This full text paper was peer-reviewed at the direction of IEEE Instrumentation and Measurement Society prior to the acceptance and publication.

Design of a Low Cost Multipurpose Wireless Sensor Network

A. Di Nisio, T. Di Noia, C. Guarnieri Calò Carducci, M. Spadavecchia
Department of Electrical & Information Engineering (DEI)
Politecnico di Bari
Via E. Orabona, 4 - 70125 Bari, Italy
[dinisio, guarnieri, spadavecchia]@misure.poliba.it, tommaso.dinoia@poliba.it

***Abstract*—This paper proposes the implementation of a low-cost and reliable multipurpose wireless sensor network framework. It has been designed to be easily customizable in order to adapt it to the particular application.**

Keywords—Wireless Sensor Network (WSN); Arduino; Event network; Internet of Things (IoT)

I. INTRODUCTION

The Internet of Things becomes every day more and more close. Whereas at present it is estimated that 99.4% of the existing objects is yet unconnected (only 10^{10} of 1.5×10^{12}) [1], in the coming years it is expected a widespread distribution of network-connected devices, with a forecast of volumes that would justify the name, that some have attributed to, as *Internet of Everything*.

However, we have not yet the real perception of the impact such a technological transition will have, we can only estimate the scope by analogy with the past. For example in the words of Prof. Michael Nelson of Georgetown University: *"Trying to determine the market size for the Internet of Things is like trying to calculate the market for plastics, in 1940. At that time, it was difficult to imagine that plastics could be in everything"*.[2]

Nevertheless, the road is still long and more work needs to be carried out at multiple layers of abstraction, represented in Fig. 1, the so-called data value pyramid.

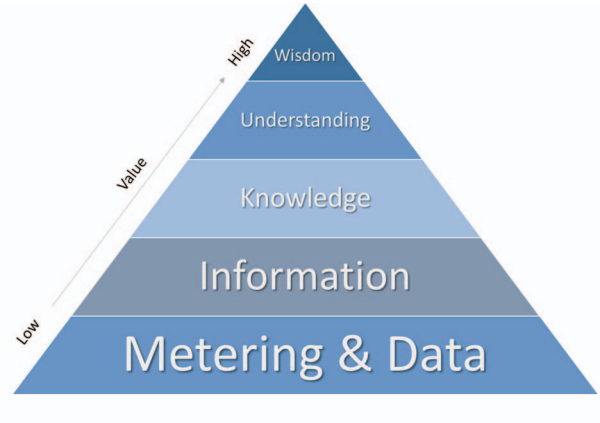

Fig. 1. Data value pyramid (from low value/raw data at the bottom, to high value/cooked data at the top)

Nowadays the research is mainly focused on the three lowest levels, while the two highest are mostly unexplored. Some research groups are working on protocols [3], other on safety standards [4], others on the algorithms for pattern recognition, whereas the explosion of low-cost and open-source electronics [5], together with the lower position held in the pyramid of value, means that the physical implementation of smart objects is often taken for granted.

Bearing in mind the actual price of most microcontrollers [6] as well as of most common sensors, the lower limit to the cost of a remote measurement node is often dictated by the use of communication modules employing proprietary protocols, whose impact on the overall cost of the sensor node can vary from 50 % up to 90 %. On the other hand, more computational power becomes available with affordable costs, which extends the number of applications and algorithms that can be implemented on Wireless Sensor Networks (WSNs), such as image processing, surveillance, remote metering and industrial process control.[7]-[10]

In this article, therefore, the authors intend to describe the implementation of a WSN through a highly versatile framework, with a constant focus on the characteristics of low-cost and low-power consumption.

Many WSNs have been proposed in the literature and several projects have been developed which employs low-cost microcontrollers and RF transceivers. For example, the nRF24 series of 2.4 GHz transceivers, by Nordic Semiconductors, has attracted the attention of developers and makers due to its low-power consumption and its availability as cheap break-out boards. Similarly, microcontrollers such as Atmel ATmega328 (and the related Arduino UNO board) and Texas Instruments MSP430 have been widely adopted inside sensor nodes. Even though it is possible to have sensor nodes with computational power and energy sufficient to host a web server and enable complex interactions with clients, the common choice is to reduce the hardware requirements of the nodes.

