 2015, revised May 9, 2015.

G. Lencse is with the Department Telecommuications, Széchenyi István University, Győr, Hungary (phone: +36-96-613-665, fax: +36-96-613-646, e-mail: lencse@sze.hu)

Á. Kovács is with the Department Telecommuications, Széchenyi István University, Győr, Hungary (e-mail: kovacs.akos@sze.hu)

II. A Short Survey of Multipath Solutions

A. Multipath TCP – a Transmission Layer Solution

Multipath TCP [6] is probably the most well-known multipath solution. MPTCP uses multiple TCP sub-flows on the top of potentially disjoint paths, see Fig. 1. Therefore it can be used for the aggregation of the transmission capacity of the underlying paths. Its channel aggregation can be very efficient: a single data-stream was transmitted at the rate of 50Gbps over six 10Gbps Ethernet Links using MPTCP [7]. MTPCP is actively researched and analyzed from different viewpoints see e.g. [8] and its references or count the Google Scholar hits for "Multipath TCP".

However, multipath TCP has its limitations and drawbacks, too. TCP provides a reliable byte stream transmission, which one is appropriate for several applications such as web browsing, sending or downloading e-mails, etc. However, its retransmission mechanism is undesirable for other applications such as IP telephony, video conference or other real-time communications where some packet loss (with low ratio) can be better tolerated than high delays caused by TCP retransmissions. Consequently, multipath TCP is not suitable for these types of applications.

B. MPT Library – the Only Network Layer Solution

The MPT network layer multipath communication library [1] uses UDP/IP protocols on the top of each link layer connection and creates an IP tunnel over them. Thus both TCP and UDP can be used over the IP tunnel, see Fig. 2. Therefore retransmissions can be omitted if they are not required. This design makes MPT more general than MPTCP thus permitting MPT more areas of applications.

The MPT library may be used for many different purposes including file and stream transmission [4], cognitive infocommunication [9], wireless network layer roaming problems

[10] and changing the communication interfaces (using different transmission technologies) without packet loss [11] (it is also called vertical handover between 3G and WiFi). For further publications about MPT, see [12] and [13].

As far as we know, MPT is the only network layer multipath communication solution.

C. OLiMPS – a Link Layer Solution

The Openflow Link-Layer Multipath Swithcing [14] is a novel solution, which uses the logic of the link-layer, that is, it calculates routes as if the nodes were connected with LANs, however, it can also operate over WANs [15].

D. Other Similar Solutions

There are some other solutions, which deal with multiple interfaces, however they are not always real multipath solutions.

The *Multiple Interfaces* Working Group of IETF has already produced many useful documents [16]. They focus on the problem that a host has multiple interfaces which are connected to different provisioning domains [17] and the interfaces can be simultaneously used for communication. It is not necessarily a multipath solution: for example, one application may use the first interface, and another one may use the second one.

Proxy Mobile IPv6 [18] allows a mobile node to connect to the same PMIPv6 domain through different interfaces. The *NETEXT* Working Group of IETF proposed a draft RFC [19] which specifies protocol extensions to PMIPv6 to distribute specific traffic flows on different physical interfaces.

III. MPT in a Nutshell

A. The Architecture of MPT

Fig. 2 shows the layered architecture of the MPT network layer multipath communication library. The most important difference from MPTCP is that MPT creates a new logical interface on the endpoint host, through which the applications can communicate, therefore the applications can use any transport layer protocol: either TCP or UDP, whichever is appropriate for them. The MPT software processes the packets from the tunnel interface. MPT makes a packet-by-packet decision about which path to choose and then encapsulates the packet into a new UDP/IP packet and finally sends it out through the appropriate link-layer interface [1].

B. The Configuration and Usage of the MPT Library

The MPT library distribution contains an easy to follow user guide [20]. To be able to use MPT between two computers, the software must be installed on both of them. One of them should be configured as *server* and the other one as *client*, but the applications see it completely symmetrical. The MPT library has simple and straight forward configuration files where the different parameters (e.g. the number of physical connections, the Linux network interface names and IP addresses for each channel, the name of the tunnel interface, etc.) can be set. When both sides are configured and the MPT

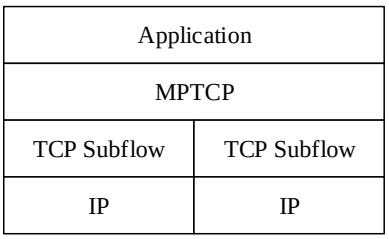


Fig. 1. The architecture of the MPTCP protocol stack [6]

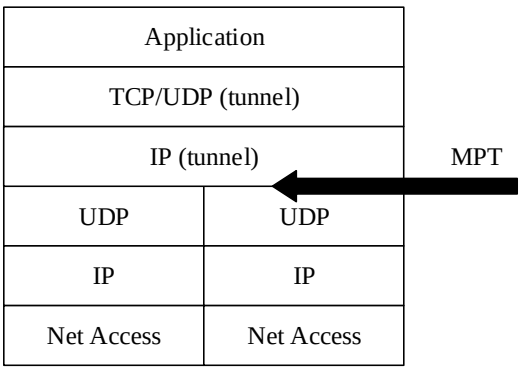


Fig. 2. The layered architecture of the MPT software [3]

software is started on both computers, the applications can use the tunnel interfaces for communication in the usual way. The MPT library distributes the user's traffic for all the configured physical channels thus the user can take the advantage of the multiple network interfaces.

IV. Test Environment

A. Hardware and Basic Configuration

Two DELL Precision Workstation 490 computers were used for our tests. Their basic configuration was:

- DELL 0GU083 motherboard with Intel 5000X chipset
- Two Intel Xeon 5140 2.33GHz dual core processors
- 8x2GB 533MHz DDR2 SDRAM (accessed quad channel)
- Broadcom NetXtreme BCM5752 Gigabit Ethernet controller (PCI Express, integrated)

Three Intel PT Quad 1000 type four port Gigabit Ethernet controllers were added to each computers. The 3x4=12 Gigabit Ethernet ports were used for the measurements and the integrated one was used for control purposes. The computers were interconnected by a Cisco Catalyst 2960 switch limiting the transmission speed to 100Mbps and separating the 12 physical connections by VLANs.

In our experiments, both IPv4 and IPv6 was used as the underlying and as the tunnel IP version (it means 2x2 series of experiments). Fig. 3 shows the topology and the IP address configuration of the test network used in the IPv4 tunnel over

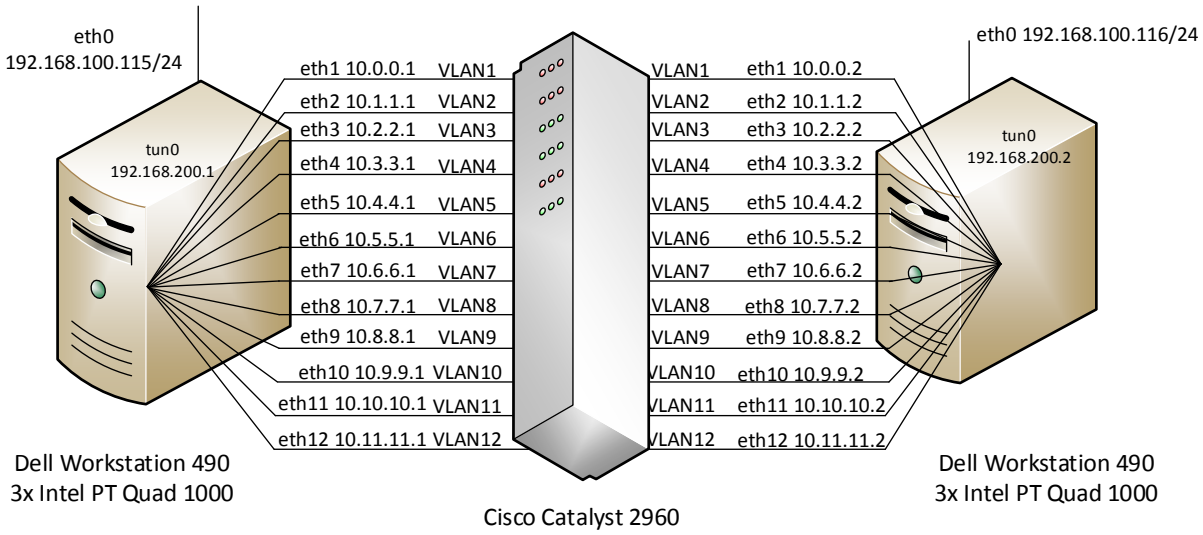


Fig. 3. The topology of the test network (IPv4 tunnel over IPv4 connections)

IPv4 connections tests. The same topology was used for the other three experiments, too. Debian wheezy 7.4 GNU/Linux operating system was installed on both computers.

B. Configuration of the MPT Software

The version of the MPT library can be identified by the name of the file which contains the date in the YYYY-MM-DD format: mpt-lib-2014-03-25.tar.gz was used first. This version of the MPT library contained precompiled 32-bit executables with statically linked libraries thus we did not need to compile it. The contents of the following two configuration files were set as follows. (Their path is relative to the installation directory of MPT.) The beginning of the conf/interface.conf file was:

```
########## Interface Information: ############
12      # The number of the interfaces
65020   # The local cmd port number
1       # Accept remote new connection request
########## Tunnel interface #################
tun0    # INT. NAME, must be the tunnel interface
192.168.200.1/24   #IPv4 address and prefix length
fd00:de:200::1/64  #IPv6 address and prefix length
############## ETH1 interface ################
eth1
10.0.0.1/24
fd00:de:201::1/64
############## ETH2 interface ################
eth2
10.1.1.1/24
fd00:de:202::1/64
```

And it was similar for all the other interfaces, which we do not list to save space. The different types of tunnels were specified in separate connection files. The IPv4 tunnel over IPv4 paths was defined in the conf/connections/IPv4overIPv4.conf file:

```
#### Multipath Connection Information: ####
1       # The number of the connections
########## New Connection ################
TILB    # CONNECTION NAME
3       # SEND(1)/RECEIVE(2) CONNECTION UPDATE
4       # IP VERSION
192.168.200.1   # LOCAL IP
65022           # LOCAL DATA PORT
192.168.200.2   # REMOTE IP
65022           # REMOTE DATA PORT
65020           # REMOTE CMD PORT
12              # NUMBER OF PATHS
0               # NUMBER OF NETWORKS
2               # KEEPALIVE TIME (sec)
5               # DEAD TIMER (sec)
0               # CONNECTION STATUS
0               # AUTH. TYPE
0               # AUTH. KEY
######### Path 0 information: ################
eth1                    # INT. NAME
4                       # IP VERSION
00:15:17:54:d7:30       # LOCAL MAC ADDR
10.0.0.1                # LOCAL IP
00:00:00:00:00:00       # GW MAC ADDR
0.0.0.0                 # GW IP
10.0.0.2                # REMOTE IP
100                     # WEIGHT IN
100                     # WEIGHT OUT
1                       # PATH WINDOW SIZE
0                       # PATH STATUS
######## Path 1 information: ################
eth2                    # INT. NAME
4                       # IP VERSION
00:15:17:54:d7:31       # LOCAL MAC ADDR
10.1.1.1                # LOCAL IP
00:00:00:00:00:00       # GW MAC ADDR
0.0.0.0                 # GW IP
10.1.1.2                # REMOTE IP
100                     # WEIGHT IN
100                     # WEIGHT OUT
1                       # PATH WINDOW SIZE
0                       # PATH STATUS
```

It was also set in the same manner for all the other paths of this connection and for the other connections as well. Note that the configuration files followed strict format, even the comment only lines had to be present. We recommended this to be changed for the commonly used free style configuration files with keyword parsing in [5]. The authors of MPT responded quickly and keyword parsing is provided in the most current version of MPT [2].

