

# Universal Synchronization Algorithm for Wireless Sensor Networks—“FUSA algorithm”

Michal Chovanec  
University of Žilina

Faculty of Management Science and Informatics  
Department of Technical Cybernetics  
Univerzitná 8215/1, 010 26 Žilina, Slovakia  
Email: michal.chovanec@fri.uniza.sk

Jana Púchyová, Martin Húdik, Michal Kochlán  
University of Žilina

Faculty of Management Science and Informatics  
Department of Technical Cybernetics  
Univerzitná 8215/1, 010 26 Žilina, Slovakia  
Email: {jana.puchyova, martin.hudik, michal.kochlan}@fri.uniza.sk

**Abstract**—Synchronization, in general, is the process of event coordination so that a system performs in unison. In wireless sensor networks (WSN), the network topology often changes dynamically in time. This brings technical hitches and challenges into time synchronization of the sensor nodes in the WSN. In such networks, data routing scheme is either data fusion, fusion of decisions or hybrid fusion. No matter, what data routing scheme is utilized, the time synchronization among the sensors is highly desirable. This paper presents an advanced synchronization algorithm for WSN. Hierarchical as well as decentralized network topology can be used with this algorithm. The designed algorithm is versatile, scalable and easy for implementation. At first, application area of the proposed synchronization algorithm is described. The detailed explanation of the synchronization algorithm at the node level and the network level is supported by the simulation results along with the implementation remarks.

## I. INTRODUCTION

HAVING wireless sensor networks (WSN), regarding their nature, they belong to the category of systems with a great measure of parallelism [1]. In order to effectively utilize the parallelism nature, to ensure real-time communication ability and to focus nodes' computational power to application-oriented algorithms, it is necessary to synchronize the sensor nodes. The time synchronization is crucial for any distributed system. In particular, WSN make extensive use of synchronized time in many contexts [1] (e.g. for data fusion, fusion of decisions, hybrid fusions, time division multiple access (TDMA) schedules, synchronized sleep periods, etc.).

This paper describes a new synchronization algorithm based on the fireflies synchronization process. The described algorithm is versatile regardless the network topology. This means, hierarchical networks with master nodes that control the synchronization process as well as fully distributed homogeneous-sensor-type networks are usable for the proposed synchronization algorithm. This new algorithm has been given the name “*Firefly-based Universal Synchronization Algorithm (FUSA)*”.

### A. Application area

WSN represents an application area of a great potential. The increasing number of WSN deployment, recent advances in micro-electro mechanical systems (MEMS) and new energy

harvesting possibilities make WSN very topical issue [2]. The WSN application potential as well as implementation potential of the proposed synchronization algorithm FUSA includes the following application areas:

- Health-care;
- Transportation;
- Military applications;
- Wearable electronics;
- Industrial applications;
- Intelligent automation/buildings;
- Monitoring of objects and environment (detection of floods, illegal logging, fires, etc.);
- and many other applications.

Health-care WSN applications are modern and interesting application area of WSN [3]. These applications mostly include wearable electronics such as intelligent watches, drug pumps, motion sensors monitoring elder people and those with diseases and disabilities so that they can be kept under the track of their vitality status without major limitations [4]. This modern application area and increasing popularity of health-care applications allow us meeting also WSN that monitor human body vital signs, track patients, monitor hospital environment and many more [5], [6], [7].

Transportation represents a significant portion (at least one third) of the national gross income in the developed countries [8]. Therefore, the need for efficient control of the traffic flows and intelligent monitoring of the traffic is in place. WSN help accomplish aims of the intelligent transportation systems such as traffic mirrors and intelligent traffic crossroads [9], traffic flow classification and quantification, intelligent car park systems and many more.

WSN concept originates in the military application field. Sensor network applications like sniper localization or battle-field monitoring are typical [10].

Wearable electronics include already mentioned health-care applications. However, large application field include sports equipments as well.

Industrial control systems that implement wireless technologies and sensors with actuators represent an advantage over the traditional distributed control systems [11], [12].

This work was supported by Tatra banka Foundation Slovakia

Mostly various diagnosing systems and production line control systems create the industrial application field of WSN.

The energy consumption of the whole society is a crucial problem in terms of the long-term sustainability of the environment. A big portion of the problems related to the environment connects to the energy consumption and energy requirements of the buildings [13], [14]. Intelligent control of the whole building ecosystems represents quite new and very interesting application field not only for the WSN [15], [16].

