The 5th IEEE International Symposium on Smart and Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems
17-18 September, 2020, Dortmund, Germany

Low Power LoRaWAN Node Based on FRAM Microcontroller

Marcel Meli [1], Claudio Furter [2]
Konstantinos Karakostas [3], Charalampos Kouzinopoulos [3], Dimitrios Tzovaras [3]
[1] ZHAW, Technikumstrasse 9, 8401 Winterthur, Switzerland, marcel.meli@zhaw.ch, https://www.zhaw.ch
[2] Cicor Group, Gebenloostrasse 15, 9552 Bronschhofen, Switzerland, claudio.furter@cicor.com, https://www.cicor.com
[3] CERTH, 6th Klm. Charilaou - Thermi Road, 57001 Thermi, Thessaloniki, Greece,
kkarakostas94@gmail.com, {kouzinopoulos, dimitrios.tzovaras}@iti.gr,
https://www.iti.gr/iti/index.html

***Abstract*—In the quest to improve the energy requirements of LPWAN nodes and make them more suitable for energy harvesting, a microcontroller with on-chip Ferroelectric Random Access Memory (FRAM) was used as a controller in a LoRaWAN node. Energy measurements showed that the performance of such a device is comparable or better than that of a similar FLASH-based microcontroller. Furthermore, the advantages resulting from the high endurance and low-power characteristics of FRAM memories can be used to improve the node.**

Keywords—LPWAN; LoRaWAN; FRAM; Low power; Energy Harvesting; LoRa

I. Introduction and Problem Statement

Energy autonomy and cost are among the issues slowing down the adoption of Low Power Wide Area Networks (LPWAN). In many cases, such nodes should last for tens of years and regularly or occasionally collect and transmit information. Installing billions of nodes to that purpose suggests affordable devices both at manufacturing and purchasing time, but also low operational and maintenance costs. Considering the reliability of electronic devices, maintenance costs can be linked to the durability of the energy source (how long the batteries will last). In order to extend the use of IoT nodes, it is therefore important to address issues such as energy and cost. This is especially true when the nodes are difficult to access, meaning that maintenance operations would increase costs during the product's lifetime. In some cases, it might even be difficult to remember where those nodes have been installed and thus laborious to find them again for any maintenance purposes. Consequently, the battery life of such systems should be as close as possible to their useful life. Energy harvesting has often been considered as an alternative to batteries. For both cases, the reduction of energy requirements is critical if nodes are to last for tens of years, doing useful work.

The adoption of energy harvesting has been plagued by issues such as cost (compared with batteries) and the need to make the system resilient to possible energy outages. Harvesters and their supporting components such as boosters, power management and storage elements all add to the cost and complexity of the system. For instance, if solar cells are used as energy harvesters, one may want to store energy during the day in case some work should be done at night. The energy requirements of LPWAN nodes are much higher than those of short-range nodes, putting an extra cost burden on such systems.

The use of power modes is a key approach in reducing energy consumption. The system works as efficiently as possible, transmits and receives data and then goes into the lowest possible energy consumption mode that fits the application. In that mode, energy requirements are low and constraints on the power management are reduced, meaning that the energy balance is easier to maintain. Due to regulations, some popular LPWAN systems such as Sigfox or LoRaWAN tend to sleep (or pause) for several minutes or even hours between communication activities. Transmission is often the phase that requires the most energy.

This work[1] investigates the use of a microcontroller with on-chip Ferroelectric Random Access Memory (FRAM) as the main controller of a LoRaWAN node. Although more expensive than FLASH, FRAM offers some interesting advantages that could help reduce energy consumption (especially in inactive states) and improve a system's behaviour when energy outages occur. The FRAM node is compared to a FLASH-based node. It is concluded that it competes successfully and even outperforms the FLASH-based system in some cases.

II. State of the Art

Currently, there is not a significant amount of published works that describe or evaluate the use of

[1] This work is the result of an internal project of the ZHAW Institute of Embedded System with links to two funded projects:
The support of Innosuisse for certain aspects of the work (grant 31354.1 IP-ICT) is acknowledged.
The work is also related to the AMANDA Project sponsored by the Horizon2020 program under the grant agreement ID: 825464.
ZHAW and CERTH are part of the AMANDA Consortium.