V. Experiments and Results

The channel aggregation capability of the MPT library was measured with two different methods: using the industrial de facto standard `iperf`, and file transfer by the `wget` Linux program over the HTTP[1] protocol. These two methods were selected because `iperf` uses UDP and `wget` uses TCP as transport layer protocols. As it was mentioned before, both IPv4 and IPv6 were used as the IP protocol for the tunnel and also as the IP protocol for the underlying channels. In addition to that, both 32-bit and 64-bit versions of the MPT library were tested. It means altogether 2x2x2x2=16 series of measurements, were the number of physical channels were increased from 1 to 12. Thus we performed 16x12=192 different tests. The tests were automated by scripts. Due to space limitations, we cannot include the complete measurement scripts, but the key commands only. The ones below belong to the IPv4 tunnel over IPv4 measurements. The `iperf` command was:

```
iperf -c 192.168.200.1 -t 100 -f M
```

This command performed a 100 seconds long test and printed the throughput in MB/s units. This is called the client side in `iperf` terminology. On the other side, the server was started with the following command line:

```
iperf -s
```

A file of 1GiB size was downloaded using HTTP with the following command line:

```
wget -O /dev/null http://192.168.200.1/1GB
```

This command downloaded the file but did not write it on the hard disk rather disposed it in `/dev/null` so that the disk writing speed would not influence our measurement results. And also the file named 1GB was put on RAM drive at the server computer to eliminate the reading from the hard disk.

The results of our measurements using the 32-bit MPT library are discussed first in details and the 64-bit results are presented later. And within the 32-bit results, we begin with the results of the `iperf` measurements; now they are presented and then discussed.

A. Results of the Iperf Measurements

The results of the `iperf` test are shown in Fig. 4. Whereas two of them (IPv4 over IPv4 and IPv6 over IPv4) are nearly linear in the whole range, the two other ones (IPv4 over IPv6 and IPv6 over IPv6) are nearly linear until 7 NICs and then they show saturation or even a small degradation until the end of the range. Our results suggest that only the version of the underlying IP protocol makes a significant difference in the channel capacity aggregation performance of the MPT library and the version of the encapsulated IP has only a minor influence on it.

When the underlying protocol was IPv4, the throughput was linear up to 12 NICs, which means that the throughput aggregation capability of the MPT library proved to be very good, and we could not reach the limits of MPT library. (These

[1]File transfer by FTP was also tested, but its performance results were so close to that of HTTP that they were finally omitted.

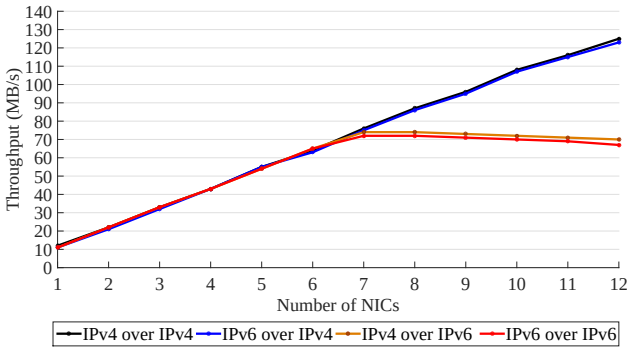


Fig. 4. Throughput results of `iperf` tests

results are very important, because MPT has been tested up to 4 physical channels having only a few Mbps speed before our experiments.)

When the underlying protocol was IPv6, the performance limit of the system was reached at 7 NICs. The maximum values were 74MB/s and 72MB/s in the case of the IPv4 over IPv6 and IPv6 over IPv6 tests, respectively. (The further increase of the number of NICs resulted in some degradation of the throughput, their respective values were 70MB/s and 67MB/s at 12 NICs.) Note that this is the performance of our system composed of the above described hardware and software. We asked ourselves whether it was a built-in limit of the MPT library or it was the performance limit of the hardware that we used for testing?

B. Investigation of the Reason of the IPv6 Performance Limit

1) Checking the CPU utilization: We measured the CPU utilization of the MPT software during the experiments on both the client and on the server during all the 4 series of experiments thus we got 2x4=8 graphs. The CPU usage of the MPT client and of the MPT server was practically the same. The version of the upper IP protocol made no significant difference, therefore we include only two significantly different ones of them. The CPU utilization of the MPT client during the IPv4 over IPv4 measurements is shown in Fig. 5. Even though the time scale is not presented (because no timestamps were logged with the CPU utilization values), the 12 measurements can be easily identified: they are separated by gaps with 0% CPU usage between them. The CPU utilization shows some fluctuations, but its near linear growth can be well observed. It reached the 160-180% interval at 12 NICs. It was checked that the CPU utilization of the `iperf` program was always under 50% thus there was free CPU capacity available from the 400% of the four CPU cores. The CPU utilization of the MPT client during the IPv6 over IPv6 measurements is shown in Fig. 6. It reached 160% at 7 NICs and it fluctuated around 160% for higher number of NICs. There is a visible correspondence between the CPU utilization and the throughput, see Fig. 4.

2) Measurements with faster CPUs: The Intel Xeon 5140 2.33GHz dual core processors of the test computers were replaced by Intel Xeon 5160 3GHz dual core processors. The

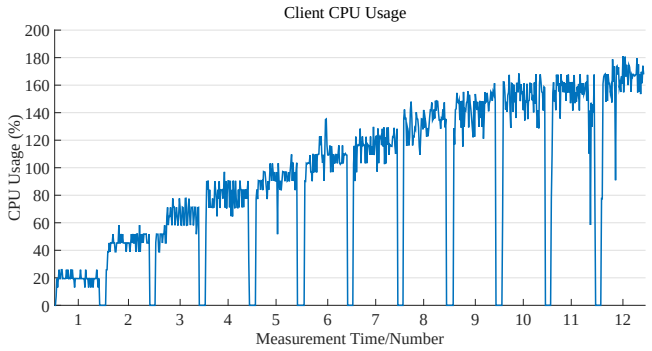


Fig. 5. MPT CPU utilization, IPv4 tunnel over IPv4

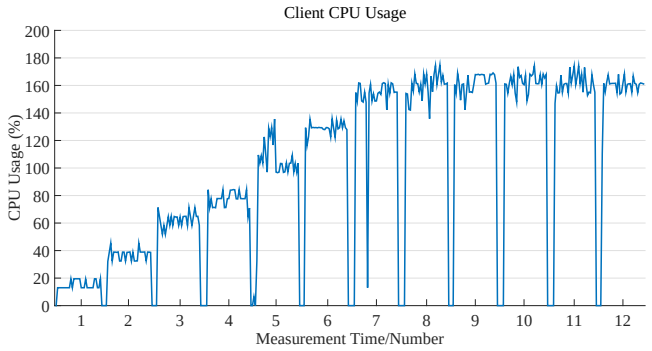


Fig. 6. MPT CPU utilization, IPv6 tunnel over IPv6

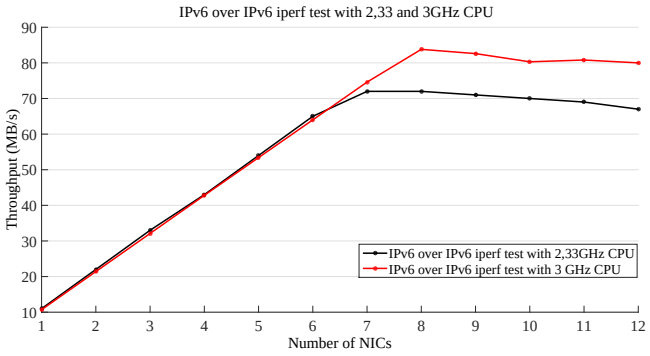


Fig. 7. The throughput results of the `iperf` test of an IPv6 tunnel over IPv6 using 3GHz CPUs

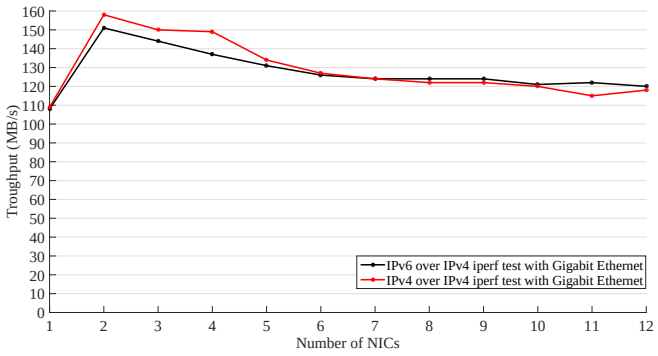


Fig. 8. The throughput results of the `iperf` test of an IPv6 or IPv4 tunnel over IPv4 using Gigabit Ethernet

IPv6 tunnel over IPv6 paths experiments were repeated with the faster CPUSs. Fig. 7 shows the throughput results. It can be observed that the faster CPUs made it possible to fully utilize the capacity of 8 NICs and the degradation started from 9 NICs. This result convinced us that the aggregation capability of MPT does not have a built-in limit, rather it depends on the performance of the CPUs.

However, a question now arises: why could not MPT increase its CPU utilization above 180% while there was still free CPU capacity? The answer is that MPT was written as a serial program and thus it is not able to fully utilize the available processing power of the multiple CPU cores. (The higher than 100% utilization is probably achieved by the overlapping of sending and receiving packets.) We believe that it would be worth improving MPT in this field, because the current trend of the evolution of the CPUs is that the number of cores is increased instead of the clock speed.

After the completion of these measurements, the original Intel Xeon 5140 2.33GHz dual core processors were put back into the test computers and they were used in all the following experiments.

C. *Investigation of the IPv4 Performance Limit*

As it can be seen in Fig. 4, the throughput scaled up nearly linearly up to 12 NICs when the underlying protocol was IPv4. We were interested in the performance limit of the system, but we could not insert more NICs into our Dell computers as they had only 3 PCI Express slots. Therefore, we increased the speed of the NICs to 1Gbps by removing the Cisco switch and interconnecting the two times 12 Ethernet ports of the two computers directly.

The results are shown in Fig. 8. In both tests, the throughput reached its maximum value (of 158MB/s and 151MB/s when the tunnel protocol was IPv4 and IPv6, respectively) at 2 NICs and it degraded for higher number of NICs (down to 118MB/s and 120MB/s at 8 NICs), but it remained still higher than the throughput of a single NIC. This is in correspondence with the values of the CPU utilization in Fig. 9. (The graph actually shows the CPU utilization of the IPv4 over IPv4 case, but the CPU utilization of the IPv6 over IPv4 case looked the same, thus we did not included it.)

D. *Results of the Wget Measurements*

The results are shown in Fig. 10. Unlike with the `iperf`, performance limits can be observed in each graph, and there are also differences between the first two graphs. The HTTP performance of the IPv4 tunnel over IPv4 shows somewhat saturation at 11 and 12 NICs, but the performance is still growing. The HTTP performance of the IPv6 tunnel over IPv4 shows not only saturation but even it definitely degrades at the end of the graph (from 100MB/s at 10 NICs to 90 MB/s at 12 NICs). The HTTP throughput of the IPv4 tunnel over IPv6 reaches its maximum value of 70MB/s at 7 NICs, and it degrades for higher number of NICs (its value is 60MB/s

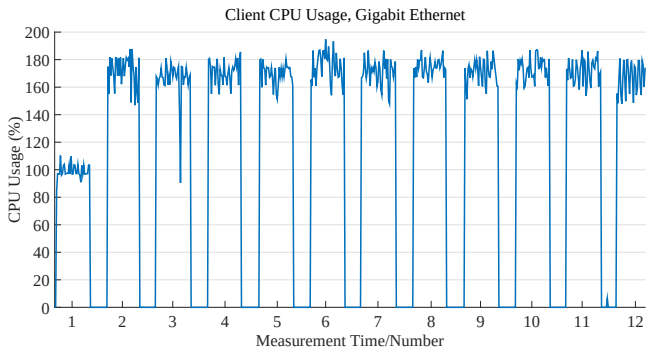


Fig. 9. MPT CPU utilization (from the total of 400%), IPv4 tunnel over IPv4, Gigabit Ethernet

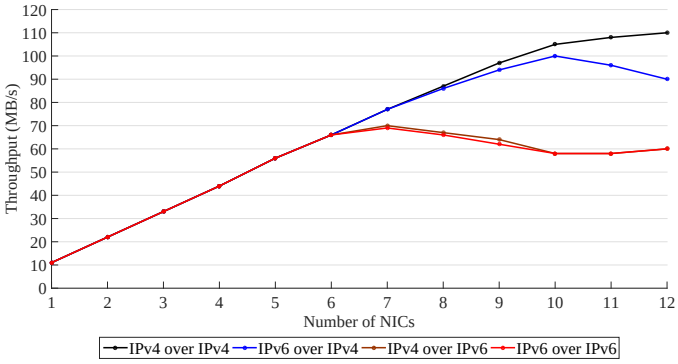


Fig. 10. Throughput results of `wget` tests

at 12 NICs). The HTTP performance of the IPv6 tunnel over IPv6 is nearly exactly the same.