Property surveillance, object monitoring, environment monitoring (floods detection, fire detection, illegal logging detection, etc.) including protected areas monitoring is another significant application field of WSN [17], [18]. From functional point of view it is very important not to have only algorithms for monitoring - image recognition, voice recognition [19], but also the proper algorithm for increasing the function potential of the whole network [20].

Having in mind WSN for monitoring open area, it is necessary to consider the main characteristics of WSN [8]:

- The network is spread in outdoor terrain, where the energy resource is limited;
- The sensor and actuator nodes have limited energy storage, memory capacity and computational power;
- The network throughput and RF communication bandwidth are limited;
- The interference, path-loss and diffraction phenomena applies in RF communication;
- The environmental conditions as well as network attributes vary in time.

Synchronization algorithm demands are application specific. Typically, applications monitoring environment have the following characteristics [20]:

- Energy efficiency - time needed for synchronization, communication window length and active power modes should be minimized;
- Scalability - usable for different number of nodes;
- Precision - the nodes are able to send the data in proper time;
- Synchronization time - the amount of time needed for synchronization should be as short as possible.

This paper assumes that the application field of the proposed synchronization algorithm is an application for monitoring forests in order to prevent the illegal wood logging situations. All simulations and application remarks are based on this assumption. However, this fact is not in contrary with the versatility of the proposed synchronization algorithm. The mentioned application field is for the illustrative and interpretation purposes.

### B. Related work

Since the WSN applications are specific, not every synchronization algorithm is suitable for WSN purposes. The following paragraphs mention synchronization algorithms whose nature allows WSN utilization.

*Cristian's Algorithm* [21] and *Berkeley Algorithm* [22] are considered as essential synchronization algorithms and we will

not discuss them. Computer networks very often use *Network Time Protocol (NTP)* [23] for time synchronization. However, the standard computer networks do not suffer from limited energy constraints.

Well known and very often used algorithms in WSN are *Reference Broadcast Synchronization (RBS)* [24] and *Timing Synchronization Protocol for Sensor Network (TPSN)* [25]. In RBS, the master node called the beacon node is used for synchronization. The synchronization of the whole network is performed from the beacon node that sends the reference broadcast towards one-hop-distant nodes from beacon node. Large networks with many sensor and/or actuator nodes are usually divided into smaller virtual networks called clusters. TPSN synchronization algorithm works with synchronization master as well. This master node is elected by all nodes. As soon as the master node is elected, the spanning tree of the network is created. The children nodes are being synchronized by their parent node. In case any change in the network topology happens (e.g. a node becomes unavailable) a new master has to be elected again.

A reference point and the construction of the network tree is also used in *Tree Structured Referencing Time Synchronization (TSRT)* [26] and *Lightweight Tree-based Synchronization (LTS)* [27].

Other class of synchronization algorithms that use master node for synchronization or the group of master nodes contains *Time Diffusion Synchronization (TDP)* [28] and *ETSP* [29], which use both TPSN and RBS methodology. They switch between the TPSN and RBS based on the threshold value. The hierarchical structure of WSN is also used in [30], where the big accuracy of synchronized clock can be achieved, but only in simulation environment. Some of the algorithms use conditional probability estimation as well [31], [32], [33].

## II. SYNCHRONIZATION ALGORITHM - FUSA

The synchronization algorithm proposed in this paper can be used in hierarchical as well as in network with fully distributed cooperation and coordination.

Since the communication subsystem in active mode consumes the significant, and in many cases the most, energy of all sensor node subsystems, minimizing active RF communication minimizes the energy consumption of the whole node too. The presented algorithm is based on the fireflies synchronization process - [34] (main idea), [35] (simple practical implementation), [36] (theory).

Each node periodically transmits synchronization packet (any data can be used). Let the basic time period be  $T$ . By using crystal-based clock, it is possible to set the period for each node precisely. However, each node starts at random time instant. This phenomena results in different timing phase start. Despite having crystal-based clock, small deviations in every clock source create deviations in time phase, thus getting all nodes out of the synchronization. This presents a problem and therefore synchronization algorithms are being used to suppress the unwanted effects.