FRAM in Wireless Sensor Networks (WSNs), especially LPWAN/WPAN. This is possibly due to the limited incorporation of FRAM memories in microcontrollers or as single IC component in embedded systems. Few manufacturers have commercially available FRAM chips and there are even less who fabricate microcontrollers with FRAM as their main memory. However, there is some recent research activity in which MCUs with incorporated FRAM have been used in self-powered IoT applications [1], [2], [3], [4].

The MSP430 by Texas Instruments has a variant that includes FRAM. When there is not enough power for the completion of the running routine in the MCU or transmission of data, the current state of the MCU can be saved in FRAM and then restored when the system's power returns to operational levels. The energy required to store data in FRAM is significantly low. Moreover, the MSP430 offers low-power modes, namely the LPM3 and LPM3.5, in which the retention of data in FRAM is available. During these states, the MCU is almost completely shut down. FRAM's non-volatility makes possible the complete shutdown of the MCU without a loss of critical data.

Meli and da Silva investigated the power saving options for a ZigBee-based WSN, powered by utilising energy harvesting techniques [1]. FRAM was used to save and restore critical parameters in battery-less ZigBee nodes, especially in the case of nodes powered by electrodynamic harvesters. It was concluded that the incorporation of FRAM, either as part of the microcontroller or as discrete component, could be a solution to the reduction of energy requirements in ZigBee applications powered by energy harvesters.

Similarly, de la Rosa et al. used the MSP430 microcontroller and exploited the above-mentioned low-power modes in order to keep the overall power consumption at the lowest levels and increase its maximum operational time [2]. In this work, the sensor node included the XBee RF module, based on the IEEE 802.15.4 communication protocol. A dynamic power strategy, switching between the sleep and standby modes of the MCU was used to minimise the non-essential power consumption.

A WPAN-based network architecture was fabricated by Purkovic et al. by pairing the MSP430 MCU with the C1120 RF chip [3]. The whole system was powered using solar energy and was operated autonomously for a whole year. Additionally, modifications were made in the communication protocol to achieve an even lower energy consumption.

Apart from WPANs, a major role in the WSN field of study is appertained to LPWANs, such as LoRa or Nb-IoT. Towards this direction, Magno et al. combined the MSP430 with the SX1276 LoRa module and the WuRx, a wake-up radio transceiver, in order to achieve a significantly low power consumption during sleep or standby mode [4].

III. Objectives

The work described in this paper is part of a long-term activity that aims at providing several paths to easily integrate energy harvesting in LPWAN nodes. The combination of new non-volatile memory devices with other technologies should allow in the long run, the implementation of cost competitive nodes capable of working for tens of years on harvested energy. This work is a step towards that long-term objective, with the following purposes:

- To design a LoRaWAN node based on an FRAM microcontroller and to use it to verify that its energy performance is comparable or better than that of nodes on the market that use conventional NV-memory elements (FLASH for code and SRAM for data).
- To generate enough measurement data in order to allow reliable predictions of the energy requirements of such a node.

IV. Outline of the Proposed Solution

A LoRaWAN node based on a commercially available FRAM microprocessor was designed, its energy performances measured and compared to those of a LoRaWAN node controlled by an ARM Cortex-M0+ compatible device. Based on the measurements made, the lifetime of the node was calculated for several scenarios. The MSP430FR5994 [5] of Texas Instrument was chosen as FRAM microcontroller. The appropriate development kit (MSP-EXP430FR5994) [6] was connected over a serial link to the LoRa transceiver SX1261 from Semtech on a SX1261MB2BAS board [7]. The BME680 [8] of Bosch was used as sensor. The different parts of the design are shown in Fig. 1.

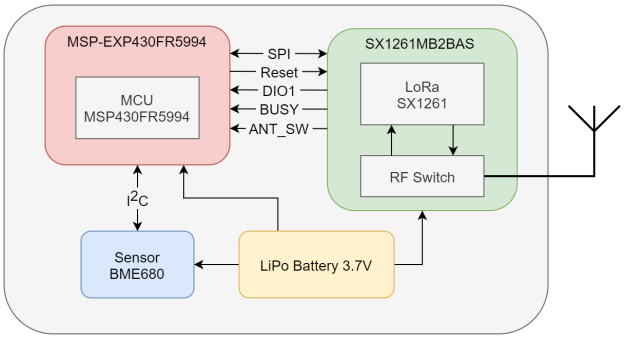

Figure 1. Different parts of the design.