Our HTTP throughput results confirm that the version of the underlying IP protocol makes the major difference in the channel capacity aggregation performance of the MPT library, but they indicate that the version of the encapsulated IP may also have a minor influence on it. However, the results of the `wget` measurements differ from the results of the `iperf` measurements because now we could reach the performance limits of our test system even when the underlying protocol was IPv4. Very likely it is caused by the higher CPU usage of the TCP protocol stack than that of the much simpler UDP. When the underlying protocol was IPv6, we reached the HTTP performance limit of the system at 7 NICs. The further increase of the number of NICs resulted in some degradation of the throughput.

E. Results with the 64-bit MPT Library

The authors of MPT library published the precompiled 64-bit version after the completion of our measurements for [5]. There we mentioned our intention of testing the 64-bit version to see if there is a difference in the performance of the 32-bit and the 64-bit version of the MPT library. We expected that the 64-bit version may more effectively handle the 128 bits long IPv6 addresses. The 64-bit results are presented in the same order as the 32-bit ones: first the `iperf` results and then the `wget` results.

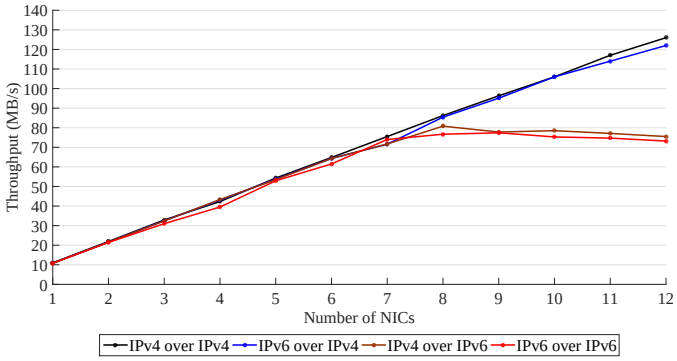


Fig. 11. Throughput results of `iperf` tests (64-bit)

1) Results of the iperf measurements: The results of the 64-bit `iperf` test are shown in Fig. 11. When IPv4 was used as the underlying protocol, the throughput scaled up nearly linearly up to 12 NICs, as we expected. When IPv6 was used as the underlying protocol, the throughput reached its maximum value of at 8 NICs. In the IPv4 over IPv6 case, the maximum value of the throughput was 81MB/s at 8 NICs, which is only by 7MB/s higher than that for the 32-bit case, where maximum value of 74MB/s (see Fig. 4) in throughput has been reached already at 7NICs.

The 64-bit library did not result in the convincing performance improvement that we expected before.

2) Results of the wget measurements: The results of the 64-bit `wget` test are shown in Fig. 12. The graphs are rather similar to graphs of the 32-bit case (see Fig. 10), though the throughput results are somewhat better here. The HTTP perfomance of the IPv4 over IPv4 is linear up to 11 NICs (instead of 10). The HTTP performance of the IPv6 tunnel over IPv4 shows no performance degradation for 11 and 12 NICs, what is an advantage of the 64-bit version over the 32-bit version of the MPT library. The HTTP performance of the IPv4 tunnel over IPv6 reaches its maximum value at 7 NICs. The maximum place of the throughput result curve of the 32-bit test is the same (Fig. 10), but here the maximum value is a little bit higher: 74.4MB/s instead of 70MB/s. And the linear degradation here is bit better than the degradation was in the 32-bit case. The HTTP performance of the IPv6 tunnel over IPv6 is also somewhat better, but rather similar to that of the 32-bit case.

Though the 64-bit version of the MPT library did not fulfill our performance expectations, but the 64-bit results are definitely never worse than those of the 32-bit version, and in many cases the 64-bit version brings some slight performance increase.

VI. Directions of our Future Research

So far, we have tested the performance and throughput aggregation capability of the MPT library in itself. We also plan to compare them with that of the standard MPTCP.

As the most important advantage of MPT over MPTCP is that MPT uses UDP/IP and therefore it is much suitable for use with real-time applications because of the elimination of

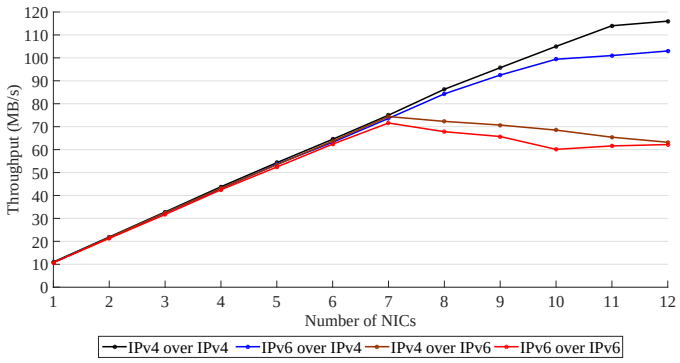


Fig. 12. Throughput results of `wget tests` (64-bit)

TCP retransmissions, we also plan to test it with real-time applications.

We also intend to test MPT as a tunneling tool. MPT seems to be a universal tunnel software in the context of IPv6 transition since it can be used as either of an IPv4 or an IPv6 tunnel over either of IPv4 or IPv6 connections.

VII. Conclusion

The throughput aggregation performance of the MPT network layer multipath communication library was examined up to twelve 100Mbps link layer connections. Measurements were taken with both `iperf` (over UDP) and `wget` (over TCP) using both 32-bit and 64-bit MPT libraries.

As for the 32-bit MPT library and `iperf` measurements, when the underlying protocol was IPv4, the throughput scaled up linearly up to 12 NICs (exceeding 120MB/s) regardless of the version of the encapsulated IP (IPv4 or IPv6). When the underlying protocol was IPv6, the throughput scaled up linearly up to 7 NICs (exceeding 70MB/s) regardless of the version of the encapsulated IP, but it could not increase more for higher number of NICs rather it showed a small degradation.

It was proved that the above performance limit depends on the computing power of the CPUs and it is not a fixed built in feature of the MPT library.

MPT was also tested with 12 Gigabit Ethernet connections to find the performance limit of our system when the underlying protocol was IPv4. It was reached at two NICs having the values of 158MB/s and 151MB/s when the tunnel protocol was IPv4 and IPv6, respectively.

As for the 32-bit MPT library and `wget` measurements, the results were similar to those of the `iperf` measurements with the exception, that we could reach the performance limit of the system even when the underlying protocol was IPv4 due to the higher CPU usage of the TCP protocol stack than that of the much simpler UDP.

As for the measurements with the 64-bit MPT library (using both `iperf` and `wget`), the results were close to the results of the measurements with the 32-bit MPT library, producing only usually a little performance benefit depending on the given test but the 64-bit results were never worse than the 32-bit ones.

We conclude the MPT network layer multipath communication library proved to be a good tool for the aggregation of the capacity of several high speed channels.

References

[1] B. Almási, A. Harman, "An overview of the multipath communication technologies", In *Proc. Conf. on Advances in Wireless Sensor Networks 2013 (AWSN 2013)*, Debrecen University Press, Debrecen, Hungary, ISBN: 978-963-318-356-4, pp. 7–11, 2013.

[2] B. Almási, "MPT library", precompiled version can be downloaded from: `http://irh.inf.unideb.hu/user/almasi/mpt/`

[3] B. Almási, Sz. Szilágyi, "Throughput performance analysis of the multipath communication library MPT", In *Proc. 36th Int. Conf. on Telecommunications and Signal Processing (TSP 2013)*, Rome, Italy, Jul. 2-4, 2013, pp. 86–90. DOI: 10.1109/TSP.2013.6613897

[4] B. Almási, Sz. Szilágyi, "Multipath ftp and stream transmission analysis using the MPT software environment", *Int. J. of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 11, pp. 4267–4272. Nov. 2013.

[5] G. Lencse, Á. Kovács, "Testing the channel aggregation capability of the MPT multipath communication library", In *Proc. World Symposium on Computer Networks and Information Security 2014 (WSCNIS 2014)*, Hammamet, Tunisia, Jun. 13-15, 2014, ISBN: 978-9938-9511-9-6, Paper ID: 1569946547.

[6] A. Ford, C. Raiciu, M. Handley and O. Bonaventure, "TCP extensions for multipath operation with multiple addresses", IETF, Jan. 2013, RFC 6824.

[7] C. Paasch, G. Detal, S. Barré, F. Duchêne, O. Bonaventure, "The fastest TCP connection with multipath TCP", ICTEAM, UCLouvain, Louvain-la-Neuve, Belgium, available: `http://multipath-tcp.org/pmwiki.php?n=Main.50Gbps`

[8] R. Khalili, N. Gast, M. Popovic, J.-Y. Le Boudec, "MPTCP is not Pareto-optimal: performance issues and a possible solution", *IEEE/ACM Trans. Networking*, vol. 21, no. 5, pp. 1651–1665, Oct. 2013, DOI: 10.1109/TNET.2013.2274462

[9] B. Almási, "Multipath communication – a new basis for the future Internet cognitive infocommunication", In *Proc. CogInfoCom 2013 Conf.*, Budapest, Hungary, Dec. 2-5, 2013, pp. 201–204, DOI: 10.1109/CogInfoCom.2013.6719241

[10] B. Almási, "A simple solution for wireless network layer roaming problems", *Carpathian Journal of Electronic and Computer Engineering*, vol. 5, no. 1, pp. 5–8, 2012.

[11] B. Almási, "A solution for changing the communication interfaces between WiFi and 3G without packet loss", In *Proc. 37th Int. Conf. on Telecommunications and Signal Processing (TSP 2014)*, Berlin, Germany, Jul. 1-3, 2014, pp. 73–77.

[12] Z. Gál, B. Almási, T. Dabóczi, R. Vida, S. Oniga, S. Baran, and I. Farkas, "Internet of things: application areas and research results of the FIRST project", *Infocommunications Journal*, vol. 6, no. 3, pp. 37–44, Sep. 2014.

[13] B. Almási, Sz. Szilágyi, "Investigating the throughput performance of the MPT multipath communication library in IPv4 and IPv6", *Journal of Applied Research and Technology*, to be published

[14] H. Newman, A. Barczyk, M. Bredel, "OLiMPS: openflow link-layer multipath switching", *DOE ASCR NGN PIs Meeting*, Rockville, Sept. 16-17, 2014, slides available: `http://www.orau.gov/ngnspi2014/presentations/newman_h.pdf`

[15] A. Barczyk, M, Bredel, H. Newman, R. van der Pol, "Openflow-based multipath networks in the WAN", In *Proc. TERENA Networking Conference (TNC 2013)*, Maastricht, Netherlands, Jun. 3-6, 2013, available: `https://tnc2013.terena.org/getfile/192`

[16] Multiple Interfaces Working Group, "Mif status pages", available: `https://tools.ietf.org/wg/mif/`

[17] M. Blanchet, P. Seite, "Multiple interfaces and provisioning domains problem statement", IETF, Nov. 2011, RFC 6418.

[18] S. Gundavelli (Ed.), K. Leung, V. Devarapalli, K. Chowdhury, B. Patil, "Proxy mobile IPv6", IETF, Aug. 2008, RFC 5213.

[19] CJ. Bernardos (Ed.), "Proxy mobile IPv6 extensions to support flow mobility", IETF, Draft, available: `https://tools.ietf.org/html/draft-ietf-netext-pmipv6-flowmob-12`

[20] B. Almási, "MPT library user guide", can be downloaded from: `http://irh.inf.unideb.hu/user/almasi/mpt/`

Gábor Lencse received his MSc in electrical engineering and computer systems at the Technical University of Budapest in 1994, and his PhD in 2001.

He has been working for the Department of Telecommunications, Széchenyi István University in Győr since 1997. He teaches Computer networks, Computer architectures, IP-based telecommunication systems and the Linux operating system. Now, he is an Associate Professor. He is responsible for the specialization of the information and communication technology of the BSc level electrical engineering education. He is a founding member of the Multidisciplinary Doctoral School of Engineering Sciences, Széchenyi István University. The area of his research includes discrete-event simulation methodology, performance analysis of computer networks and IPv6 transition technologies. Dr. Lencse has been working part time for the Department of Networked Systems and Services, Budapest University of Technology and Economics (the former Technical University of Budapest) since 2005. There he teaches Computer architectures and Computer networks.