For example, the architecture described in [11] is composed by many Arduino and nRF24 based sensor nodes and a more powerful gateway capable of Wi-Fi communication, which exchanges data with a dedicated cloud service or directly with clients. On the contrary, the design proposed in this paper is aimed not only at keeping at the minimum hardware and

software complexities and costs of the tiny sensor nodes, but also of the Wi-Fi gateway, whose sole intended purpose is to permit the communication between the sensor network and the IP network.

The architecture of the MySensors project [12] includes Arduino compatible nRF24 devices with different roles: sensors/actuators, repeaters and gateway. The gateway allows the communication between the sensor network and a third-party controller through a serial or Ethernet cabled link; the controller has more powerful computational capabilities and performs functions such as configuring the sensors, collecting data, interfacing with the users and scheduling operations. In the architecture described in this paper, the gateway has Wi-Fi capabilities and the controller, placed anywhere in the Internet, is substituted by any generic platform with Apache/MySQL/PHP/Python capabilities.

A third example is provided in [13], where the base station, based on the Arduino board with a WiFly Wi-Fi module [14], is capable of sending Tweets. In our system the gateway supports bidirectional communication between the sensor network and the server, where data storage and complex user interfaces can be implemented.

The proposed system can be used in different fields ranging from education [15]- [16] to automotive [17], environment [3], energy monitoring [18]-[19], automation and robotics.

II. Description of the Network

The proposed system is composed of *N* smart sensor nodes connected through Industrial Scientific and Medical (ISM) wireless modules to a gateway, which subsequently gives access and visibility to the rest of the network through a common Wireless Access Point (WAP) (Fig. 2).

At the other end of the network, a Linux server is responsible for gathering and sending data from/to transducer nodes, for populating a dedicated database, for generating events of interest and for providing a graphical interface to display trends. In the following, each block of the network is described in detail.

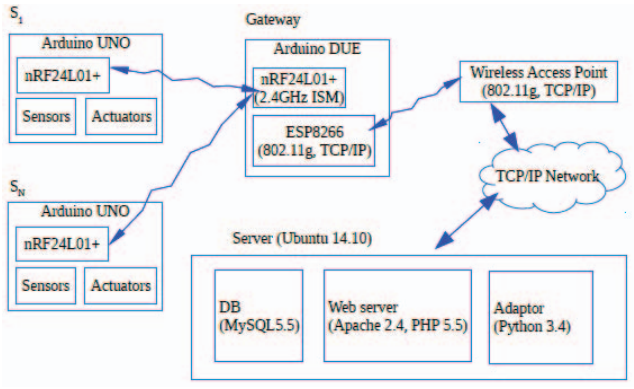

Fig. 2. Network architecture scheme

A. Smart sensor nodes

Each node of the network has been implemented using an Arduino UNO board (Fig. 3), which is based on the 8-bit Atmel ATmega328P microcontroller and presents many benefits:

- high number of GPIOs;
- board cheapness;
- small prototyping time;
- low power consumption;
- easy code portability to other microcontrollers of the same family with different features.

These boards communicate with the Gateway using an SPI connected, extremely cheap and reliable module, the nRF24L01+ chipsets from Nordic Semiconductor, working in the ISM band at 2.4 GHz. Thanks to the native Enhanced ShockBurst™ mode, they are able to perform automatic packet assembling, detection, validation and retransmission, giving to the SPI master the almost exclusively task of reading and sending packets. The microcontroller board can be easily connected to sensors and/or actuators.

The currently adopted topology is a star network, however, despite the NRF modules do not have native support for mesh networks, these can be easily emulated through the use of additional libraries, providing a higher flexibility.

It is important to emphasize that this chip has a power consumption of only 3 µW in Power-down mode and 72 µW in Standby mode I, in which it is able to generate an interrupt upon the reception of a packet, in order to wake up the microcontroller from its power-down state in which it typically requires a current consumptions of 0.1 µA [20], leading to a power consumption of 0.3 µW if working with 3 V supply. A further detailed analysis on power consumption and battery lifetime is reported in section III.

B. Gateway

Differently, the gateway has been implemented using an Arduino DUE board (Fig. 4), based on the Atmel SAM3X8E ARM Cortex-M3 CPU, a 32-bit ARM microcontroller.