With 256kB FRAM and 8kB SRAM, the 16-bit microcontroller has enough memory to accommodate the LoRaWAN stack and the application. The MSP430 has less computing resources than 32-bit devices. This is however enough for some applications.

The reference solution is the LoRa Discovery Kit B-LO72Z-LRWAN1 of STMicroelectronics [9]. It features an STM32L072CZ (Cortex-M0+) microcontroller [10] and an SX1276 transceiver [11]. The microprocessor has

up to 192KB FLASH, 20KB SRAM, 6KB EEPROM. A LoRaWAN stack is also provided with several examples. Sensing is achieved with a NUCLEO-IKS01A2 board.

The SX1276 requires slightly more energy that the SX1261 when the latter is used in DC/DC mode.

- 9.9 mA receive current for the SX1276 at 3.3 V
- 4.6 mA receive current for the SX1261 at 3.3 V
- 25.5 mA transmit current at 3.3 V and +14 dBm for the SX1261
- 29 mA transmit current for the SX1276 at 3.3 V and +13 dBm

These differences are reflected in the energy requirements of the nodes when the transceiver's energy is dominant. Long intervals between the communication operations reduce the differences and amplify the energy consumption in the low-power modes.

V. FIRMWARE

The open source LoRaWAN stack "loramac-node", available on GitHub, was ported to the MSP430. As shown in Fig. 2, the LoRaWAN stack is built up from different parts. It consists of the MAC layer, the radio module driver, the encryption unit and additional modules such as a time server. The SX1261 driver is part of the LoRaWAN stack and controls the module using the SPI interface and some GPIOs.

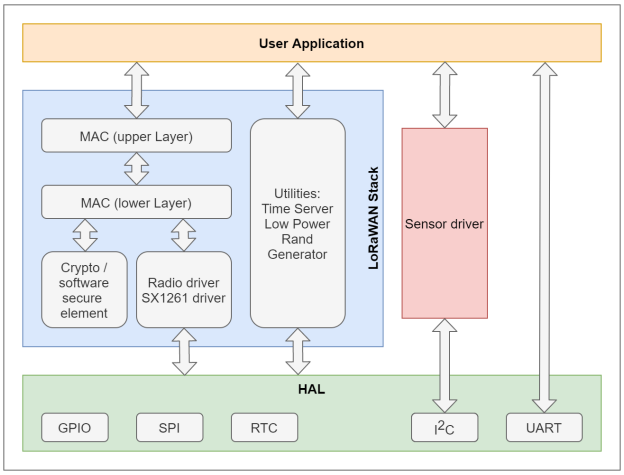

Figure 2. Architecture of the firmware.

The MAC layer controls the physical layer using the time server. The time server provides functions to start or stop tasks. For example, monitoring that the permitted duty cycle of the uplinks is adhered to. The MAC also performs AES encryption for the MAC header and user data. To make it easier to address the LoRaWAN Class A state machine at an application level, the MAC has been divided into a lower and an upper layer. All MAC functions are implemented in the lower layer. The upper layer provides a simpler interface to register application packages and have them executed by the MAC.

The HAL is responsible for controlling the hardware required by the application. The RTC has a central task as a time unit, which continues to run even in sleep mode. Interrupts of the RTC are used to activate the system at programmed times.

The driver for the BME680 sensor contains all necessary functions to communicate with the sensor. The access takes place directly from the application. The sensor driver controls data exchange with the sensor via the I^2C interface.

