Hardware Architecture of a Low-Cost Scalable Energy Monitor System

E. M. Pinheiro, S. D. Correia
School of Technology and Management, Polytechnic Institute of Portalegre
Portalegre, Portugal

Abstract

Energy efficiency is decisive for the success of any business. For this reason, the implementation of an Energy Management System is highly advisable. A key component of energy management is monitoring. This paper details the hardware implementation of a low cost and highly scalable energy monitoring solution based on the Raspberry Pi and Arduino platforms, and aimed at non-domestic use.

Keywords: *Raspberry Pi, Arduino, Energy Monitor, Open Hardware*

I. INTRODUCTION

For most of their existence, humans have used fire to produce energy. Wood, coal, and later, fossil fuels, have been burned to harvest the chemical energy contained within. These primary energy resources are a vital part of any country natural capital, and a decisive element on their economic competiveness. However, these resources are also finite, and with increased demand, especially in fossil fuels, there is the indication that they might soon run out [1].

Because of this, all OECD, and many non-OECD countries, have adopted policy measures on energy efficiency. Energy efficiency consists in the attempt to reduce the amount of energy required to provide a certain product or service [1], the corollary being the ability to produce the same with less energy, or to produce the same level of energy from fewer quantities of fossil fuels. These are the objectives of energy management.

Energy management is defined as the strategy of adjusting and optimizing energy consumption, in order to maintain system output while reducing its energy cost [2]. To achieve this goal, an Energy Management System is required, and an integral part of it is energy monitoring.

Most energy management systems are aimed at industrial and grid-based level. However, these are expensive and difficult to use [3]. Furthermore, the increase of the average number of household appliances per residence, and the increase of energy expenditure that follows, has opened the way to the development of home energy monitoring systems (HEMS). These are much simpler to use, and far less expensive, but at the cost of severe limitations, whether in terms of scaling, or information production.

In the following paper, we provide an architecture for an energy monitoring system that aims to combine the cost and usability of a HEMS with the scale and functionality of a commercial industrial level one, i.e., we aim to apply the hardware paradigm of HEMS to an industrial setting.

II. STATE OF THE ART

With the dwindling of natural resources, a significant level of attention has been given to energy monitoring systems, particularly of the domestic variety. The reason is twofold. The advent of the Raspberry Pi in the context of IoT solutions opened the field to new possibilities. The Raspberry Pi is a small all-in-one single-board computer initially designed as an IT education aid. While not comparatively the most powerful (that honour goes to Udoo x86) [4], it is however inexpensive and versatile enough to be highly adopted, not only for its intended purpose, but also by hobbyists and professionals alike. This affordability made the development of Raspberry Pi solutions, including HEMS, a *grassroots* movement, rather than a corporate push, which resulted in a myriad of applications (see [5][6][7][8][9], among many others), often very distinct from one another, and incorporating different hardware. While some rely solely in the Raspberry Pi, other incorporate other easy access, affordable and easy to use hardware platforms, most notably, the Arduino. While affordable and simple to program, its limited processing power has prevented it from being used for more complex tasks. While this has not prevented the development of complex solutions based solely on the Arduino platform [10], it usually comes at the expense of some functionality, especially in terms of user experience (e.g. minimal or absent GUI).

However, the combination of both devices allows for complex, and often distributed, tasks to be performed at a competitive cost.

On the other end, the complexity and closed nature inherent to commercial industrial level EMS, tends to relegate its development to a team of professionals with corporate backup, rather than individual or small number of programmers.

The use of embedded energy monitor systems in industrial settings is a common occurrence, although generally as part of an Energy Power Quality (EPQ) suite [11]. However, the

advent of IoT has begun to change the dominant paradigm at an exponential rate. Coming to prominence in 2005 [12], it has been highly adopted ever since, with a projected growth in the number of connected devices from 14.4 billion in 2015 to 30.7 billion in 2020 and 70.4 billion in 2025 [13]. In this context, the Raspberry Pi, with its Linux based OS, came to the forefront of development, bringing the capabilities and ease of use of a Personal Computer to the domain of sensor networks[14].