The gateway connects the sensor network based on the nRF24L01+ modules, in which it has the logic address 0, with the rest of the network based on IP protocol. The latter link is provided by the ESP8266 Wi-Fi chipset, which integrates a complete TCP/IP protocol stack. It adds, at the beginning of the payload of each packet received by the nRF24L01+ module, the address of the sender and forwards it to the server via the ESP8266 Wi-Fi link. Conversely, in receive mode, the gateway extracts the first byte of the packet received by the ESP8266 (Fig. 5) in order to obtain the address of the destination smart sensor, then forwards the packets by means of the nRF24L01+.

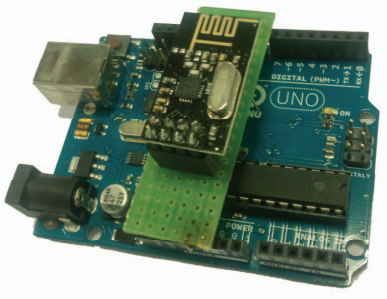

Fig. 3. Smart sensor prototype (Arduino UNO + nRF24L01+)

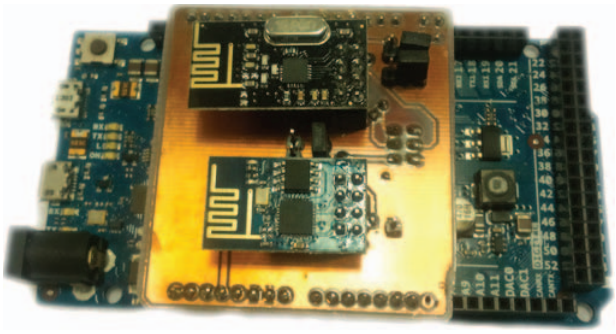

Fig. 4. Gateway test unit (Arduino DUE + nRF24L01 + ESP8266)

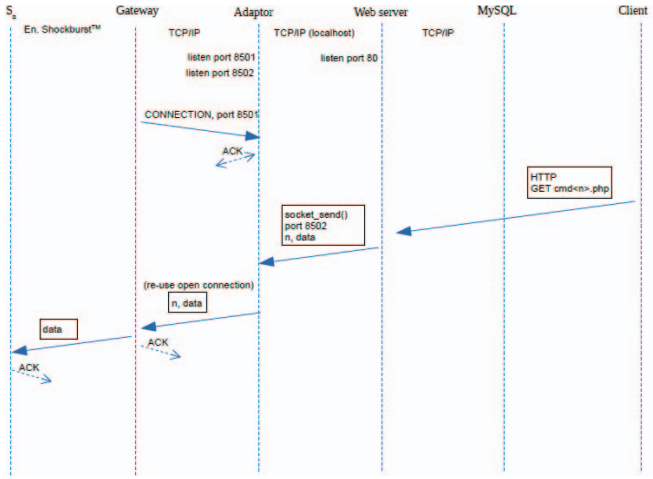

Fig. 5. Remote smart sensor control sequence diagram

The ESP8266 module is used in a basic configuration, and is flashed with a firmware that makes it a generic communication module with a UART interface. Since the ESP8266 includes an application processor, the use of the Arduino DUE board is not strictly necessary in final applications; it has been used in this architecture in order to easily test the code for protocol translation. Therefore, unless a complex local intelligence in the gateway is necessary, the translation protocol engine can be embedded directly on the ESP8266 using the same code written for Arduino DUE board, which in this case would communicate directly with the nRF24L01+ through SPI interface.

C. *Server*

The server is implemented in Ubuntu 14.10 and provides SSH and SFTP services for remote administration and content uploading.

Three main blocks run on the server:

- **Adaptor** server: it is a multi-threaded server implemented in Python with a listening port (8501) for packets coming from the gateway. Each received packet (Fig. 6) generates two HTTP POST requests to the Web server:
  1) to *http://localhost/store0.php*
  2) to *http://localhost/store<n>.php*, where *<n>* is the smart sensor address of the incoming packet.

  The Adaptor has an additional listening port, 8502, for payloads arriving from the Web server, which are subsequently forwarded to the gateway. These payloads must have, as the first byte, the address of the smart sensor.

- **MySQL** server: the *smart_objects_01* database is available in the local network and can also be administered through phpMyAdmin. All the smart sensors interactions are stored in the *logger* table (Fig. 7), which is populated by *store0.php*. The database also contains the *data<Sensor><n>* table that is populated by the script *store<n>.php*.