The application is built in an endless loop. It manages the low-power handling, the interrupt handlers (GPIO or RTC) and calls the LoRa Class A routines when associated tasks are to be completed. The UART interface is accessed directly from the application for debug purposes. The application has essentially two types of operation, event or time controlled. Fig. 3 shows the basic application procedure. When the application is time controlled, the sensor data is periodically sent to the LoRa network. During the transmission pauses, the MSP430 is always in LPM3 with the RTC running. The event-based application type changes to LPM4.5 mode after a successful join process, whereby the entire microcontroller is switched off. The microcontroller is woken up by an external interrupt, transmits a packet with sensor data to the LoRa network and switches to LPM4.5. Before entering LPM4.5, the entire RAM content is copied to a sector in FRAM, because the RAM content will be lost in LPM4.5. After waking up from LPM4.5, the RAM content is restored from FRAM and the application can continue running with the previous initialisation.

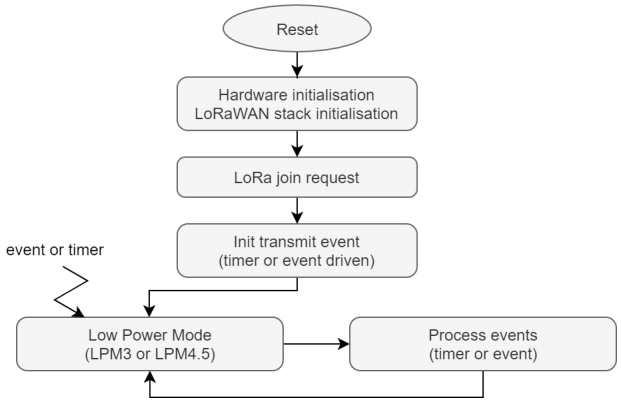

Figure 3. Simplified application flow.

For the reference node, the stack delivered by the manufacturer was used, including the available timer functions.

VI. MEASUREMENTS, RESULTS AND DISCUSSION

In order to evaluate the energy performance of the nodes, basic energy measurements were made. The nodes were programmed to communicate with the server in SF7BW125 (referred to in this work as SF7) and in SF12 modes. In both cases, the maximum output power for Europe was set (+14 dBm) and a payload of 20 Bytes was

used. SF7BW125 has a higher bit rate, resulting in the shorter airtime and therefore a lower energy requirement while transmitting. The range is also the shortest. In SF12, the bit rate is low, thus the frame transmitted requires the highest airtime. This results in higher energy requirements.

Two use cases were considered:

- A first case where the nodes are regularly woken-up by a timer in order to measure and transmit.
- A second case where the device shuts down after the active part and can only be woken up by an external interrupt. In such a case, the timer is not active and there is no variable retention in SRAM. This works well for the microcontroller with FRAM, since it can keep crucial parameters in the FRAM and rewrite those memory positions billions of times (important for long life).

  In a variation of the second case, the FRAM microcontroller can be completely powered off and restarted only when there is enough energy. This emulates the use of harvested energy, when there is not enough energy to keep the system powered. The node is subsequently restarted when the power management detects that enough energy is available.

In order to perform energy measurements, a power analyser was used. The voltage and the current were then recorded according to the setup shown in Fig. 4. The microcontroller was connected to the peripherals using serial links. The energy of the microcontroller and of the radio transceiver were measured separately. A similar set-up was used for the STMicro LoRaWAN kit.

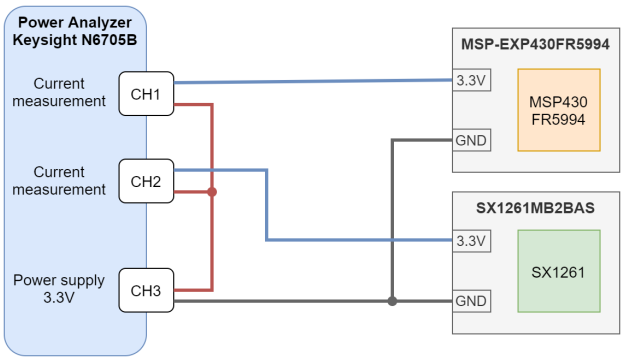

Figure 4. Set-up to measure the dynamic power profile of the nodes.

The following figures (Fig. 5, Fig. 6, Fig. 7, Fig. 8) show the recorded current profiles. The power profiles can be deduced by a multiplication with the voltage, which is constant in these cases. The transmission airtime is represented by the transceiver transmit time. The MSP430 runs at 8 MHz and requires more time for computing activities than the STM Microcontroller that runs at 32 MHz.