Let's denote the phase for  $N$  nodes by  $\phi_n \in \langle 0, 1 \rangle$ . Then, the maximum phase error in the network can be defined as:

$$\phi_{max}(t) = \max_n(\phi_n(t)) - \min_n(\phi_n(t)) \quad (1)$$

This definition can be represented as network synchronization quality.

The network is fully synchronized when  $\phi_{max}(t) = 0$ . After this point, all nodes are allowed to switch to a sleep mode and they wake up only for a short period of time. In discrete time domain, the best way how to divide the time period  $T$  is to divide it into  $D$  count of the same parts (frames)  $T_d$ , i.e.  $T = D \cdot T_d$ . For the reason of simple practical implementation, in simulations, the dividing factor  $D$  has been equal to  $D = 128$ . The nodes have transmitted the data in  $1s$  time period, which implies  $128Hz$  interrupt timer frequency. Each node is allowed to transmit data only in its time frame, while the rest of the time frames (assigned to other nodes) this node is expected to respect the radio silence (usually turns into the sleep mode).

Each node has a time counter which is incremented periodically in the timer interrupt routine. When it reaches the maximum, then the timer starts decrementing the counter value and at the same time the synchronization packet is transmitted. When the counter reaches the minimum value, the timer starts incrementing. The described process generates triangle wave output, which can be transformed into a phase represented as the sawtooth wave. The triangle (sawtooth) wave is important, as presented in [34], the function cannot be homogeneous (fe.  $time = time \% 128$  will never work).

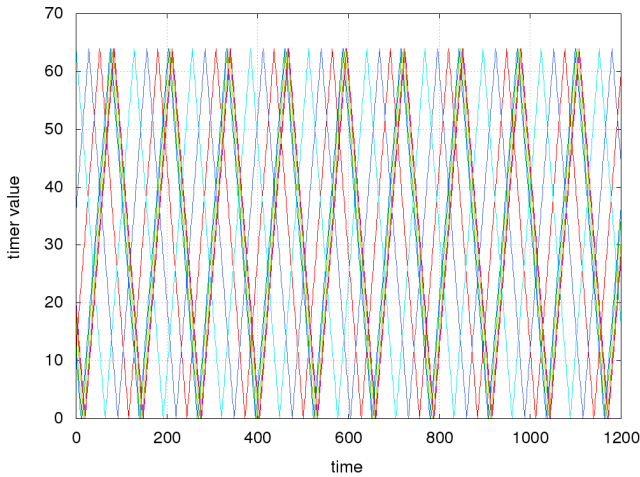


Fig. 1. Phases of nodes without synchronization

Simulations were performed with 64 nodes, in a  $8 \times 8$  grid network topology, where each node can see only four neighboring nodes. This grid network topology is called an *anuloid grid*. Fig. 1 and Fig. 2 demonstrate the network without synchronization - random initial phases. Fig. 1 represents phase values of first 8 nodes. Synchronization quality can be evaluated using (1) and is show in Fig. 2. It is obvious that the

maximum phase error oscillates around the maximum phase value (128) and the average phase is in the center (64).

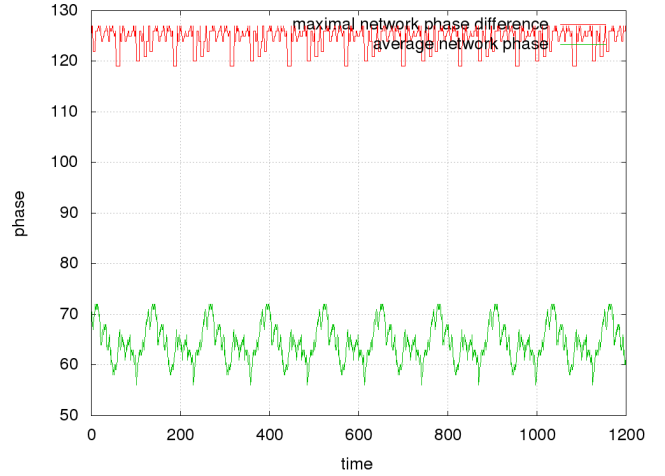


Fig. 2. Maximum and average phase of the network without synchronization

The source code (Algorithm 1) demonstrates the synchronization process in Ruby programming language.