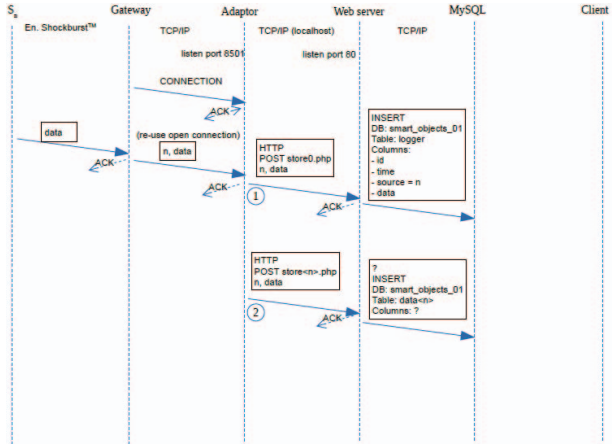

Fig. 6. Data storing sequence diagram for smart sensors

- **Web** server: it is an Apache 2.4 server with PHP 5.5. It executes the previously mentioned scripts: *store0.php* that stores the incoming data from each smart sensor in the *logger* table and *store<n>.php*, which accepts HTTP POST requests generated by the Python Adaptor upon reception of data from the node *<n>*. Fig. 8 shows the sequence diagram relevant to page *read0.html*, which is able to read and plot (see Fig. 9) data from the table with automatic updating via AJAX technique, aided by the script *read_update0.php*.

#	Name	Type	Collation	Attributes	Null	Default	Extra	Act
1	id	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	
2	time	timestamp		on update CURRENT_TIMESTAMP	No	CURRENT_TIMESTAMP	ON UPDATE CURRENT_TIMESTAMP	
3	source	int(3)		UNSIGNED	No	None		
4	data	varbinary(512)			No	None		