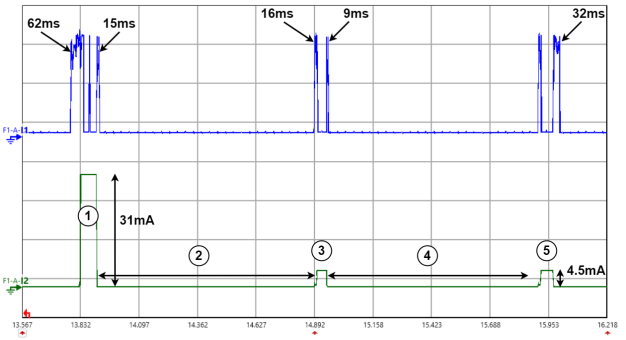

Figure 5. The upper trace shows the activity of the MSP430 microcontroller and the lower trace shows the activity of the transceiver in SF7 mode. (2.25 seconds in total in SF7. 3.99 seconds in SF12).

In Fig. 5, the activities of the MSP430 are shown in parallel with those of the transceiver. The system works in SF7. After initialisation, the uplink frame is sent (1). The system then switches to low-power mode (2 and 4) with the timer/RTC active to determine when the receive windows should be activated (3,5). The microcontroller activities around those windows are linked to the CPU accessing the transceiver in receive mode. The transmit time (1) is around 72 ms. The receive time (3) is 42 ms. The time between the end of transmission and the receiver activation (2) is close to 1000 ms. The interval between the 2 receive windows is about 960 ms. These times are comparable for both nodes, since they are defined by the LoRaWAN protocol. The transmission time depends on the payload size and the spreading factor used. The receive window time varies with the size of the message from the gateway to the node (downlink).

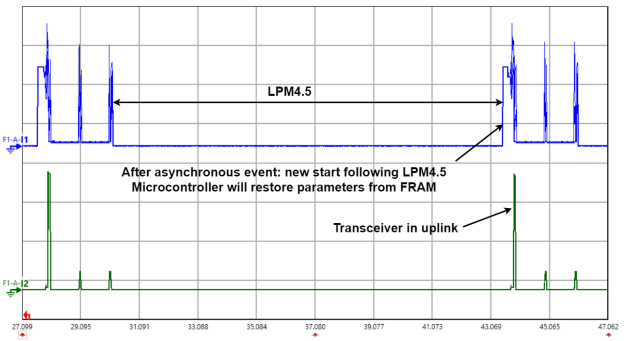

Figure 6. Asynchronous events for MSP430 node.

Fig.6 shows 2 consecutive node activities with saving and restoring of parameters. The FRAM is used to save parameters before the device is shut down. Between the 2 frames, the node goes in lowest possible power mode. It is restarted by an asynchronous event. In this case, the microcontroller needs about 80 nA in LPM4.5. The transceiver requires about 270 nA at the same time. In a variation where the node is switched off, both elements will require 0 nA. The microcontroller needs about 300 ms to restart, copying the data from FRAM and initialising the system.

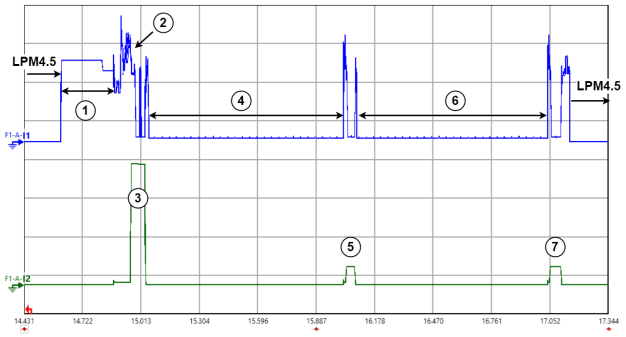

Figure 7. Restart by asynchronous event.