This rapid growth of IoT implementations didn't come without its growing pains. While its applications multiply, there hasn't been a consensus on what a standard architecture common to all IoT systems ought to be.[15] However, most IoT systems observe a similar structure, divided in four different layers [16]. A *sensing layer* collects the data being measured; an *access layer* is responsible for transmitting said data; a *service layer*, to process the collected data and other computations; and finally, an*interface layer*, responsible for the interactions between the system and its users, and optimally, in a full IoT environment, other systems. Each of these layers have their own requisites, and often, although not always, require dedicated hardware. This need arises from the design principle that most, if not all, of the layers must be balanced between the devices themselves, and the surrounding infrastructure, such as servers or personal computers. An example can be found inKurkovsky, S and Williams, C 2017 (Fig. 1):

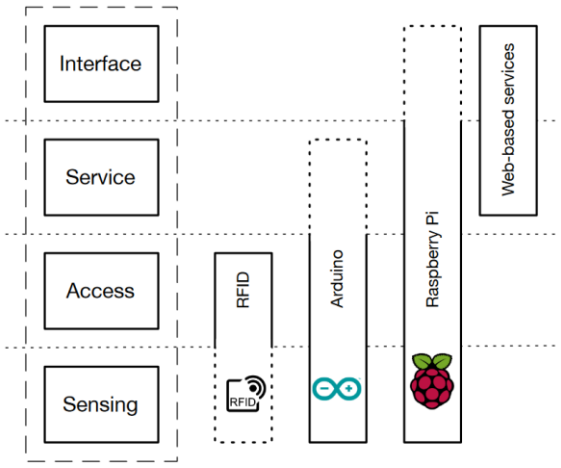

Fig. 1 Hardware Platforms for the Internet of Things Systems and Functional Layers of their Architecture

III.FUNCTIONAL LAYERS

Given the above description, our proposed solution generally follows the given architecture, although with some concessions, to achieve our goals of high scalability and low-cost (Fig. 2).

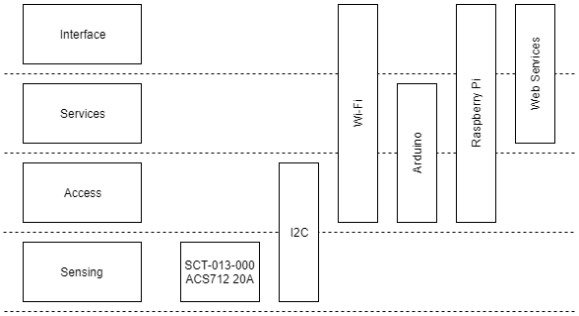

Fig. 2 Hardware and Functional Layers of the Proposed Solution

As can be seen in the above diagram, the main changes occur in the access layer. To increase the number of sensors connected to a single Arduino, an I2C bus is used. To minimize the impact of this solution on pre-existing infrastructure, a dedicated Wi-Fi network is used, by setting up the Raspberry Pi as an Access Point. Naturally, this decision is implementation depended, and if needed or wanted, a general propose wireless LAN can be used

IV. COMPONENTS

A. *Sensors*

The proposed solution utilizes two different sensors. In both cases, the appropriate one must be chosen. Both series, SCT-013 and ACS712, feature a wide array of sensors, with different inputs and outputs, with further details being available in their respective datasheets. It should be noted that, to measure voltage, a voltage divider circuit, consisting of two resistors in series, must be used to reduce the voltage to appropriate levels to be read by the Arduinos' analogue inputs

1) ***SCT-013-XXX:***The SCT-013 is a series of non-invasive current transformer sensor used for measuring alternating current. Being split core type, they are ideal for measuring energy consumption or production of an entire building, since they can be clipped into the live or neutral wire coming into the building without requiring any high-voltage work.

2) ***ACS712 XXA:***The ACS720 is a series of Hall-Effect based linear current sensors. It's designed specifically for easy interface with microcontrollers, such as the Arduino used in this application, since it outputs analogue voltage, from 0 to 5v, depending on the current flowing through the wire.

B. *Communications*

1) ***I^2C:*** The use of I^2C (often spelled I2C, Inter-Integrated Circuit) protocol allows several slaves to be controlled by a single master. The opposite is also true, but of no consequence to the current implementation. This allow for several sensors to be connected to a single Arduino. I2C protocol requires two cables to communicate with a

master, Serial Data (SDA) and Serial Clock (SCL). This first allows for the transmission of data between master and slave, while the second carries a periodical signal which functions as a clock in synchronous communication. To each slave is assigned a unique identifier, a 7 or 10-bit address, which allows the master to communicate with a specific slave. The usage of an I2C interface board is required in order to have a single master controlling several slaves. The specific interface board should be chosen in accordance to the number of necessary sensors.