---

#### Algorithm 1 The node synchronization

---

```
def tick(fired)
  @fired_tmp = false

  if fired
    @timer = TIMER_MAX
  end

  if (@state == 0)
    if (@timer >= TIMER_MAX)
      @timer -= 1
      @state = 1
      @fired_tmp = true
    else
      @timer += 1
    end
  else
    if (@timer <= 0)
      @timer += 1
      @state = 0
    else
      @timer -= 1
    end
  end
end
```

---

The method *tick* is called periodically, in discrete time, on each node. The input parameter *fired* has two values:

$$fired = \begin{cases} true & \text{when } node[j][i+1].fired \text{ or} \\ & node[j][i-1].fired \text{ or} \\ & node[j+1][i].fired \text{ or} \\ & node[j-1][i].fired \\ false & \text{else} \end{cases}$$

When any of the neighboring nodes fires (timer is on top), this node sets the timer counter to *TIMER\_MAX* value. Depending on the state, in the next step, the timer is either decremented, or incremented and compared to the top value. If the counter reaches the maximum value, the node fires, and the state is switched.

---

**Algorithm 2** Network synchronization
 

---

```

for j in 0..@net.size-1
  for i in 0..@net[j].size-1
    if (@net[(j+1)%@net.size][i].get_fired) or
      (@net[(j-1)%@net.size][i].get_fired) or
      (@net[j][(i+1)%@net[j].size].get_fired) or
      (@net[j][(i-1)%@net[j].size].get_fired)

      fired = true
    else
      fired = false
    end

    @net[j][i].tick(fired)
  end
end

```

---

The network synchronization must work in discrete time so *fired* flag is stored in *@fired\_tmp* first. After all nodes call the method *tick*, the *fired* flags are updated.

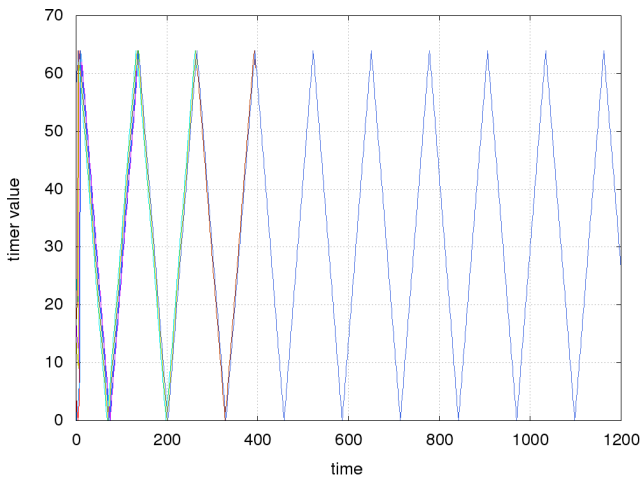


Fig. 3. Phases of nodes with synchronization

Fig. 3 and Fig. 4 demonstrate the network synchronization process. Fig. 3 illustrates the phase of the nodes when synchronization process applies. On Fig. 4 we can see fluent decreasing of the phase error, until it reaches 0.

The following figures Fig. 5, Fig. 6, Fig. 7 and Fig. 8 represent the phase synchronization process and demonstrate the *synchronization wave* in the network in different iterations of the synchronization algorithm. As it can be seen, the waves with the increasing number of synchronization algorithm iterations slightly disappear.