Fig. 7 shows with more details what happens when the node is woken up by an asynchronous event, after the parameters were saved in the FRAM. The microcontroller wakes up from LPM4.5 following an interrupt and copies the data from FRAM to RAM (1). After initialising the peripherals an performing the sensing operation, the uplink frame is sent (2 and 3). The system then switches to low-power mode (4 and 6) with the timer/RTC active to determine when the receive windows should be activated (5 and 7). After waiting for an eventual downlink message in both receive windows, the system saves parameters in FRAM and switches to LPM4.5 again. The whole process requires 2.55 seconds in SF7 (4.32 seconds in SF12)

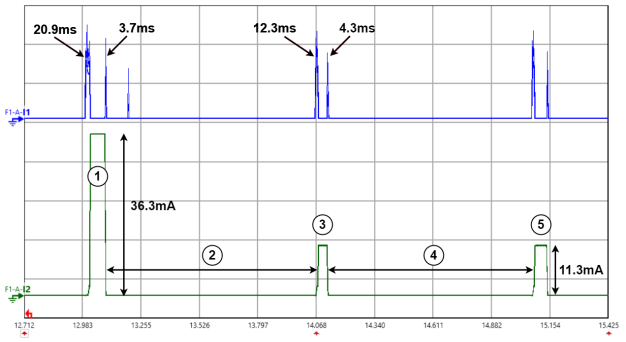

Figure 8. STM node. The upper trace shows the activity of the M0 microcontroller and the lower trace shows the activity of the transceiver in SF7.

Fig. 8 shows the profile of the STM node. After initialisation, the uplink frame is sent (1). The system then goes into low-power mode (2 and 4) with the timing resources active to determine when the device should go in receive mode (3,5). The microcontroller activities around the receive windows are linked to the CPU accessing the transceiver. Higher currents in transmit and receive modes, compared to the node with the MSP430, are due to the use of the SX1276 which draws more current that the SX1261.

Based on the measurements made in SF7 and SF12 (some of them shown above) and several other measurements, the energy consumptions of the nodes were computed. The battery lifetimes were estimated, assuming an energy source of 2000 mAh and 288 uplinks per day. The results are shown in Table 1 below.

TABLE I. LIFETIME CALCULATIONS

Node	Active period (TX and RX)			Saving mode		
	Duration (s)	Meas Energy Micro (µJ)	Meas energy Radio (µJ)	I system (nA)	I Radio(nA)	Life time in years
STM in SF7. RTC wake up	2.15	2'130.00	12'570.00	1'500.00	234.00	13.78
STM in SF12. RTC wake up	3.91	2'140.00	225'780.00	1'500.00	234.00	0.98
MSP430 in SF7. On Chip Timer wake up	2.25	728.50	9'010.00	40'000.00	234.00	4.59
MSP430 in SF12. On Chip Timer wake up	3.99	1'020.00	184'670.00	40'000.00	234.00	1.00
MSP430 in SF7. Outside event wake up	2.55	1'450.00	8'900.00	80.00	234.00	21.21
MSP430 in SF12. Outside event wake up	4.32	1'700.00	176'800.00	80.00	234.00	1.26
MSP430 in SF7. Power off Restart when energy	2.55	1'450.00	8'900.00	0.00	0.00	21.84
MSP430 in SF12. Power off Restart when energy	4.32	1'700.00	176'800.00	0.00	0.00	1.27

The first column shows the node. STM for the reference node based on the STMicro kit, MSP430 for the node with FRAM. The spreading factor (SF7 or SF2) is added. The wake-up mode that is assumed in the energy and lifetime calculations is also indicated.

- RTC: the RTC peripheral is used for the STM.
- On-chip timer: the method that is used for the timing in the MSP430 for synchronous timing
- Outside event: in this case, the device goes in the lowest possible sleep mode and is woken up by a very low-power external timer. This allows a lower consumption for the MSP430, compared to the use of the internal timer.
- Power off: the MSP430 device is powered off after saving important data in the FRAM. No energy is consumed in power off. The device is restarted after power is applied again.

The second column shows the duration of the activity of the node, as measured in SF7 or SF12.

The third and fourth columns are the energy required for the activity, for the controller part and for the transceiver part.

The fifth and sixth columns are the currents for the chosen saving modes. Respectively for the controller and for the transceiver.

For a given number of frames per day, the interval between frames is calculated and assumed to be a time when the node is in low-energy mode.