Fig. 7. *Logger* table

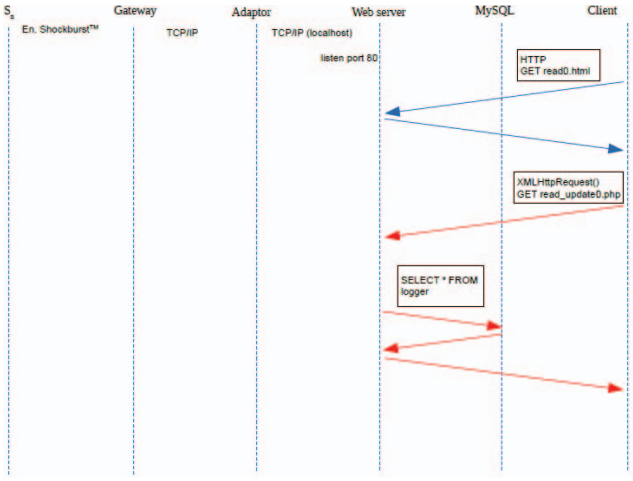

Fig. 8. Sequence diagram

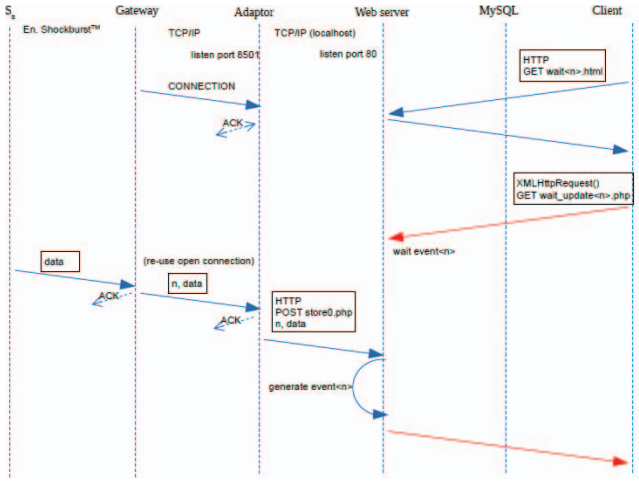

Fig. 10. Events handling and data visualization sequence diagram

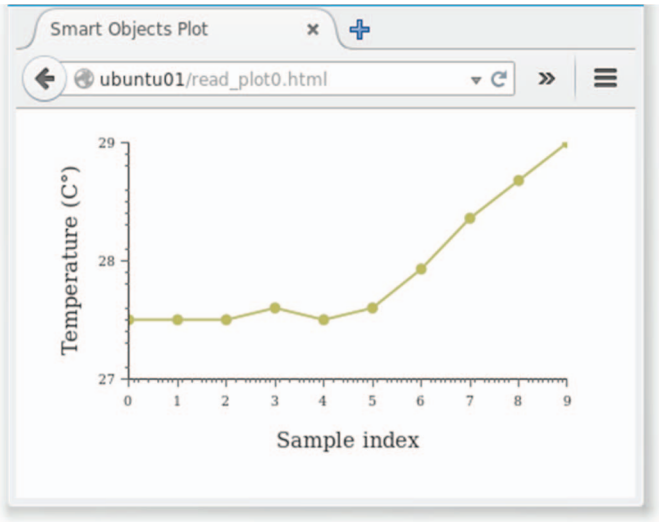

Fig. 9. Data plot

The webpages *wait<n>.html* have been created to report in real time alarm messages and updates from any smart sensor. This has been made possible by using the Sync extension for PHP and inserting a special Event object in scripts *store<n>.php* and *wait_update<n>.php* (Fig. 10) The script *wait_update<n>.php* remains in fact locked (with timeout) until it receives a signaling event from *store<n>.php* and subsequently, thanks to an AJAX request, the *wait<n>.html* page is able to detect the event and updated its content dynamically.

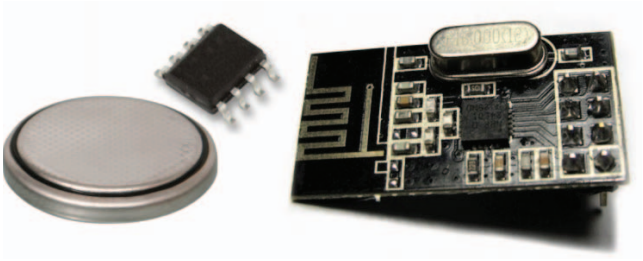

Fig. 11. Basic sensor node components: CR2032 battery, ATtiny85, nRF24L01+

TABLE I. PROTOTYPES BUDGET, AS OF 2015

WSN BLOCKS	End-User Prices	
	Modules	*Cost*
Node 1	ATtiny85 [a] + nRF24L01	1.04 € + 0.44 € = 1.48 €
Node 2	ATmega328P [a] + nRF24L01	1.2 € + 0.44 € = 1.64 €
Gateway	ESP8266 [b] + nRF24L01	2 € + 0.44 € = 2.44 €

[a] *SMD package*

[b] *ESP8266 esp-12*

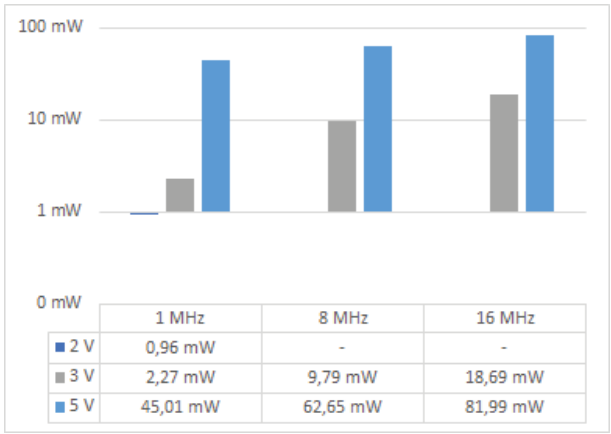

Fig. 12. ATmega328P power consumption

III. Power Consumption and Cost Analysis

As previously mentioned, the presented solution is extremely versatile in both software and hardware side. The particular adopted family of Atmel AVR 8-bit microcontrollers with pico-Power technology implies a sensor node can have high duration if programmed appropriately. The choice of Arduino UNO as a prototyping platform is also justified by the simplicity in the portability of the generated code on similar devices.

Assuming, in fact, that a sensor node requires only one analog input to read a resistive sensor and a digital output for LED signaling or driving a switch, the whole node may be implemented with only three basic components (Fig. 11): a 3.0 V button battery (e.g. CR2032), an nRF24L01+ module and an ATtiny85 microcontroller with only 8 pins, to which we must add the sensor and the actuator. TABLE I. shows the costs for the different blocks of the proposed WSN.