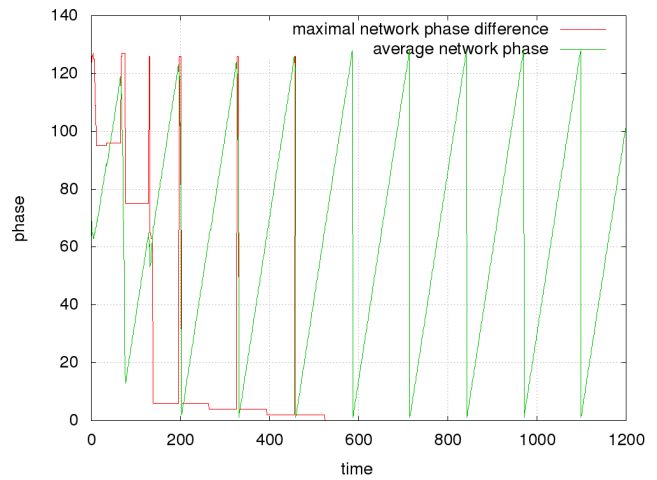


Fig. 4. Maximum and average phase of the network with synchronization

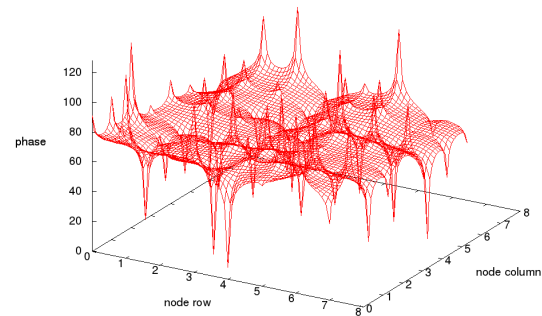


Fig. 5. Network initial phases, iteration 0

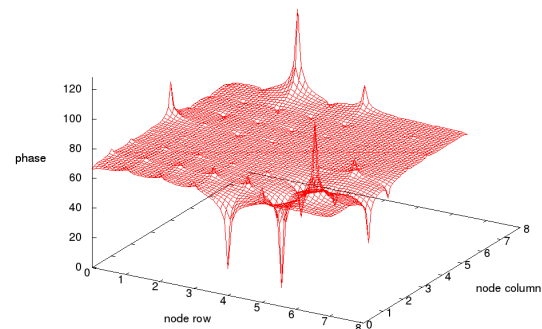


Fig. 6. Network synchronization, iteration 10

By comparing Fig. 6 and Fig. 7 we can see the *synchronization wave*. When the network is fully synchronized, all

phases are the same and we get a straight plane as illustrated in Fig.8. During the simulations performed on the self-developed simulator, we found out that the proposed synchronization algorithm is fully functional despite of node failure. This has also no effect on the overall WSN operation.

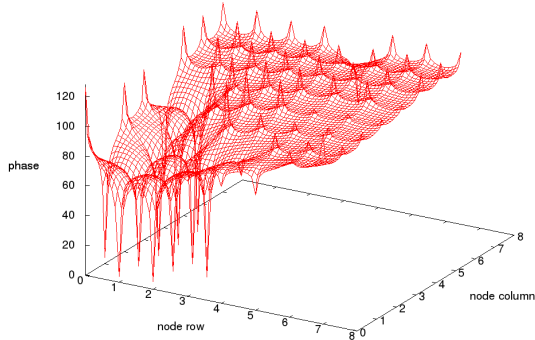


Fig. 7. Network synchronization, iteration 70

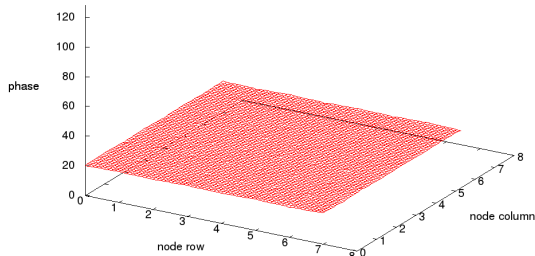


Fig. 8. Network synchronization, iteration 1180

### III. NODE IMPLEMENTATION

The experimental evaluation of the proposed synchronization algorithm has been performed on wireless sensor nodes based on Texas Instruments MSP430 family microcontrollers (MSP430F2232, 8kB FLASH, 512b RAM) (Fig. 9). The communication subsystem (RF) part is based on Texas Instruments transceiver CC1101 - a low power transceiver operating at 868MHz ISM band. Other sensor node subsystems built-in on-board are 3.3V low-drop regulator, three-axis magnetometer, two LEDs and an UART peripheral for debugging. Further details about the nodes can be found in [9]. Red board in Fig. 9 was used as a power supply and for debugging purposes.

After general input/output peripheral (GPIO) initialization and RF module initialization, the *timer0a* interrupt is set

to 256Hz periodic invoke (derived from external 32.768kHz crystal-based clock source). This initialization is described in the Algorithm 3.

---

#### Algorithm 3 Low power timer initialization

---

```

/* f = ACLK (32768Hz) / TACCRO*/
TACCRO = 128;

/* select ACLK/1, up mode, and clear the TAR */
TACTL = TASSEL_1 + MC_1 + TACLR;

/* enable timer interrupt*/
TACCTL0 = CCIE;

```

---

The Algorithm 1 described in the previous paragraphs is implemented in three program subroutines as follows:

- Synchronization with received packet (in GPIO pin interrupt routine);
- Periodical packet transmission and timer control (in timer interrupt routine);
- Higher-level network functions (in the main loop routine).