Ákos Kovács received MSc in electrical engineering with specialization in infocommunication systems and services at the Széchenyi István University in 2013.

He started working as laboratory engineer at the Department of Telecommunications in 2008. During this time he got familiar with high-end computer systems, virtualization and cloud computing. He also has high skills in the field of computer networks, and networking security. He teaches computer networks and virtualization technology in BSc and holds practical lessons in MSc in the field of IP-based telecommunication.

Multi-Radio Mobile Device: Evaluation of Hybrid Node Between WiFi and LTE Networks

Pavel Masek, Krystof Zeman, Dalibor Uhlir, Jan Masek, Chris Bougiouklis, and Jiri Hosek

***Abstract*—With the ubiquitous wireless network coverage, Machine-Type Communications (MTC) is emerging to enable data transfers using devices / sensors without need for human interaction. In this paper we introduce a comprehensive simulation scenario for modeling and analysis heterogeneous MTC. We demonstrate the most expected scenario of MTC communication using the IEEE 802.11 standard for direct communication between sensors and for transmitting data between individual sensor and Machine-Type Communication Gateway (MTCG). The MTCG represents the hybrid node serving as a bridge between two heterogeneous networks (WiFi and LTE). Following the idea of hybrid node, two active interfaces must be implemented on this node together with mechanism for handling the incoming traffic (from WiFi network) to LTE network. As a simulation tool, the Network Simulator 3 (NS-3) with implemented LTE/EPC Network Simulator (LENA) framework was used. The major contribution of this paper therefore lies in the implementation of logic for interconnection of two heterogeneous networks in simulation environment NS-3.**

***Keywords*—LTE, MTC Communication, MTCG, Network Simulator 3, WiFi.**

I. Introduction

Machine-Type Communication (MTC) represents the way how to enable the connectivity between several (from tens to hundreds) nodes (sensors or actuators) without or with minimal human interaction e.g. Internet of Things (IoT) or smart power grids [1]. Following the information given in [2], [3], the amount of mobile data traffic is predicted to increase by around six times in the period 2014 - 2019. The data traffic is distinguished into two main categories: Human-to-Human (H2H) and Machine-to-Machine (M2M) communication. In comparison with the traditional conception of data traffic represented by H2H (services as voice, web streaming etc.), M2M comes with different requirements on a communication system [4] where the M2M applications should have minimal impact on existing H2H services [5]. The key differences between both communication types are shown in Table I.

The key idea of the M2M communication network is to connect a server with millions of devices deployed worldwide (interacting with other sensors, different environments and people). With the rapid development of cellular networks, M2M communication via the Long Term Evolution (LTE) network is expected to play a significant role in M2M scenarios. Today, the cellular networks represent the common data access to public network (Internet); as a consequence they are under pressure trying to handle unprecedented data flows from the side of mobile devices. The dramatic increase of transmitted data via cellular networks is a burning question for telecommunication operators with the limited resources of radio spectrum [6]. The complex scenario of M2M architecture is shown in Fig. 1.

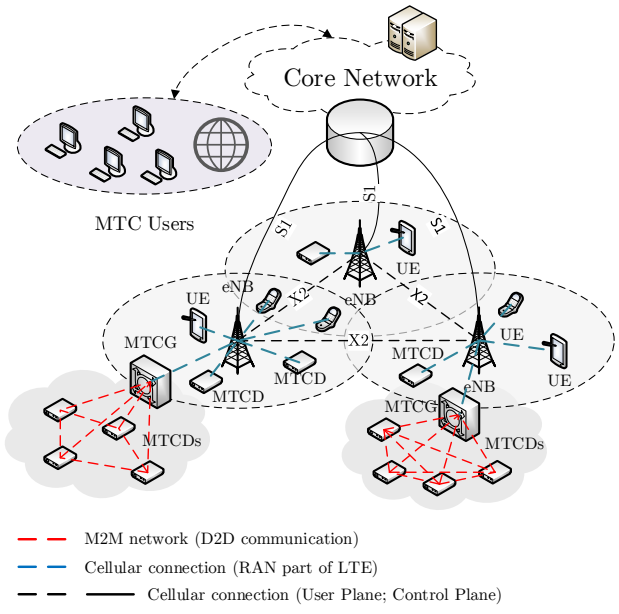


Fig. 1. LTE networks with M2M communications

The depicted architecture considers two different ways for managing connection of M2M devices to the core part of LTE network [8]:

- Cellular connectivity: connection through access network to core networks where each single device has its own Subscriber Identity Module (SIM) card for cellular connectivity.

P. Masek, K. Zeman, D. Uhlir, and J. Hosek are with Department of Telecommunications, Brno University of Technology, Brno, Czech Republic (e-mails: xmasek12@phd.feec.vutbr.cz, xzeman43@stud.feec.vutbr.cz, xuhlir15@stud.feec.vutbr.cz, hosek@feec.vutbr.cz).

J. Masek is with Institute of Structural Mechanics, Brno University of Technology, Brno, Czech Republic (e-mail: masekj3@study.fce.vutbr.cz).

Ch. Bougiouklis is with Technological Educational Institute of Crete, Greece (e-mail: chrisbougiouklis@hotmail.com).

Manuscript received May 6, 2015; revised May 15, 2015. Research described in this paper was financed by the National Sustainability Program under grant LO1401 and by the project CZ.1.07/2.3.00/30.0005 of Brno University of Technology. For the research, infrastructure of the SIX Center was used.

- M2M networks: M2M devices may create M2M area networks using short range technologies represented by the standards IEEE 802.15.6, IEEE 802.15.4(e), or IEEE 802.11. These M2M area networks can be then connected to the core networks via M2M gateways [9], [10], [11].

As a possible way how to deal with the overloading of Random Access Network (RAN) of LTE network, the offloading techniques can be used; offloading mechanisms refer to using alternative network infrastructure for transmitting data originally targeted for cellular network when this network becomes overloaded [7] [1]. Depending on delay (content delivery time) it is possible to divide offloading techniques into two categories: *nondelayed offloading* and *delayed offloading* [7].

In this paper we address the specific type of delayed offloading where the Machine Type Communication Gateway (MTCG) act as a hybrid node which interconnects two different networks (in case of this paper, WiFi and LTE network are considered as heterogeneous networks). The attention is also paid to the implementation of Quality of Service (QoS) for H2H and M2M communication where the high prioritized traffic is represented by the H2H communication (e.g. Voice over IP (VoIP)); QoS is implemented on MTCG node. Furthermore, QoS requirements of M2M services depend on the MTC service features: group-based communication, mobility, time-controlled / time-tolerant, amount of transmitted data, power consumption [12].

[1] Current visions from analytical claim that by 2019, 54 percent of total mobile data traffic will be offloaded over WiFi networks.

We performed extensive simulations to evaluate the role of MTCG node in LTE architecture with M2M communication. For modeling WiFi and LTE networks, data traffic and logic of MTCG node, the simulation tool Network Simulator 3 (NS-3) [13] with the framework LTE / EPC Network Simulator (LENA) [14] was used.

The rest of the paper is organized as follows. Section II presents the description of MTCD-Related communications in LTE network. Section III deals with the selected simulation environment NS-3 together with the LENA framework. In section IV the description of created simulation scenario is given. Section V presents the obtained results and finally, in section VI we draw a conclusion with our future plans in this research area.

II. LTE Network and M2M Communication

The Current RAN for LTE network consists of eNodeB (eNB) that provides the user plane and control plane protocol stack for the User Equipment (UE). LTE represents the fully distributed radio access network architecture, where the eNB can be interconnected with other eNBs by the X2 interface. The eNBs are then connected to the core part of LTE network through the S1 interface, see Fig. 1. Each eNB includes layers below that implement the functionality of user plane, header compression and encryption [12]:

- PHY: Physical layer,
- MAC: Medium Access Control layer,
- RLC: Radio Link Control layer,
- PDCP: Packet Data Control Protocol layer.

	Machine-to-Machine (M2M)	Human-to-Human (H2H)
Traffic Direction	Uplink data; data received from sensors. For a specific type of the applications, the symmetric uplink and downlink is needed to fulfill the requirements for the dynamic interaction.	Downlink data; although during last few years the amount of uploaded data is growing fast, in case of H2H, download still represents the main part of data traffic.
Message Size	Size of data from sensors is in general very small (e.g. data size of Wireless M-BUS data unit is usually max. 50 B).	Using multimedia and realtime applications, the size of data units is several times higher in comparison with the M2M.
Access Delay	For the dynamic interaction between sensors and actuators, delays should be very short.	In case of H2H communication, longer access delays are usually tolerated.
Transmission Periodicity	The range of transmitting period can be from units of seconds (e.g. alarm systems) up to tens of minutes (e.g. energy meters).	Nature of human based traffic is mostly random and bursty. Therefore, the often sending of control information is required (to ensure QoS).
Mobility	For the main group of sensors, mobility does not represent a big issue (sensors are mostly located at the stable position).	For humans, mobility management represents a key requirement for ensuring seamless connectivity and roaming.
Data Importance	Some of the M2M sensors can transmit critical data (e.g. status of alarm system). Following this fact, M2M data requires high priority.	There are no big differences between users. The differences could be found between the applications for individual users (with respect to QoS and QoE).
Amount of devices	Hundreds or thousands of devices connected via one access point to the network.	Typically tens of devices which are connected via access point to the network.
Lifetime; Energy Efficiency	Using specific energy profiles, devices are able to operate for years of decades without human maintenance.	In case of devices used by humans, it is common to recharge batteries in a daily manner (smartphones, laptops).

TABLE I
Differences between H2H and M2M communication [8]

Following the fact that the current 3G cellular networks are designed only for H2H communications, the introduction of M2M communications introduce the new requirements on LTE networks; the network architecture needs to be improved to fulfill M2M services without sacrificing the current H2H applications.

In this section, the attention will be given to a description of types of M2M communication (especially a description of connection of the MTCD and MTCG nodes to LTE network will be described).

A. Machine Type Communication

To enable M2M communication in cellular networks (3G / 4G), the two new types of nodes Machine Type Communication Devices (*MTCD*) and Machine Type Communication Gateway (*MTCG*) were introduced. The MTCD represents the UE which is supposed to work as a sensor which communicates through the cellular network with the remote MTC node (e.g. database server) or (and) with other MTCDs in range. As was proven in [15], the high number of MTCDs connected at the same time to one eNB may cause overloading of this network entity. Therefore, the cellular network requires an MTCG node to facilitate communications among a great number of MTCDs. The MTCG will enable the intelligent way how to manage power consumption of MTCDs and provide an efficient path for communication between MTCDs without the need of connection to the LTE network. Three different M2M communication methods were introduced during last few years, see Fig. 2 [12]. These methods are described (in the text) below.

1) Direct Transmission Between MTCD and eNB: The first method is similar to the classic UE where the MTCD is able to establish the direct connection to the eNB; therefore similarities between eNB-to-UE and eNB-to-MTCD exist. On the other hand, the MTCDs are represented in a large amount of sensors / UEs; in certain time period, intensive competition for radio resources may occur. Therefore, the additional efforts have to be covered by the telecommunication operators to solve the problems, when the large number of MTCDs communicate with the eNB directly [12].

2) Multihop Transmission using MTCG: With respect to mitigate or eliminate negative effect of M2M communication on H2H communication in cellular network, the MTCG node can be deployed as a hybrid node, where all MTCDs are connected to the eNB indirectly using the MTCG node as a gateway. The eNB-to-MTCG connection is based on the 3GPP (Third Generation Partnership Project) LTE specifications. The MTCG-to-MTCDs and MTCD-to-MTCD communications can be established via 3GPP LTE specifications or via the non-3GPP communication technologies such as IEEE 802.11, IEEE 802.15.x [12], [16].

3) P2P Transmission Between MTCDs: An MTCD may communicate in local area with other MTCDs and with the eNB. Compared to other non-3GPP local connectivity solutions (IEEE 802.11, IEEE 802.15.x), direct communication between MTCDs is done by cellular network which can broadcast data within a much wider coverage area. For service discovery, the MTCDs do not have to scan all the time for the available access point (APs) as in the case of standard IEEE 802.11 [12], [16], [17].