In order to evaluate the power performance of the Arduino based wireless node, different configurations have been analyzed. Clearly, as the clock frequency and the operating voltage increase, a corresponding increase in consumption takes place. For those nodes that do not require continuous stream of data or long time computational tasks, an acceptable compromise (Fig. 12) appears to be the operation with internal oscillator set to 8 MHz and supply voltage equal to almost 3.0V. In the case of sensor nodes based on a minimal ATtiny, the internal oscillator frequency can be reduced without compromising performance even to 1 MHz, further decreasing power consumption.

In the case of ATmega328P, the microcontroller is put into sleep with the possibility of being awakened periodically by the internal watchdog or upon reception of an external interrupt from the wireless module, with a power consumption of respectively 12.6 µW and 0.3 µW [20], depending on the working features. Clearly, there are different behavioral combinations that lead to very different values of lifetime.

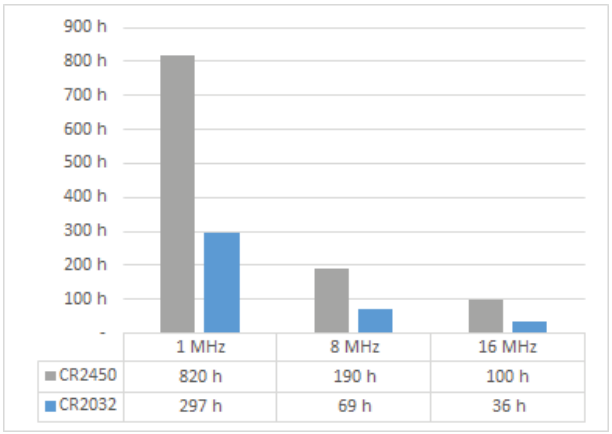

Fig. 13. ATmega328P based node lifetime with respect to the two most suitable button batteries

If the application does not require the remote control of the sensor node, then the functionality of the wake-up interrupt is not required. The microcontroller can be brought into power-down together with the communication module by reducing the global consumption to almost 15.6 µW. In this case, periodic internal watchdog interrupts, with an interval of up to 8 s, can trigger the chip awakening with a reaction time of 0 clock cycles, assuming that an immediate stability of the internal oscillator is not required. At each internal interrupt generated by the watchdog, if the elapsed time is equal to the desired one, the microcontroller will complete the measure and will log the data, otherwise it will return to power-down state.

Several tests (Fig. 13) show that it is likely to get more than 60 hours of run-time operation at 8 MHz with a cheap CR2032 battery. The node's lifetime can be estimated according to

$$life\ (\text{years}) = 6.85 \times \frac{H\ (\text{hours}) \times N\ (\text{minutes})}{T\ (\text{milliseconds})} \qquad (1)$$

where H is the measured runtime life in hours, T is the time in milliseconds requested to perform the entire task and N is the time interval in minutes between tasks. Considering that the entire log procedure, including wake-up, data acquisition, data sending and sleep, can be accomplished in less than T=10 ms, and assuming a sampling period N=1/6 min, (1) gives a lifetime of about 7 years.

Shorter reporting times and/or longer lifetimes can be obtained by exploiting wireless power transmission techniques [21] and energy harvesting techniques, such as those based on photovoltaic and thermoelectric generators [22].

Conclusion

A WSN that can be used in different application contexts was presented. It was developed by using open source software and readily available electronic modules. In spite of its low cost, it permits the bidirectional communication and the report of asynchronous events to connected users. An energy consumption analysis was carried out, which confirmed the possibility of life-long operation.

The design was motivated by the necessity of having low costs for the deployed network, while moving complexities to the central server, where more powerful hardware is easily available and its cost might be spread across several different applications. The first intended use of the system was in laboratory classes for graduate students. In this respect, a useful characteristic of the system was that each function was allocated to a different hardware and or software sub-system, allowing group of students to be focused on a single portion of the system at a time and work concurrently without collapsing the entire network.

References

[1] J. Bradley, J. Barbier, D. Handler, "Embracing the Internet of Everything to capture your share of $14.4 trillion," White paper, Cisco, 2013.

[2] The Hammersmith group, "The Internet of things: Networked objects and smart devices," Research report, Feb 2010 [Online]. Available: http://www.internet-of-things.eu/resources/documents/networked-objects, visitend June 25, 2015.

[3] F. Adamo, F. Attivissimo, C. Guarnieri Calò Carducci, A. M. L. Lanzolla, "A smart sensor network for sea water quality monitoring," IEEE Sensors Journal, vol. 15, pp.2514-2522, May 2015.