In GPIO initialization phase, RF receive pin is configured as an interrupt pin. After this steps the microcontroller (MCU) is allowed to enter the LPM3 sleep mode, which it can wake up from with 256Hz period and/or on the packet receive interrupt. The received packet processing is described in the following algorithm - Algorithm 4.

---

#### Algorithm 4 Packet receive interrupt service routine

---

```

interrupt (PORT2_VECTOR) Port_2 (void)
{
    if ((__state__ == 2) &&
        (TI_CC_GDO0_PxIFG & TI_CC_GDO0_PIN))
    {
        u8 len = PACKET_LENGTH-1;
        if (RFReceivePacket((u8*)rxBuffer,&len))
        {
            __state__ = 0;

            i16 add = TIMER_MAX - __phase__;
            __phase__ += add;

            if (__phase__ > TIMER_MAX)
                __phase__ = TIMER_MAX;

            if (__phase__ < TIMER_MIN)
                __phase__ = TIMER_MIN;
        }
    }

    TI_CC_GDO0_PxIFG &= ~TI_CC_GDO0_PIN;
    LPM3_EXIT;
}

```

---

This routine (Algorithm 4) is executed only when the program is in *\_\_state\_\_* == 2. That means packet receiving is enabled, which maximizes the sleep time. When a packet from other node is received, this interrupt routine is executed.

The phase synchronization is done on the following source code lines:

```
i16 add = TIMER_MAX - __phase__;
__phase__ += add;
```

In the timer interrupt routine the timer phase control is processed. This routine is exemplified in Algorithm 5.

---

**Algorithm 5** Timer interrupt service routine

---

```
interrupt(TIMER0_A0_VECTOR)
TIMER0_A0_ISR( void )
{
    __phase__ += __sh_output__ / (i16)TIMER_MAX;

    if ( __phase__ > TIMER_MAX )
        __phase__ = TIMER_MAX;

    if ( __phase__ < TIMER_MIN )
        __phase__ = TIMER_MIN;

    if ( (__phase__ + DEVICE_ADDR) == TIMER_MAX )
        __state__ = 1;

    if ( __sh_output__ == TIMER_MAX )
    {
        if ( __phase__ >= TIMER_MAX )
            __sh_output__ = TIMER_MIN;
    }
    else
    {
        if ( __phase__ <= TIMER_MIN )
            __sh_output__ = TIMER_MAX;
    }

    LPM3_EXIT;
}
```

---

There is important to add a small phase difference, to avoid transmission collision, which is done as follows:

```
if((__phase__ + DEVICE_ADDR) == TIMER_MAX)
    __state__ = 1;
```

When program enters into `__state__ = 1` the received packet can be released to the main loop.

#### IV. CONCLUSION AND FUTURE WORK

For the proper function of each WSN, the node synchronization is very important part of proposed solution. In this paper the algorithm for the synchronization was proposed, which is very easy for implementation. This algorithm is not only for hierarchical networks, it is universal and scalable. Functionality of synchronization algorithm was tested on mesh network up to 1024 nodes. It is good to note that in this paper only part for synchronization process is written. In future the authors would like to extend the algorithm and use algorithm together with monitoring techniques. That means

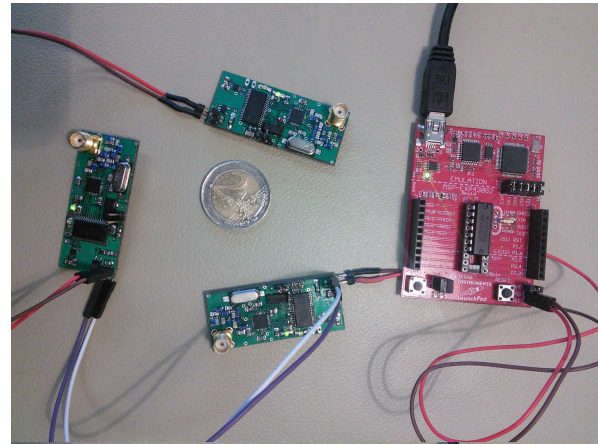


Fig. 9. Testing nodes photo

that the communication window should be divided into the smaller slots for sending the data.