III. LENA Framework in NS-3

During the last years, several network simulation platforms have been developed as a tool available for networking research: OPNET Modeler [19], OMNET++ [18], NS-2 [13], NS-3 [13]. Based on the fact that this paper deals with the M2M communication in LTE network, the simulation environment NS-3 together with the LENA framework [21] were used. In our work, we used NS-3 in version 3.21 together with the LENA framework in version 8. Using the LENA inside NS-3 provides for us the way for design and performance evaluation of Heterogeneous Networks (HetNets). Fig. 3 shows the implementation of the end-to-end LTE-EPC data plane protocol stack of LENA framework. The biggest change in comparison with the standard implementation of data plane protocol stack of LTE is the merge of the Serving Gateway (SGW) and PDN Gateway (PGW) functionality within one single (SGW)/(PGW) node in NS-3. This change causes that there is no need to have S5 and S8 interfaces which are specified by 3GPP. The S1-U protocol stack and the LTE radio protocol stack specified by 3GPP, are also described in Fig. 3.

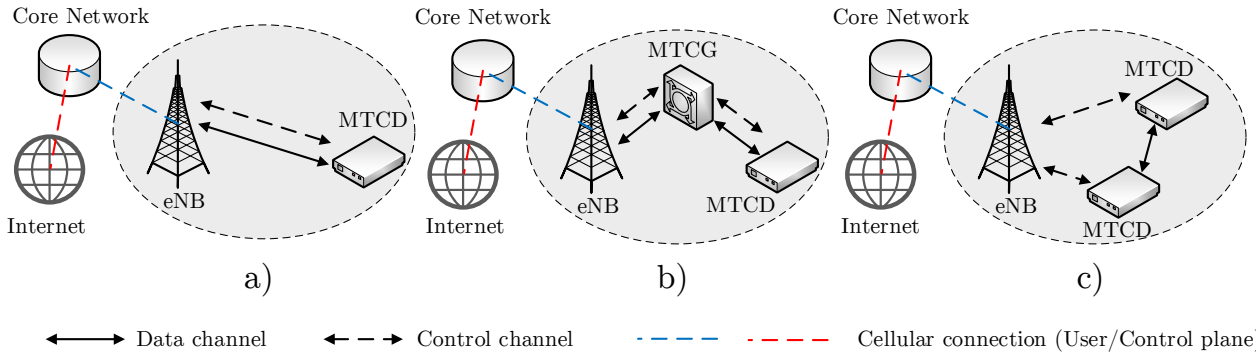


Fig. 2. MTCD-related transmissions: a) Direct transmission, b) Multihop transmission, c) Peer-to-peer transmission

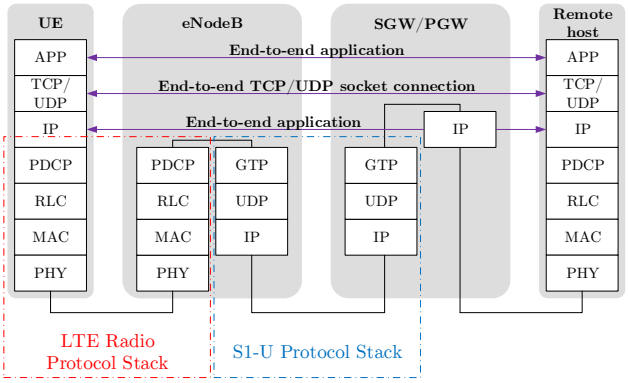


Fig. 3. LTE-ECP data plane protocol stack in LENA framework

IV. Model of M2M Communication in LTE Network

As described in Section II, the created scenario includes the MTCD nodes together with the MTCG node which enables the interconnection between the local network and the public network (represented by the remote host which is accessible through the LTE network). The local side of the implemented scenario is represented by sensors / UEs using IEEE 802.11g, IEEE 802.11ah [23], [24] and Wireless M-BUS [22] which represent the most preferred technologies for M2M. The data is sent through the hybrid node (MTCG) to a remote host which is accessible via the LTE network. The overall structure of the created scenario is depicted in Fig. 4.

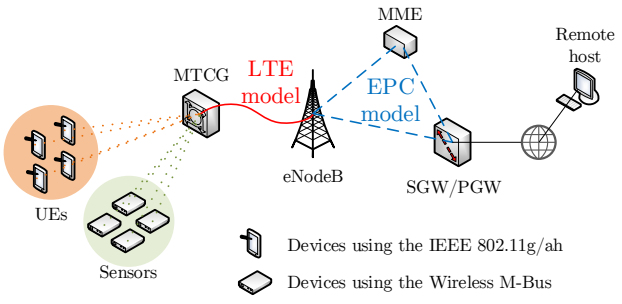


Fig. 4. M2M communication scenario in NS-3

A. Parameters of Simulation Scenario

The key parameters of created simulation model are shown in Table II (a list of the parameters of created LTE network using the LENA framework).

UEs were created as wireless nodes using the IEEE 802.11 g and IEEE 802.11 ah for connection to MTCG node. The sensors implemented the Wireless M-BUS communication protocol (868 MHz) where the sensors were set to Mode T1 (one-directional communication) and MTCG was set to the Mode T2 (bi-directional communication).

TABLE II
Parameters of LTE network in NS-3

Parameter	Setting
Cell Layout	1 eNodeB, 1 sector
Duplex Format	LTE-FDD
Maximum transmit power	30 dBm
System Bandwith	3 MHz (∼ 15 PRBs)
Scheduler	Pf Df Mac Scheduler
Path loss model	Friis Spectrum Propagation Loss Model
Direction	Download
eNB antenna model	Isotropic Antenna Model
Frequency Reuse Factor	1

B. IP Address Scheme

The address scheme for two groups of nodes is depicted in Fig. 5. For the WiFi nodes (IEEE 802.11 g/ah) the address space 10.3.0.0 with prefix 24 was used. In case of Wireless M-BUS nodes, unique addressing scheme is implemented following the [25]. The address of each WM-BUS node is represented by the serial number of sensor.

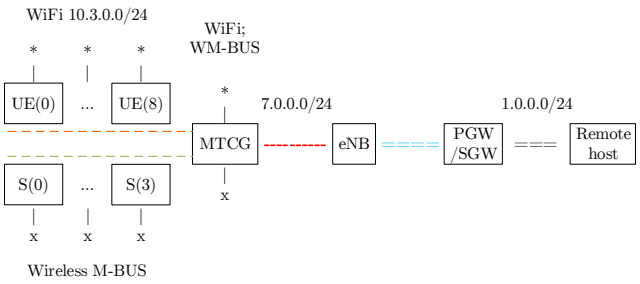


Fig. 5. Address Scheme of created model

The transmission of data from sensors is performed as a broadcast communication when only the MTCG node (in T2 mode) can receive the information from sensors. Data from MTCG node goes through the core part of LTE network (7.0.0.0 / 24) to the destination node (remote host; 1.0.0.0 / 24).

C. Parameters of Data Traffic

Data traffic is generated independently by a group of UEs (H2H) and sensors (M2M). Data traffic from UEs represents the voice service defined as follows [26]: UDP transport protocol; packet size 160 B; Maximal Transfer Unit (MTU) 1500 B. Traffic from sensors was generated with these attributes: WM-BUS communication protocol (do not follow the TCP/IP reference model), packet size 50 B, transmission interval 30 seconds. Both groups of devices (UEs and sensors) generate traffic during the whole simulation; simulation time was set to 10 minutes.

V. In-depth Results Discussion

From the implementation point of view, the two active interfaces on one node in NS-3 represent a challenging task. This task is going to be more complex when one

of the implemented interface does not follow the requirements given by TCP/IP reference model; this is an example of Wireless M-BUS, which was implemented from the scratch in NS-3 as a specific representative of M2M data traffic. Merging the data traffic (the part of static routing for WiFi nodes which have been configured with the static route to the MTCG node) is briefly described in a part of the source code below.

```
Ipv4StaticRoutingHelper ipv4RoutingHelper;
...
for (uint32_t i=0; i<wifiStaNodes.GetN (); ++i)
    {
      Ptr<Node> wifiNode = wifiStaNodes.Get(i);
      Ptr<Ipv4StaticRouting> wifiStaticRouting =
          ipv4RoutingHelper.GetStaticRouting (
          wifiNode->GetObject<Ipv4> ());
      wifiStaticRouting->SetDefaultRoute (
          WifiInterfaces.GetAddress (0), 1);
    }
Ptr<Node> MTCGNode = MTCG.Get (0);
...
Ptr<Ipv4StaticRouting> ueStaticRouting =
    ipv4RoutingHelper.GetStaticRouting (MTCGNode
    ->GetObject<Ipv4> ());
ueStaticRouting->SetDefaultRoute (epcHelper->
    GetUeDefaultGatewayAddress (), 1);
```

The example of one data frame received from sensors using the Wireless M-BUS on MTCG is shown below [2].

```
DATA RECEIVED: 28442
    B414452127002027Ab400000004FB2C0b00000001FD
490b02FD590b00022B0b0004030b00000
-------------------------------------------------
MbusApp:Receive. Power: -41.2005 Size: 81. Time:
    +158639932504.0ns
Base address: 0. Sender address: 72783391635506
AccReceived: 180
IDReceived: 70125244
MeterTypeReceived: 02
FrequencyReceived: 11
VoltageReceived: 11
CurrentReceived: 11
PowerReceived: 11
WorkReceived: 0
-------------------------------------------------
MbusApp::Sendtime: +174539062500.0ns. Packet size
    :81.
Acc=169, inc=12
Data=28442
    B414452127002027Ab500000004FB2C0c00000001FD
490c02FD590c00022B0c0004030c000000
```

A. Analysis of Data Traffic

To evaluate the correct behavior of created traffic from UEs and sensors, the trace files were created during the simulation in a compatible format for network protocol analyzer Wireshark [31]. As depicted in Fig. 3, the UDP is used as a transport protocol in local network. On the other hand, in LTE network the data is encapsulated via the GPRS Tunneling Protocol (GTP) [32] which is used over S1-U, X2, S4, S5 and S8 interfaces of the Evolved Packet System (EPS); note that the S5 and S8 interfaces are not implemented in LENA framework yet. GTP is an important IP/UDP based protocol used in Global System for Mobile Communications (GSM), Universal Mobile Telecommunication System (UMTS) and LTE core networks.

The correct handling with the data traffic is depicted for the UE(0) in Fig. 6 and Fig. 7.

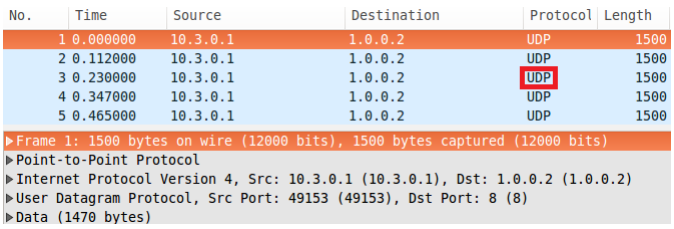


Fig. 6. Captured data traffic in local network; UDP protocol

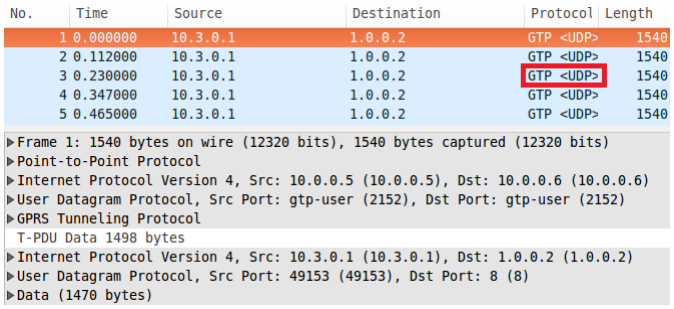


Fig. 7. Captured data traffic in LTE network; GTP protocol

B. Enabling QoS for H2H Traffic in Created Model

The support of QoS for VoIP data traffic (originated from WiFi nodes (UEs)) was implemented on MTCG node. The situation with and without the implemented QoS features is depicted in Fig. 8.