[4] IEEE Standard for a Smart Transducer Interface for Sensors and Actuators-Transducer to Microprocessor Comunication Protocol Transducer Electronic Data Aheet (TEDS) Formats, IEEE Standard 1451.2-1997, 1997.

[5] A. P. J. Chandra, C. R. Venugopal, “Novel design solutions for remote access, acquire and control of laboratory experiment on DC machines,” IEEE on Instr. & Meas., vol. 61, pp. 349-358, Feb.2012.

[6] Arduino website, www.arduino.cc, visited June 25, 2015.

[7] M. Rahimi, R. Baer, O. I. Iroezi, J. C. Garcia, J. Warrior, D. Estrin, and M. Srivastava, “Cyclops: in situ image sensing and interpretation in wireless sensor networks,” in Proc. 3rd Int. Conf. on Embedded networked sensor systems (SenSys '05), San Diego, Nov. 2-4, 2005, pp. 192-204.

[8] F. Adamo, F. Attivissimo and A. Di Nisio, "Calibration of an inspection system for online quality control of satin glass," IEEE Trans. on Instr. and Meas., vol. 59, no. 5, pp. 1035-1046, May 2010.

[9] L. Ferrigno, V. Paciello, A. Pietrosanto, "Visual sensors for remote metering in public networks," IEEE Instrumentation and Measurement Technology Conference (I2MTC), Binjiang, 10-12 May 2011, pp-1-6.

[10] V.C. Gungor, G.P. Hancke, "Industrial wireless sensor networks: challenges, design principles, and technical approaches," IEEE Trans. on Industrial Electronics, vol.56, no.10, pp.4258-4265, Oct. 2009.

[11] V. S. Gupta , P. Raspaile “Low cost standard Internet of Things,” International Journal of Engineering Science & Advanced Technology, vol. 5, no. 2, pp. 78-80, Mar-Apr 2015.

[12] MySensors, online: http://www.mysensors.org, visited August 08, 2015.

[13] Md. N. Chowdhury, MD. M. H. Bhuiyan, S. Islam, “IOT: detection of keys, controlling machines and wireless sensing via mesh networking through Internet,” Global Journal of Researches In Engineering, vol. 13, no. 13, pp. 1-8, Oct. 2013.

[14] Microchip WiFly RN-171. Datasheet [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/70005171A.pdf, visited August 08, 2015.

[15] M. Chirico, A. M. Scapolla, A. Bagnasco, “A new and open model to share laboratories on the Internet,” IEEE Trans. on Instr. & Meas., vol. 54, June 2012, pp. 1111-1118.

[16] F. Adamo, F. Attivissimo, G. Cavone, N. Giaquinto, “SCADA/HMI systems in advanced educational courses,” IEEE Trans. on Instrum. Meas., vol. 56, pp. 4-10, Feb. 2007.

[17] D. I. Katzourakis, E. Velenis, D. A. Abbink, R. HAppee, E. Holweg, “Driver’s arms time-variant neuromuscolar admittance during real car test-track driving,” IEEE Trans. on Instr. & Meas., vol. 63, pp. 221-230, Jannuary 2014.

[18] W. Chuyuan, L. Yongzhen, "Design of energy consumption monitoring and energy-saving management system of intelligent building based on the Internet of things," 2011 International Conference on Electronics, Communications and Control (ICECC), 9-11 Sept. 2011, pp. 3650-3652.

[19] F. Adamo, F. Attivissimo, A. Di Nisio, M. Savino, M. Spadavecchia, “A spectral estimation method for nonstationary signals analysis with application to power systems,” Measurement, vol.73, pp 247-261, Sept. 2015.

[20] Atmel 8-bit AVR Microcontroller Family. Datasheet [Online]. Available: http://www.atmel.com/images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf

[21] L. Angrisani, G. D'Alessandro, M. D'Apuzzo, M. D'Arco, "Enabling induction and wireless power transmission technologies aimed at supplying remote equipment in critical logistic scenarios," IEEE International Workshop on Measurements and Networking Proceedings (M&N), 7-8 Oct. 2013, pp.184-188.

[22] F. Attivissimo, A. Di Nisio, A.M.L Lanzolla, M. Paul, "Feasibility of a photovoltaic–thermoelectric generator: performance analysis and simulation results," IEEE Trans. on Instrumentation and Measurement , vol.64, no.5, pp.1158-1169, May 2015.