#### ACKNOWLEDGMENT

This publication is the result of the project implementation "WSN pre monitoring a ochranu územia pohoria Malá Fatra" with grant no. 2013et032 in grant program "E-Talent 2013" supported by Tatra banka Foundation Slovakia.



#### REFERENCES

- [1] J. Elson, and K. Römer, "Wireless sensor networks: A new regime for time synchronization", in *ACM SIGCOMM Computer Communication Review*, Vol. 33, No. 1, 2003, pp. 149–154.
- [2] M. Kochláň, J. Miček, and M. Hyben, "Wireless sensor network energy harvesting: radio frequency harvesting case study", in *Intelligent transportation systems 2013 (ITS 2013)*, August 26-30, 2013, ISSN 1339-4118, ISBN 978-80-554-0763-0, pp. 93–97.
- [3] M. Kochláň, M. Hodoň, and J. Púchyová, "Vital functions monitoring via Sensor Body Area Network with smartphone network coordinator", in *MEMSTECH 2013: perspective technologies and methods in MEMS design*, April 16-20, 2013, Polyana, Ukraine, Lviv Polytechnic Publishing House, ISBN 978-617-607-424-3, pp. 143–147.
- [4] J. Púchyová, M. Kochláň, and M. Hodoň, "Development of special smartphone-based Body Area Network: Energy Requirements", in *Federated conference on computer science and information systems (FedCSIS)*, September 8-11, 2013, Kraków, Poland, ISBN 978-1-4673-4471-5, pp. 915–920.
- [5] D. Laqua, and P. Husar, "Intelligent Power Management enables Autonomous Power Supply of Sensor Systems for Modern Prostheses", in *Biomedizinische Technik/Biomedical Engineering (Impact Factor: 1.16)*, September, 2012, pp. 247–250. DOI:10.1515/bmt-2012-4055.
- [6] J. Miček, O. Karpiš and P. Ševčík, "Body area network: analysis and application areas", in *International journal of engineering research and development (IJERD)*, ISSN 2278-800X, Vol. 6, No. 8, 2013, pp. 22–26.
- [7] A. Hofmann, D. Laqua, and P. Husar, "Piezoelectric Based Energy Management System for Powering Intelligent Implants and Prostheses", *Biomedizinische Technik/Biomedical Engineering (Impact Factor: 1.16)*, September, 2012, pp. 263–266. DOI:10.1515/bmt-2012-4265.
- [8] M. Kochláň, and J. Miček, "Indoor Propagation of 2.4GHz Radio Signal: Propagation Models and Experimental Results", in *Digital Technologies 2014 (DT2014)*, July, 9-11, 2014, [in print].