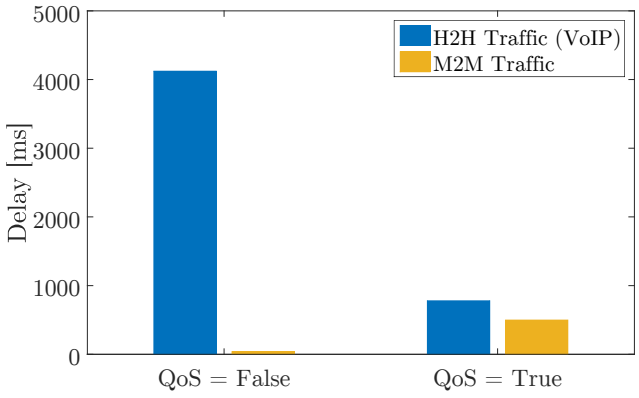


Fig. 8. Implemented QoS features on MTCG node

The values of delay were originally for H2H traffic (VoIP) 4123 ms and 40 ms for M2M traffic. It is clearly visible that without the implemented QoS, the VoIP services can not be used with respect to fulfill users expectation. Therefore, the QoS was implemented on MTCG node and the delay decreased to 780 ms (this means an improvement of 81.08 % in comparison with the original delay for VoIP).

[2]The specifications of sensors manufacturers Bonega [27], WepTech [28], Pikkerton [29], and ZPA [30] were implemented and evaluated.

VI. Conclusion

M2M communications represent an emerging technology which illustrates the principles of the IoT. Therefore, it has gained an increasing attention in LTE / LTE-A cellular network design. In this paper, we give the overview of the required network architectural improvements with the description of the various transmission schemes / types for MTCDs. We chose the multihop transmission from described transmission schemes, see Fig. 2. This type can be represented by the MTCG node which acts as a hybrid node between several heterogeneous networks. In this paper we implemented three types of networks: WiFi, Wireless M-BUS and LTE. Between these networks the MTCG node was deployed in a role of the bridge where the incoming data traffic is routed towards the destination node (e.g. a remote server located in Internet).

The implementation was done using the simulation environment NS-3 with the LENA framework, see section III. The simulation results, see section V, confirm the correct handling of data traffic with respect to meet the QoS and QoE requirements for the H2H traffic in case when the M2M services are deployed in parallel with the H2H. Although we have achieved an improvement of delay of 84 % (from 4123 ms to 780 ms) for VoIP services, it is evident that further investigation of aggregation scheme on MTCG node is still needed.

Acknowledgment

The described research was supported by the National Sustainability Program under grant LO1401 and by the project CZ.1.07/2.3.00/30.0005 of Brno University of Technology. For the research, infrastructure of the SIX Center was used.

References

[1] F. Ghavimi and Ch. Hsiao-Hwa, *M2M Communications in 3GPP LTE/LTE-A Networks: Architectures, Service Requirements, Challenges and Applications,* Communications Surveys & Tutorials, IEEE , vol.PP, no.99, pp.1,1 doi: 10.1109/COMST.2014.2361626

[2] Cisco, *Visual Networking Index: Global Mobile Data Traffic Forecast Update 20142019 White Paper.* Cisco [online]. Available from: http://bit.ly/1b13ryX

[3] Ericsson, *Annual Report 2014.* [online]. Available from: http://bit.ly/1Gc9lId

[4] 3GPP TS 22.368 v1.0, *Service requirements for Machine-Type Communications (MTC) Stage 1 (Release 10),* Mar. 2010.

[5] 3GPP TR 23.888 v0.5.1, *System Improvements for Machine-Type Communications (Release 10),* July 2010.

[6] D.H. Hagos and R. Kapitza, *Study on performance-centric offload strategies for LTE networks,* Wireless and Mobile Networking Conference (WMNC), 2013 6th Joint IFIP , p.1-10, 23-25 April 2013. DOI: 10.1109/WMNC.2013.6548999

[7] F. Rebecchi, D. Amorim, V. Conan, A. Passarella, R. Bruno and M. Conti, *Data Offloading Techniques in Cellular Networks: A Survey, IEEE Communications Surveys & Tutorials,* vol.PP, no.99,.

[8] A. Laya, L. Alonso and J. Alonso-Zarate, *Is the Random Access Channel of LTE and LTE-A Suitable for M2M Communications? A Survey of Alternatives,* Communications Surveys & Tutorials, IEEE , vol.16, no.1, pp.4,16, First Quarter 2014 doi: 10.1109/SURV.2013.111313.00244

[9] P. Masek, J. Hosek, D. Kovac and F. Kropfl, *M2M Gateway: The Centrepiece of Future Home.* In 2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT). St. Petersburg, Russia: 2014. s. 286-293. ISBN: 978-1-4799-5290- 8.

[10] J. Hosek, P. Masek, D. Kovac, M. Ries and F. Kropfl, *IP Home Gateway as Universal Multi- Purpose Enabler for Smart Home Services.* Elektrotechnik und Informationstechnik ÃŰVE - Verbandszeitschrift, 2014, roÄD. 131, ÄD. 5, s. 1-6. ISSN: 0932- 383X.

[11] J. Hosek, P. Masek, D. Kovac, M. Ries and F. Kropfl, *Universal Smart Energy Communication Platform.* In 2014 International Conference on Intelligent Green Building and Smart Grid (IGBSG). 1. Taipei, Taiwan: IEEE, 2014. s. 1-4. ISBN: 9781467361217.

[12] K. Zheng, H. Fanglong, W. Wenbo, W. Xiang and M. Dohler, *Radio resource allocation in LTE-advanced cellular networks with M2M communications,* Communications Magazine, IEEE , vol.50, no.7, pp.184,192, July 2012 doi: 10.1109/MCOM.2012.6231296

[13] Network Simulator 3: *Discrete-event network simulator.* NSNAM [online]. Available from: www.nsnam.org

[14] LENA: *LTE-EPC Network simulAtor.* CTTC [online]. Available from: http://networks.cttc.es/mobile-networks/software-tools/lena/

[15] P. Masek, J. Hosek and M. Dubrava, *Influence of M2M Communication on LTE Networks.* In Sbornik prispevku studentske konference Zvule 2014. 1. 2014. s. 53-56. ISBN: 978-80-214-5005-9.

[16] P. Marsch et al., *Future mobile communication networks: Challenges in the design and operation,* IEEE Veh. Technol. Mag., vol. 7, no. 1, pp. 16âĂŞ23, Mar. 2012.

[17] A. Aijaz, H. Aghvami and M. Amani, *A survey on mobile data offloading: technical and business perspectives,* Wireless Communications, IEEE, vol.20, no.2, pp.104,112, April 2013 doi: 10.1109/MWC.2013.6507401

[18] OMNeT++: *Discrete Event Simulator.* [online]. Available from: www.omnetpp.org

[19] Riverbed Modeler: *Network Performance Management.* Riverbed [online]. Available from: http://bit.ly/1Mpu9OC

[20] Network Simulator 2: *Discrete event simulator targeted at networking research.* [online]. Available from: http://www.isi.edu/nsnam/ns/

[21] LENA: *LTE-EPC Network simulAtor.* CTTC [online]. Available from: http://networks.cttc.es/mobile-networks/software-tools/lena/

[22] S. Spinsante, M. Pizzichini, M. Mencarelli, S. Squartini and E. Gambi, *Evaluation of the Wireless M-Bus standard for future smart water grids,* in Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International, 2013, pp. 1382âĂŞ1387.

[23] Y. Zhou, H. Wang, S. Zheng, Z.Z. Lei, *Advances in IEEE 802.11ah standardization for machine-type communications in sub-1GHz WLAN,* Communications Workshops (ICC), 2013 IEEE International Conference on , vol., no., pp.1269,1273, 9-13 June 2013 doi: 10.1109/ICCW.2013.6649432

[24] T. Adame, A. Bel, B. Bellalta, J. Barcelo, M. Oliver, *IEEE 802.11AH: the WiFi approach for M2M communications,* Wireless Communications, IEEE , vol.21, no.6, pp.144,152, December 2014 doi: 10.1109/MWC.2014.7000982

[25] EN 13757-4. *Communication systems for meters and remote reading of meters - Part 4: Wireless meter readout (Radio Meter reading for operation in the 868-870 MHz SRD band).* Brusel: European Committee for Standardization, 2003. Available from: http://oldfjarrvarme.unc.se/download/1309/fj

[26] Cisco. *Voice Over IP - Per Call Bandwidth Consumption.* Cisco [online]. Available from: http://www.cisco.com/c/en/us/support/docs/voice/voice-quality/7934-bwidth-consume.html

[27] Bonega: *Water-Meters and Accessories.* Available from: http://bit.ly/1F0LQo1

[28] WepTech: *Wireless Technology.* Available from: https://www.weptech.de/

[29] Pikkerton: *Wireless M-BUS devices.* Available from: http://www.pikkerton.com/

[30] ZPA Smart Energy: *Going the Smarter Way.* Available from: http://www.zpa.cz/

[31] Wireshark: *Network protocol analyzer.* Available from: https://www.wireshark.org/

[32] 3GPP TS 29.281: *General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U).*

Multi Service Proxy: Mobile Web Traffic Entitlement Point in 4G Core Network

Dalibor Uhlir, Dominik Kovac, Jiri Hosek

***Abstract*—Core part of state-of-the-art mobile networks is composed of several standard elements like GGSN (Gateway General Packet Radio Service Support Node), SGSN (Serving GPRS Support Node), F5 or MSP (Multi Service Proxy). Each node handles network traffic from a slightly different perspective, and with various goals. In this article we will focus only on the MSP, its key features and especially on related security issues. MSP handles all HTTP traffic in the mobile network and therefore it is a suitable point for the implementation of different optimization functions, e.g. to reduce the volume of data generated by YouTube or similar HTTP-based service. This article will introduce basic features and functions of MSP as well as ways of remote access and security mechanisms of this key element in state-of-the-art mobile networks.**

***Keywords*—4G cellular network, LTE, Mobile traffic, Multi Service Proxy, Security, SSL, Web services.**

I. Introduction

Mobile networks went throw a huge development during last 30 years. From NMT (Nordic Mobile Telephony, 1G) based on analog technology, over GSM (Groupe Special Mobile, 2G) with both data and voice signals sent over circuit switched network and 3G, where voice is sent via circuit switched and classic data is sent as IP (Internet Protocol) based flow, to LTE (Long Term Evolution, 4G) network where both data and voice (VoLTE – Voice over LTE) use IP channel. Current mobile network is complex communication system composed of large number of nodes. However, the emerging LTE deployment includes three key parts: 1) RAN (Radio Access Network), 2) core network and 3) IMS (IP Multimedia Subsystem) as optional but very common component (see Fig. 1).

The current situation in utilization of cellular networks and growing demands of their users introduce several challenges to which the mobile operators will have to face sooner or later. Especially the following facts need to be taken into a consideration:

- Traffic from wireless and mobile devices will exceed traffic from wired devices by 2016 [1].
- HTTP traffic is taking the pole position in residential broadband Internet traffic [2], [3].
- Around 34% of HTTP traffic was found to be multimedia [4] and one of dominant multimedia server is YouTube.

This paper is addressing the MSP server located in LTE core network and describes mainly security access for its managing via TSL (Transport Layer Security) and also a protection of user's secure entitlement traffic. The article discusses the system of certifications and their renewal used by MSP and moreover offers the way how to increase security using current solution and shows alternative ways for certification. The article also critically analyzes the key weaknesses of core network components in today's environment and explain potential security holes.

D. Uhlir, D. Kovac and J. Hosek are with the Department of Telecommunications, Brno University of Technology, Czech Republic e-mail: xuhlir15@stud.feec.vutbr.cz, xkovac23@phd.feec.vutbr.cz, hosek@feec.vutbr.cz
Manuscript received June 6, 2015.

II. Multi Service Proxy

MSP (Multi Service Proxy) is the element in mobile core network which contains several types of nodes. The key part is the database system composed of several servers which handle network traffic. The other parts of MSP include the traffic servers (TS), administration nodes and the jump start server. Figure 1 shows logical position of MSP in mobile network and its interconnections to other core nodes.

MSP network elements are deployed as several chassis (each chassis has several blades) within one rack. Each chassis is de facto the UNIX machine with web server, database server and NetBackup solution installed.