The calculations show that:

- The STM node performs better when synchronous events (timer) are used to wake-up the processor. This is because the MSP430 CPU was combined with its RTC to deliver the timing accuracy needed, leading to a higher energy consumption in sleep mode. This can be improved by using a better (external) RTC for the MSP430 node. There is also room for improving the performance of the STM kit by optimising the use of the RTC or by using an external RTC that requires less energy.

- In asynchronous mode (outside event), the MSP430 can go into a very low current mode, leading to a better performance when inactive times are long.
- The last two rows (power off) reflect the case when the MSP430 would be switched off. This will lead to a slightly improved performance.

It can be concluded that the use of the MSP430 microcontroller with FRAM resulted in a node with a performance comparable or better than the reference node it was compared to. Furthermore, the advantages of the FRAM can be used to enhance the energy requirements of the node, especially for energy harvesting.

It is important to note that when using batteries, the practical lifetime also depends on the battery type and the conditions of use, according to the manufacturer's specifications. The values computed here are theoretical values that should be combined with the boundaries given by the manufacturer. For instance, if a battery is guaranteed only for 7 years, that value will limit the lifetime of the node. Such limitations also amplify the case for energy harvesting.

VII. Conclusions and Future Work

This work has shown that a LoRaWAN FRAM microcontroller has the potential of outperforming FLASH-based devices in some applications. Thanks to the properties of FRAM, energy reduction can be maximised in power saving modes. The device can even be totally switched off. Saving important parameters in FRAM allows a reboot without fear of losing critical data or status. Furthermore, the FRAM is a better solution in terms of energy and number of read/write cycles, when these parameters are important for the application over many years. The use of FRAM also provides other options when the node is powered by energy harvesting. For instance, it opens the door to designing LPWAN nodes that can easily recover from power outages and thus enable applications that can last for tens of years.

In a future work, the node will be further developed by adding energy harvesters and associated power management in order to achieve a completely energy autonomous system and further reduce the limitations related to the use of batteries. Other wake-up sources will also be added to increase the flexibility of the system. Similar nodes will be built for other LPWAN systems. The effective implementation and verification of a state machine that can be frozen upon energy outage and restarted when enough energy is available will also be accomplished.

References

[1] M. Meli, and M. da Silva, "Battery-less sensor nodes for 802.15.4/ZigBee wireless networks," *Wireless Congress*, Munich, November 2011

[2] E.O. de la Rosa, J.V. Castillo, M.C. Campos, G.R.B. Pool, G.B. Nuñez, A.C. Atoche and J.O. Aguilar, "Plant microbial fuel cells–based energy harvester system for self-powered IoT applications," *Sensors*, vol. 19, pp. 1378, 2019

[3] D. Purkovic, M. Hönsch, and T.R.M.K. Meyer," An energy efficient communication protocol for low power, energy harvesting sensor modules," *IEEE Sensors Journal*, vol. 19, n. 2, pp. 701-714, 2019

[4] M. Magno, F.A. Aoudia, M. Gautier, O. Berder and L. Benini, "WULoRa: An energy efficient IoT end-node for energy harvesting and heterogeneous communication," *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2017

[5] Information page of microprocessor MSP430FR5994. [Online]. Available: https://www.ti.com/product/MSP430FR5994 [Accessed: Jul. 27, 2020].

[6] MSP430FR5994 LaunchPad development kit. [Online]. Available: https://www.ti.com/tool/MSP-EXP430FR5994 [Accessed: Jul. 27, 2020].

[7] Information page about SX1261 LoRa transceiver and SX1261MB2BAS board. [Online]. Available: https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1261 [Accessed: Jul. 27, 2020].

[8] Product information of BME680. [Online]. Available: https://www.bosch-sensortec.com/products/environmental-sensors/gas-sensors-bme680/ [Accessed: Jul. 27, 2020].

[9] Information on LoRa Discovery kit B-L072Z-LRWAN1. [Online]. Available: https://www.st.com/en/evaluation-tools/b-l072z-lrwan1.html [Accessed: Jul. 27, 2020].

[10] Datasheet of STM32L072CZ microcontroller. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32l072v8.pdf [Accessed: Jul. 27, 2020].

[11] Information on SX1276 LoRa transceiver. [Online]. Available: https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1276 [Accessed: Jul. 27, 2020].