- [9] M. Hodoň, M. Chovanec, and M. Hyben, "Intelligent traffic-safety mirror", in *Studia informatica universalis*, ISSN 1621-7545, 2013, Vol. 11, No. 1, pp. 87–101.
- [10] O. Karpiš, and J. Miček, "Sniper localization using WSN", in *ICMT'11: International conference on military technologies*, Brno, Czech Republic, May 10-11, 2011, ISBN 978-80-7231-787-5, pp. 1063–1068.
- [11] V. C. Gungor, and G. P. Hancke, "Industrial Wireless Sensor Networks: Challenges, Design Principles and Technical Approaches", in *IEEE Transactions in Industrial Electronics*, Vol. 56, No.10, 2009.
- [12] A. Flamminia, P. Ferraria, D. Mariolia, E. Sisinnia, and A. Taronib, "Wired and wireless sensor networks for industrial applications", in *2nd IEEE International Workshop on Advances in Sensors and Interfaces*, Vol. 40, No. 9, September 2009, pp. 1322–1336.
- [13] U. S. Department of Energy, "2011 Building Energy Data Book", 2012.
- [14] European Commission, "Report from the Commission to the European Parliament and the Council", Brussels 18.4.2013.
- [15] P. Ševčík, and O. Kovář, "Alternative energy sources for WSN node power supply", in *ITS 2013: Intelligent transportation systems 2013*, August 26-30, 2013, ISSN 1339-4118, ISBN 978-80-554-0763-0, pp. 146–149.
- [16] J. Miček, and M. Kochláň, "Energy-efficient communication systems of wireless sensor networks", in *Studia informatica universalis*, ISSN 1621-7545, 2013, Vol. 11, No. 1, pp. 69–86.
- [17] J. Miček, and J. Kapitulík, "WSN sensor node for protected area monitoring," in *Federated Conference on Computer Science and Information Systems*, 2012, pp. 803–807.
- [18] O. Karpiš, J. Juríček, and J. Miček, "Application of wireless sensor networks for road monitoring," in *10th IFAC workshop on programmable devices and embedded systems*, Vol. 3, 2013, pp. 611–617.
- [19] M. Hyben and M. Hodoň, "Low-cost command-recognition device," in *Przegląd teleinformatyczny*, 2013, ISSN 2300-5149, Vol. 1, No. 3, pp. 19–28.
- [20] J. Papán, M. Jurečka, and J. Púchyová, "WSN for forest monitoring to prevent illegal logging", in *FedCSIS: proceedings of the Federated conference on computer science and information systems*, September 9-12, 2012, Wrocław, Poland, ISBN 978-83-60810-51-4.
- [21] F. Cristian, "Probabilistic clock synchronization," *Distributed Computing*, vol. 3, 1989, pp. 146–158.
- [22] R. Gusella and S. Zatti, "The accuracy of the clock synchronization achieved by TEMPO in Berkeley UNIX 4.3BSD," in *IEEE Transactions on Software Engineering*, 1989, pp. 847–853.
- [23] D. L. Mills, "Internet time synchronization: the Network Time Protocol," in *IEEE Trans. Communications COM-39, 10*, 1991, pp. 1482–1493.
- [24] E. Jeremy, G. Lewis and E. Deborah, "Fine-grained network time synchronization using reference broadcasts," in *Fifth Symposium on Operating Systems Design and Implementation OSDI*, 2002.
- [25] S. Ganeriwal, K. Ram and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *First ACM Conference on Embedded Networked Sensor Systems* 2003.
- [26] A. Rahamatkar and A. Agarwal, "A Reference Based, Tree Structured Time Synchronization Approach and its Analysis in WSN," *International Journal of Ad hoc, Sensor & Ubiquitous Computing*, Vol. 2, 2011, pp. 20–31.
- [27] J. V. Greunen and J. Rabaey, "Lightweight Time Synchronization for Sensor Networks," in *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications (WSNA)*, San Diego, CA, September 2003.
- [28] W. Su and I. F. Akyildiz, "Time-Diffusion Synchronization Protocols for Sensor Networks," in *IEEE/ACM Transactions on Networking*, Vol. 13, 2005, pp. 384–397.
- [29] K. Shahzad, A. Ali and N.D. Gohar, "ETSP: An Energy-Efficient Time Synchronization Protocol for Wireless Sensor Networks," in *22nd International Conference on Advanced Information Networking and Applications - Workshops*, 2008, pp. 971–976.
- [30] R. Albu, Y. Labit, G. Thierry and B. Pascal, "An Energy-efficient Clock Synchronization Protocol for Wireless Sensor Networks," *Wireless Days*, 2010, pp. 1–5.
- [31] A. P. Sage and G. W. Husa, "Algorithms for Sequential Adaptive Estimation of Prior Statistics," *IEEE Symposium on Adaptive Processes Decision and Control*, 1969, pp. 760–769.
- [32] C. Bo, D. Enqing, L. Xiaoyang, Z. Dejing and W. Jiaren, "A time synchronization algorithm based on bimodal clock frequency estimation," in *18th Asia-Pacific Conference on Communications (APCC)*, 2012, pp. 75–78.
- [33] J. Kim, J. Lee, E. Serpedin and K. Qaraqe, "A robust clock synchronization algorithm for wireless sensor networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 3512–3515.
- [34] R. E. Mirollo and S. H. Strogatz, "Synchronization of Pulse-Coupled Biological Oscillators," *SIAM Journal on Applied Mathematics*, vol. 50, 1990, pp. 1645–1662.
- [35] A. Tyrrell, G. Auer and C. Bettstetter, "Firefly Synchronization in Ad Hoc Networks," in *3rd MiNEMA Workshop*, Lueven, Belgium, 2006.
- [36] A. Tyrrell, G. Auer and C. Bettstetter, "Fireflies as Role Models for Synchronization in Ad Hoc Networks," *Bio-Inspired Models of Network, Information and Computing Systems*, 2006, pp. 1–7.