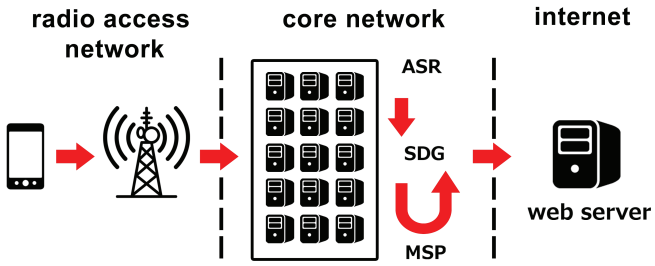


Fig. 1: MSP in core network

MSP is located on the same level as ASR (Accounting Start Request) which is an extended edge router providing functionality of SGW/PGW (Serving Gateway / Packet Data Network Gateway) and SDG (Service Delivery Gateway). The SDG works beside others also as load balancer. S/P gateways in LTE networks have similar functionality as SGSN (Serving GPRS Support Node) and GGSN (Gateway General Packet Radio Service Support Node) in 3G networks.

MSP is located between SDG and Internet so when a user using his smartphone, laptop or other mobile device sends HTTP message, the SDG forwards it automatically to MSP. Then, the MSP processes this kind of traffic and sends the message back to SDG. After that, SDG sends traffic to Internet. The HTTP traffic is recognized by a source or destination port 80. Besides HTTP, MSP processes also secure entitlement traffic which is utilizing HTTPS. However, it represents smaller percentage share of all web traffic. The detailed topology of LTE core network is depicted in Fig. 2).

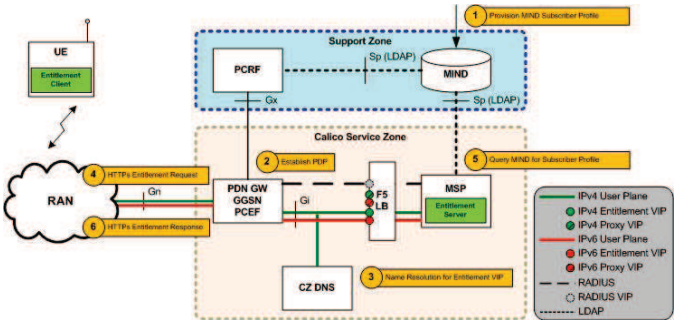


Fig. 2: LTE core network solution including the MSP

There are several methods how to process HTTP traffic on MSP. The most frequently used mechanisms are a header enrichment, compression, implementation of URL (Uniform Resource Locator) rules, pacing, etc. Traffic compression for example is mostly applied to following video formats: mp4, quicktime, x-f4v, x-flv and 3gpp.

When the network traffic is processed by MSP and sent back to SDG, it than goes to ASR and RAN network. This is default behavior for each user, but can be changed by custom configuration of MSP which means that MSP will not modify the traffic for selected users. This is done exceptionally, usually based on a user's request (if confirmed by network provider). From the point of view of physical implementation, the SDG, ASR and MSP are all located in the same rack.

III. MSP Administration

MSP is managed via MSA (Multi Site Admin) to assure service and network availability and sufficient performance of MSP. MSA client is running in web browser which connects to web server running on a database server in MSP system.

Via MSA GUI (Graphical User Interface, see Fig. 3) a network administrator maintains the configuration of MSP. It is possible to manage whole chassis, traffic servers, database server and admin server as well. The configuration management is used to view and edit the values of centrally stored parameter used for browsing, streaming and push services. In order to keep the configuration as efficient as possible, the centralized approach is applied. It means that a network administrator connects to MSP database server does his job (update the settings) and then database server propagates all updates to traffic servers automatically.

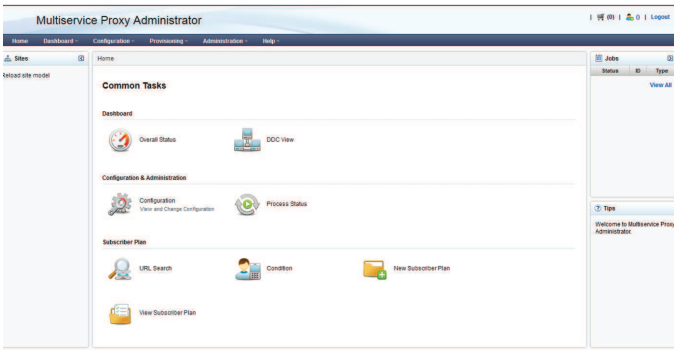
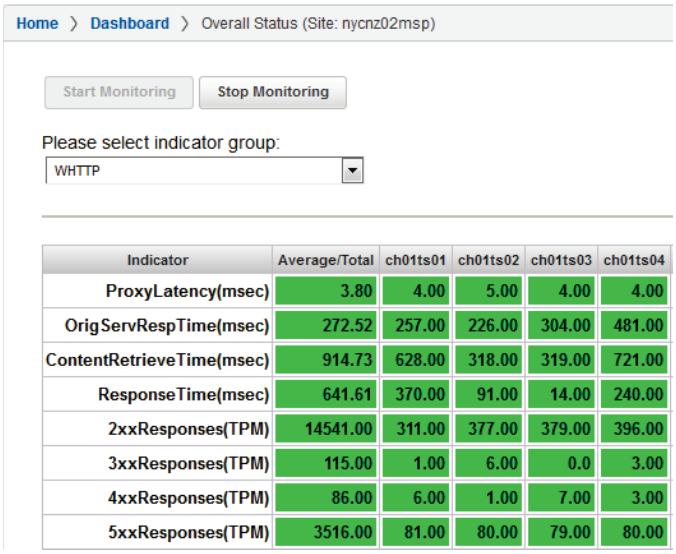


Fig. 3: MSA web interface to maintain MSP

MSA GUI provides also the dashboard which enables to check the current status of MSP and see its performance introduced as KPIs (Key Performance Indicators) collected via SNMP (Simple Network Management Protocol) from each node (see Fig. 4). The KPIs includes the following metrics [4]:

- amount of traffic,
- percentage of successful requests from users,
- distribution of HTTP status codes received by users,
- services running on TS and their status,
- utilization of memory, processors and file systems on all MSP elements.



Indicator	Average/Total	ch01ts01	ch01ts02	ch01ts03	ch01ts04
ProxyLatency(msec)	3.80	4.00	5.00	4.00	4.00
OrigServRespTime(msec)	272.52	257.00	226.00	304.00	481.00
ContentRetrieveTime(msec)	914.73	628.00	318.00	319.00	721.00
ResponseTime(msec)	641.61	370.00	91.00	14.00	240.00
2xxResponses(TPM)	14541.00	311.00	377.00	379.00	396.00
3xxResponses(TPM)	115.00	1.00	6.00	0.0	3.00
4xxResponses(TPM)	86.00	6.00	1.00	7.00	3.00
5xxResponses(TPM)	3516.00	81.00	80.00	79.00	80.00

Fig. 4: Example of MSP dashboard

One of available settings on MSP (manageable via MSA) is video content adaptation (see Fig. 5), sometimes referred also as multimedia content adaptation, content re-purposing, content reuse or re-authoring. Using this technology, mobile operators are able to reduce amount of network traffic and so save the bandwidth. The content adaptations is based on the transformation of logical set of video streams, images, text and other media from a source to one or more destinations. The most frequently used mechanism is the media transcoding to another format (format conversion) in order to reduce the bitrate.

MSP acts as proxy server and sends the requests to the server on behalf of the client. When the reply from server is received, MSP performs the content adaptation and sends adapted data to the client. The content adaptation covers five basic techniques:

- information abstraction,
- modality transformation,
- data transcoding,
- data prioritization,
- purpose classification.

IV. Entitlement Traffic and Server

The Secure Service Based Entitlement (SSBE) architecture is secure method for an entitlement client on a device to query

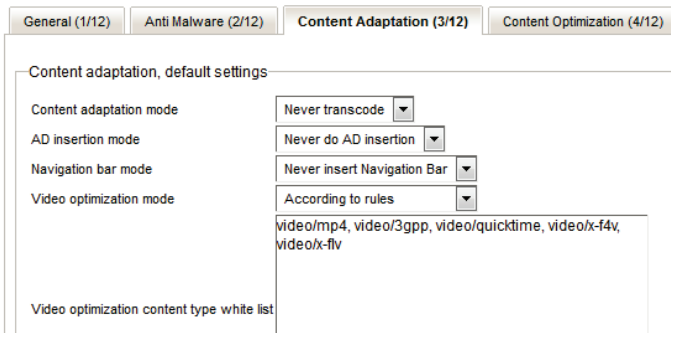


Fig. 5: Video content adaptation on MSP

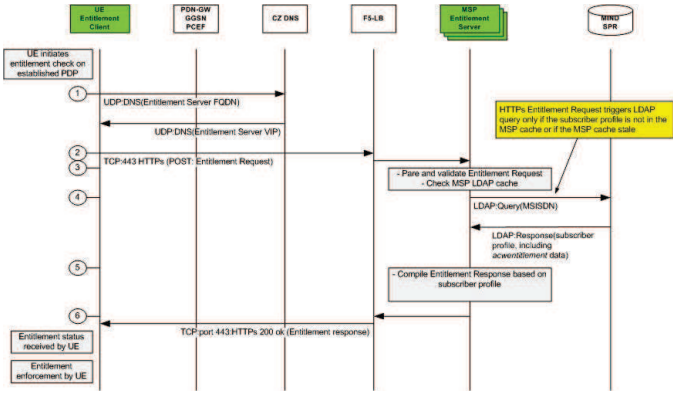


Fig. 6: Secure entitlement workflow

provider's network for entitlements of specific services (tethering, FaceTime, etc.). It is enhancement to MSP to provide SSBE functionality. The entitlement server is a software running as daemon on MSP which processes queries entitlements of services. Entitlement enforcement is in responsibility of the client (e.g. smartphone).

The entitlement client is configured with an entitlement URL and communicates with the MSP secure entitlement server. Entitlement server (daemon located on traffic servers) contains workflow scripts to handle incoming HTTP requests from mobile devices requesting entitlement status of a service. There are two kinds of entitlements - simple one utilizing HTTP and secure one which is carried over HTTPS. The location of entitlement server and client is shown in Fig. 2 as green boxes.

A. Secure Entitlement

In case of secure entitlement traffic, the load balancer (SDG) forwards TCP:443 HTTPS POST Entitlement requests (included in the body of HTTPS POST request) to MSP. In secure entitlement, two requests are supported: getEntitlement and getPhoneNumber. The getEntitlement is used by the device to query the entitlement server for the entitlement status of services that this subscriber should or should not be allowed to use. On the other side, the getPhoneNumber is used by the device to request the entitlement server to inform about the MSISDN (Mobile Subscriber/Station Integrated Services Digital Network) and corresponding signature. In order to generate such signature, a certificate and entitlement server's private key files need to be configured in the secure entitlement parameters subgroup in MSA. The algorithm used to encrypt the message is SHA1 (Secure Hash Algorithm). More detailed description is provided in the next section V.

When a GetPhoneNumber action is received within the secure entitlement request, the entitlement server will generate a signature that will be sent back to the client as part of the response. The whole secure entitlement workflow process is shown in Fig. 6 [5].

Secure entitlement traffic can be created by different applications. One example is the FaceTime which is videotelephony and voice over IP application from Apple [6]. At the beginning, the FaceTime worked only via WiFi networks, however starting with iOS 6 also the support for mobile networks is implemented. To be able to operate FaceTime over mobile network, carrier's support is required. Apple devices have the mechanism to decipher the request sent by MSP. In other words, user equipment (UE) resolves the entitlement FQDN (Fully Qualified Domain Name) in the entitlement URL to the secure entitlement server via DNS (Domain Name System). Then the UE sends a HTTPS POST entitlement request to one of the secure entitlement servers (selected by load balancing process on SDG). The MSP secure entitlement server parses and validates the entitlement request. For valid requests, the secure entitlement server uses the IP source address of the UE to check the MSP LDAP DDC (Lightweight Directory Access Protocol Distributed Data Cache) cache for the profile of the subscriber. MSP retrieves the subscriber profile and based on it, the secure entitlement server compiles the entitlement response. After that, MSP sends a HTTP 200 back to the UE for valid requests. The HTTP message body contains the complete entitlement response.