2) ***Wi-FI****:* In order to minimize the impact on preexisting network infrastructures, the proposed solution utilizes a dedicated wireless network. This is achieved by configuring the Raspberry Pi as an Access Point to which the Arduinos connect. A dedicated shield might be required for earlier versions of Raspberry Pi, but from version 3 forward, a Wi-Fi module is present on-board. In order to connect the Arduinos to the network, an ESP8266 module is used.

3) ***ESP8266***: The ESP8266 is a complete, stand-alone wi-fi networking solution operating on 802.11 b/g/n protocol. It is capable of hosting an application on itself or being used for networking purposes by other devices. When being used as a Wi-Fi adapter it allows connectivity to any other device through UART interface. Furthermore, its low price and the fact that it can be powered by Arduinos' 3.3v pin make it an ideal solution for this application. Since ESP8266 uses the UART interface, if serial communication is required for other purposes, a software serial must be used.

C. Services

1) ***Arduino****:* The Arduino platform was introduced in 2005 and soon garnered widespread use. It consists in a family of embedded processors which can be easily expanded through the use of additional peripherals, granting them additional functionalities. It's low cost and flexibility makes it ideal for the development of networked IoT applications. Because of its popularity, a thriving community grew around it, providing constant development of new libraries, which further facilitate the development of new applications[17].

However, its limited processing power means that it alone cannot bear the bulk of the service and interface layers. Nevertheless, in the proposed solution, it operates as the master to the I2C slaves, reading data from the sensors, and making the necessary calculations. Furthermore, since the processor spends most of its time on idle, it scans the range of I2C addresses, in order to identify recently plugged in devices, thus providing plug-and-play functionality to the solution.

While the sensors could communicate directly with the Raspberry Pi, the use of Arduinos as sensor nodes allow monitoring of wider or multi-building installations through a single Raspberry, ultimately reducing hardware cost.

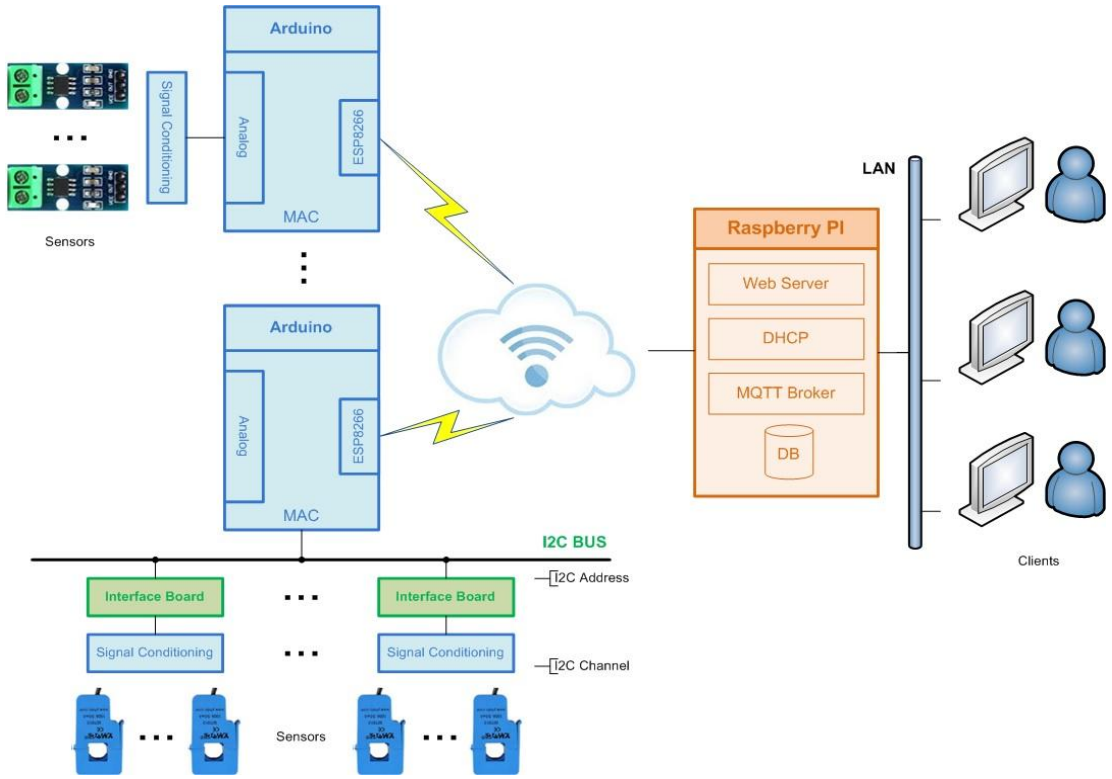

Fig. 3 System Architecture

2) ***Raspberry Pi****:* As stated above, one design principle of IoT implementations is the balancing of layers between different devices. While the Arduino nodes make the necessary computation for calculating current and tension, the Raspberry Pi is responsible for the majority of the services layer, and the entirety of the interface layer. Consisting in an all-in-one single-board computer, sporting a 4x ARM Cortex-A53 1.2GHz CPU, 1GB LPDDR2 of RAM and a SD card slot for storage, it is robust enough to carry the remaining of the required services. In the access layer, not only it serves as an access point, but also as DHCP server and MQTT broker, to allow communication with the different Arduino nodes. In the services layer it runs a mySql server to store readings, as well as calculating statistics on energy consumption, thus being instrumental in turning data into information. Finally, an Apache server is used to provide a web-interface, through which users might configure the system and consult data, corresponding to the interface layer.

V. ARCHITECTURE

Considering the functional layer model presented, as well as the different components, the system architecture begins to emerge. Arduinos read the measurements from the sensors through an I2C bus, and transmit them over through Wi-Fi to the Raspberry Pi, where they are stored in a database, available for consultation when needed.

In order to assure plug-and-play capabilities to the whole solution, between readings, the Arduinos sweep the entire range of I2C addresses in order to identify recently plugged sensors. If a new device is detected, its address is communicated to the Raspberry Pi, where it is stored in the database until configured. Afterwards, the configuration parameters are transmitted back to the respective Arduino, so that the necessary calculations can be made. Likewise, when a new Arduino connects to the Raspberry Pi wireless network, its MAC address is sent to the Raspberry and stored in the database, so that the user might configure it through the web interface. Afterwards, configuration parameters are sent to the Arduino so that it might start operations. An overview of the entire architecture can be glanced from Fig. 3.

VI. RESULTS AND DISCUSSION

The open nature of the solution allows for it to be expanded at will. Additional sensors and functionalities would turn the solution into a fully-fledged Energy Management System. By concentrating most of the services in a single Raspberry Pi it is possible to add functionalities with minimum interference with the established systems. Similarly, the use of Arduinos as nodes allows the use of additional sensors without modifications to the current structure, if these are given a dedicated node. The use of the I2C protocol for communications between the sensors and the Arduinos is the main reason for the proposed solution high scalability. For example, the use of a ADC chip with 8 channels and 3 address lines, such as Maxim Integrated MAX127 allows for up to 64 sensors connected to a single Arduino. The use of I2C interface boards also allows each implementation to be tailored to its needs. By using different interface boards, one can scale the solution to its ideal size. In contrast with the previous example, the use of a LTC2309, with 8 channels, but only 2 address lines, allows for up to 32 sensors connected to a single node, a more cost-effective solution for smaller installations.

The communication structure between the Arduinos and the Raspberry Pi can be expanded without changes to its current form, allowing completely different nodes to be seamlessly integrated with the system, providing that the necessary changes are made to the services layer, e.g. add a configuration interface for the new sensors.

The limitations of the proposed solution lie in the processing power of the Raspberry Pi and network bandwidth. The first one can be overcome by replacing the Raspberry Pi with a more powerful device, although that would defeat the purpose of a low-cost solution. However, consistent improvements and successive version of the hardware are being released into the market, and soon this limitation might be inexistent. The second one can be minimized by using a dedicated access point connected to the Raspberry Pi through an Ethernet interface. While it would increase the cost and complexity of the system, it would nonetheless significantly improve its performance.