B. Simple Entitlement Architecture

In a simple entitlement architecture, the requests are sent in form of HTTP GET request with an empty body. The workflow scripts that support simple entitlement requests are configured as subscriber plan specific scripts. Here are two possible response codes as an answer to a simple entitlement request:

- 200 OK - if the client is entitled to the requested service.
- Pre-configured 4xx response - in case the client is not entitled or an internal error occurs.

The entitlement server can be then configured to add, remove or modify entitlement services on a URL basis (entitlement URL) for both secure and simple entitlements. Each entitlement URL is associated with an entitlement protocol, an entitlement service, LDAP attribute and entitlement value. Example of simple entitlement is the determination of whether a subscriber is allowed to use tethering with their rate plan. The entitlement enforcement is responsibility of each client.

V. SSL Certificates

SSL (Secure Socket Layer) is cryptographic protocol that performs a security related functions and applies secure communication. SSL encrypts data of network connections in

the application layer of OSI model and uses both symmetric (communication between client and server - AES, DES) and asymmetric key (to authenticate and change symmetric key) [7]. There are three types of SSL certificates: extended, organization and domain validation. In mobile network we need to authorize network engineers on servers in network and carrier's clients to access services [8], therefore SSL is also used for the connection to maintenance interfaces of MSP.

A. MSA SSL Certificate Implementation

MSA SSL certificate is installed on database servers to allow a secure connection to the MSA GUI. Client (network engineer) connects to MSA via HTTPS (to requests secure page). The database server sends back to client its public key and certificate (generated with keytool command and received with the signature from Certification Authority). A client checks whether the certificate was issued by a trusted Certificate Authority (CA), if the certificate has valid date and if it is related to MSA. Then client uses this public key (which was sent from database server together with the certificate) to encrypt a random symmetric encryption key and sends it to the server (together with request for the web page). The server decrypts the symmetric encryption key using its private key and uses the symmetric key to decrypt requested URL and sends requested page (HTML) to client.

Based on information from [9], in order to receive the certificate for MSA and get it implemented, the VeriSign Managed PKI Server Certificate Registration Request is sent via Security Request Center (SRC) as first. This is an internal process in any carrier's communication system.

When approved, the Certificate Signing Request is generated (CSR file) on database server using UNIX command "keytool -genkey" (a new pair of private and public keys is generated) and "ketytool -certreq" (generates the certificate request using the private key created in previous step). In other words, when public and private keys are generated the certificate request is sent to CA. File *request.cer* is generated and uploaded to Authority Servers Managed PKI for SSL Subscriber Services. After that the certificate (.cert file) together with primary, secondary and root certificates are received back from CA and implemented using UNIX command "keytoll -import".

B. SE SSL Certificate Implementation

When it is approved to request certificate via Security Request Center (following the same internal process as described above), RSA (Rivest Shamir Adleman) private key is created using "openssl genrsa -out file.key" command, see Fig 7.

```
hstnz01msp4ts11:~/test # openssl genrsa -out server.key.new 2048
Generating RSA private key, 2048 bit long modulus
.....................................................................................+++
.....................................+++
e is 65537 (0x10001)
hstnz01msp4ts11:~/test # openssl rsa -noout -text -in server.key.new
Private-Key: (2048 bit)
modulus:
    00:ad:27:25:f2:a7:7a:93:1a:64:5c:08:7e:e6:6c:
    4c:c8:0a:26:ce:10:bf:47:74:ca:8d:a2:f8:e6:87:
    03:55:a3:89:0d:7b:17:ef:bd:5c:8f:b8:8c:9f:56:
    1b:24:c6:76:c4:d8:76:4d:72:2f:39:94:3c:9e:52:
    bc:ab:94:08:f5:60:f0:4a:a4:2e:98:6c:a1:18:68:
    36:98:28:f3:35:ee:3e:06:41:74:8c:45:93:01:ef:
    43:cd:f3:7f:2c:c8:3b:21:4e:93:0e:4e:44:74:1c:
    cf:a6:bf:80:17:07:f3:bf:29:ff:2a:d6:94:b2:a4:
```

Fig. 7: Private key generation

Then the file.csr (request for certificate) is created from file.key (private key generated in the step before) using command "openssl req -key file.key -out file.csr", see Fig 8 and so the CSR is obtained. This file is sent to CA and later the SSL certificate (.pem file) is received as answer. Together with intermediate certificates and private file.key, the SSL certificate is uploaded to MSP traffic server.

```
hstnz01msp4ts11:~/test # openssl req -new -key server.key.new -out server.csr.new
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:CZ
State or Province Name (full name) [Some-State]:Bohemia
Locality Name (eg, city) []:Brno
Organization Name (eg, company) [Internet Widgits Pty Ltd]:VUT
Organizational Unit Name (eg, section) []:FEKT
Common Name (e.g. server FQDN or YOUR name) []:Dalibor
Email Address []:dalibor.uhlir@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Fig. 8: Certificate request

Container file server.pem (contains public certificate) is used to sign the response for entitlement request if user is entitled for the services. The getPhoneNumber request signed with the public key from secure entitlement server can be sent (after using certificate to make sure that the key is valid) and the response sent from secure entitlement server is decoded by mobile devices utilizing the already mentioned built-in function.

As signature algorithm, the SHA1 is used and signature of the issued key (x509) in server.pem has to match signature of the certificate (RSA) in the server.key file.

VI. Attacks Against MSP

Mobile carriers need to have a good level of security to protect especially the nodes and systems used for billing and users' data privacy purposes. Without those security mechanisms, any technically skilled users would be able to manage the billing of his service profile their data to another user's account. Another potential risk (when no security is implemented) is a sniffing and modifying mobile network traffic and so get access to users private information like for example their accounts' credentials. Therefore, the SSL and other security algorithms utilized in mobile networks play crucial role. However, even that the security is implemented there are several recognized types of attack against the SSL:

- beast TLS attack [10],
- renegotiation attack [11],
- version rollback [12],
- poodle attack [13],
- RC4 attack [14],
- Heartbleed [15],

Besides the above listed SSL attacks examples, there are other possible security issues related to HTTP traffic and its processing in mobile network. Some of them is e.g. the CA issue. Nowadays, there are too many CAs and some of them could be compromised or they can be corrupted so they issue certificates even for addresses that are banned to require a certificate (e.g. localhost) or CA can issue even a

fake certificate. Another well-known vulnerability point is a carrier's own employee.

In order to avoid the security problems, there are several standard ways how to improve it:

- Increase the length of private key.
- Change the fundamental principles of security system. Currently, the web browsers expect that server sends SSL certificate and browser then validates it against the set of root CA integrated in a browser or operating system. Assuming this we can implement different models:
  1) DNSEC (Domain Name System Security Extension) - mapping public key on DNS
  2) Web of trust - everyone can generate own PGP (Pretty Good Privacy) key
  3) Perspective project - new approach to help computers communicate securely on the Internet [**?**]

VII. Conclusion

In this article, the MSP element as one of key components of state-of-the-art mobile networks has been introduced. The focus has been given especially to its mechanisms for HTTP / HTTP traffic processing and adoption. Also the implemented security algorithms and means to maintain the MSP have been introduced. We have also discussed some potential security risks and offers solutions which decrease possibility of successful attack. The network traffic optimization is currently highly discussed issue in 4G networks and therefore the development of MSP and similar solutions is very active. The key contribution of this paper lies in uncovering the internal procedures and mechanisms used in the core part of cellular network in order to relieve the network load. Such information is usually protected by vendors and operators, however, to develop high quality mobile service, it is important to understand inner processes employed by the network nodes.

Acknowledgment

For this research, the infrastructure of the SIX Center was used.

References

[1] Cisco Visual Networking Index: Forecast and Methodology, 20132018 http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html/

[2] G. Maier, A. Feldmann, V. Paxson, and M. Allman, On dominant characteristics of residential broadband internet traffic, in Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference, ser. IMC 09. New York, NY, USA: ACM, 2009, pp. 90102

[3] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, Internet Inter-Domain Traffic, in Proc. of the ACM SIGCOMM 2010 Conference on SIGCOMM, New Delhi, India, Aug. 2010.

[4] J. Erman, A. Gerber, M. T. Hajiaghayi, D. Pei, and O. Spatscheck, Network-Aware Forward Caching, in WWW. ACM, 2009, pp. 291 300.

[5] A. Kalgaonkar: ATT Matrix 1.0 Project - Workflow Scripts Administration Guide, 2013.

[6] Apple Inc., iOS Security - White Paper, 2015.

[7] Wikipedia contributors. "Public-key cryptography." Wikipedia, The Free Encyclopedia. 2015.

[8] C. Timberg: "Huge flaw that undermines privacy of mobile phone networks revealed by German researchers", Washington Post, 2014.

[9] F. Herrgoss, Gus Bourg, Kieran Kavanagh: Calico 2.3, 2011.

[10] I. Ristic, Is BEAST Still a Threat?, Qualys Blog, 2013. Available from: https://community.qualys.com/blogs/securitylabs/2013/09/10/is-beast-still-a-threat

[11] M. Ray, Understanding the TLS Renegotiation Attack, Educated Guesswork, 2009. Available from: http://www.educatedguesswork.org/2009/11/understanding_the_tls_renegoti.html

[12] H. Zhang, Three attacks in SSL protocol and their solutions, University of Auckland, 2011. Available from: https://www.cs.auckland.ac.nz/courses/compsci725s2c/archive/termpapers/725zhang.pdf

[13] B. Maller, T. Duong, K. Kotowicz, This POODLE Bites: Exploiting The SSL 3.0 Fallback Security Advisory, Google, 2014.

[14] J. Lv, B. Zhang, D. Lin, Distinguishing Attacks on RC4 and A New Improvement of the Cipher, 2013. Available from: https://eprint.iacr.org/2013/176.pdf

[15] A. Borges, How to perform a Heartbleed Attack, 2014. Available from: http://alexandreborgesbrazil.files.wordpress.com/2014/04/heartbleed_article_rev_b.pdf

KALASHNIKOV IZHEVSK STATE TECHNICAL UNIVERSITY

XI International Conference

"Instrumentation Engineering in the XXI Century. Integration of Science, Education and Production"

The Conference is held in Kalashnikov Izhevsk State Technical University annually since 2004. We invite scientists, specialists, postgraduate and undergraduate students to discuss their achievements in field of electronic instrument engineering, to strenghten creative communications, increase the efficiency of universities, research organisations and enterprises scientific potential implementation in addressing the priorities of instrument making scientific and practical problems.

Instrumentation Engineering, Electronics and Telecommunications – 2015

1st International Forum

IEET-2015

We proudly announce the launch of the International Forum "Instrumentation Engineering, Electronics and Telecommunications - 2015" that will be held for the first time within the framework of the Conference with the aim to gather the brightest scientific achievements mainly, but not limited, in the following areas:

1. **Instrumentation Engineering** (Nanomaterial and Nanotechnology; Renewable Energy; Embedded Systems and Robotics; Industrial Automation and Control; Test, measurement and instrumentation; Space Instrument Engineering; Non-Destructive Testing; Control and Monitoring Systems; Physical Methods for Instrument Engineering; Biomedical Instrumentation; Embedded Systems and Robotics).
2. **Electronics** (Micro Machines and Microelectronics; Micro/Nano-Electromechanical Systems; Electric and Electronics Engineering; Design and Manufacture of Electronic Means; Analog and digital circuit design; Sensors and Actuators; Analogue and Digital Signal Processing, Radio Engineering; Algorithms and Software).
3. **Telecommunications** (Communication Theory; Transport Telecommunication Systems; Modelling, Simulation and Measurements for Telecommunications; Wireless Networking; Mobile Communication Systems; Network Services and Applications; P2P Networks; Mobile Ad hoc Networks; Internet of Things).

All accepted papers will be published in the electronic IEET 2015 Forum Proceedings. Authors out of Russia may present their work using videoconference and participate without any registration fee. The Proceedings with presented papers will be submitted to Conference Proceedings Citation Index (CPCI) of Thomson Reuters, SCOPUS, RSCI, and Google Scholar databases. Selected papers will be published in special issue of the International Journal IJATES^2. For more details about IEET 2015 and contacts, please visit http://pribor21.istu.ru.

PAPER SUBMISSION DEADLINE: OCTOBER, 23, 2015

IZHEVSK, RUSSIA
NOVEMBER, 25–27
2015

http://pribor21.istu.ru

ISSN 1805-5443