VII.CONCLUSION

Due to the growing necessity of efficient energy use, energy management systems are becoming paramount to the success and competiveness of any business. While there are many alternatives for home use, monitoring industrial, sales or office buildings is still reliant on expensive and complex commercial solutions.By applying the lessons and hardware models of home energy monitor systems to a larger scale, it is possible to develop a low cost, highly scalable energy monitor based on open hardware that can be used in wider installations. The usage of I2C protocol allows for a large numberof sensors to be connected to a smaller number of nodes, comprised of Arduinos, and thus reduce the overall cost of the solution.Additionally, the use of a Raspberry Pi as the centre piece of the entire architecture eschews the need of large and expensive servers, which might not be available in some cases.Thus, the implementation of the proposed

solution will result in a low-cost, efficient, scalable, and open monitor system, capable of fulfilling the needs of many different adopters.

REFERENCES

[1] S. F. Hafeez and S. D. Dalvi, "Global trends of energy efficiency programs: 2010 to 2025" *2017* Asian Conference on Energy, Power and Transportation Electrification (ACEPT), Singapore, 2017, pp. 1-8.

[2] M. Hassan, N. Hassan, M. Miah, F.A. Sohaly, I Arefa and Z. H. Mahmood, "Energy Auditing: Necessity for Energy Management System" 3rd International Conference on Mechanical Industrial & Energy Engineering, Khulna, 2014, PI 140379

[3] Jayanth S, Poorvi MB and Sunil MP, "Raspberry Pi based energy management system," 2016 Online International Conference on Green Engineering and Technologies (IC-GET), Coimbatore, 2016, pp. 1-5.

[4] "Udoo x86 Datasheet", SECO USA Inc., Arezzo, Italy

[5] P. van Staden and B. Kotze, "Wireless node energy monitor using common development platforms: Using a Raspberry Pi to monitor energy consumption by a WSN," 2017 IEEE AFRICON, Cape Town, 2017, pp. 1514-1519.

[6] S. M. Patil, M. Vijayalashmi and R. Tapaskar, "IoT based solar energy monitoring system," 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), Chennai, India, 2017, pp. 1574-1579

[7] W. T. Hartman, A. Hansen, E. Vasquez, S. El-Tawab and K. Altaii, "Energy monitoring and control using Internet of Things (IoT) system," 2018 Systems and Information Engineering Design Symposium (SIEDS), Charlottesville, VA, USA, 2018, pp. 13-18

[8] A. Gupta, R. Jain, R. Joshi and R. Saxena, "Real time remote solar monitoring system," 2017 3rd International Conference on Advances in Computing,Communication& Automation (ICACCA) (Fall), Dehradun, 2017, pp. 1-5.

[9] N. A. Othman, M. R. Zainodin, N. Anuar and N. S. Damanhuri, "Remote monitoring system development via Raspberry-Pi for small scale standalone PV plant," *2017 7th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, Penang, 2017, pp. 360-365.

[10] H. B. Patil, V. M. Umale, "Arduino Based Wireless Biomedical Parameter Monitoring System Using Zigbee", International Journal of Engineering Trends and Technology (IJETT), Vol. 28 (7), October 2015

[11] J. A. Back, L. P. Tedesco, R. F. Molz and E. O. B. Nara, "An embedded system approach for energy monitoring and analysis in industrial processes" Energy, Volume 115, Part 1, 2016, Pages 811-819.

[12] International Telecommunication Union, "The Internet of Things" ITU Internet Reports, November 2005.

[13] S. Lucero (2016), IHS Technology. IoT platforms: enabling the Internet of Things. March 2016

[14] M. Maksimovic, V. Vujovic, N. Davidović, V. Milosevic and B. Perisic, "Raspberry Pi as Internet of Things hardware: Performances and Constraints" IcETRAN 2014, Vrnjacka Banja, 2014

[15] M. Weyrichand C. Ebert,"Reference Architectures for the Internet of Things"IEEE Software, Vol. 33(1), pp.112-116, 2016

[16] S. Kurkovsky and C. Williams, "Raspberry Pi as a Platform for the Internet of Things Projects: Experiences and Lessons*",* Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education *(ITiCSE '17)*, New York, 2017, pp. 64-69

[17] S. Hodges, S. Taylor, N. Villar, J. Scott, D. Bial and P. T. Fischer, "Prototyping Connected Devices for the Internet of Things" Computer, vol. 46, no. 2, pp. 26-34, Feb. 2013