Demonstration of SAMD21 Development Kit use in IoT through Bluetooth Low Energy (BLE) communication protocol

Felipe Seitenfus and
Leandro Frazzon Saccol
Center of Technology
Federal University of Santa Maria
Santa Maria, Brazil
Email:{felipe.seitenfus, leandro.saccol}
@ecomp.ufsm.br

Carlos Henrique Barriquello
Dept. of Electronics and Computing
Federal University of Santa Maria
Santa Maria, Brazil
Email: barriquello@gmail.com

***Abstract*—Technologies aimed at Internet of Things (IoT) are object of interest to commercial companies and academic research. Nowadays, there are already several hardware and software projects with the purpose of integrating different types of devices, cheaply and effectively. This paper intends to demonstrate the use of SAMD21 development kit, from Microchip Technologies, applied to IoT with the Bluetooth Low Energy (BLE) communication protocol. This will be done showing, as an example of use, the interaction capacity between this embedded system and an external device, such as a mobile phone.**

***Index Terms*—Bluetooth, BLE, Embedded System, IoT**

I. Introduction

BLE Technology, also known as Bluetooth Low Energy or Bluetooth Smart, is a wireless personal area network technology conceived and marketed by the Bluetooth Special Interest Group®. If we compare BLE with conventional Bluetooth, BLE is designed to provide power consumption and reduced costs, with an equivalent, though not very stable, range of communication. BLE can be considered superior to other location-based technologies for smartphones because it is more focused [1]. As the most significant disadvantages, BLE technology, as average Bluetooth, has low transmission range and little security, being very vulnerable to attacks [2].

A BLE device remains in *Idle* mode, also called *sleep mode*, for most of the execution time. This happens because BLE devices are usually designed for applications that need to send little information, seldom leaving idle mode to make connections whose duration lasts about a few milliseconds. Thus, energy consumption has peaks of 15mA, for example, but with an average of only 1μA [3]. Tab. 1 shows the main differences between average Bluetooth and BLE.

As can be seen, BLE technology has smaller power consumption and a bigger battery lifetime. This makes it more suitable for dedicated or continuous-use devices, which need to have a low energy consumption, but also should be constantly - or at least during a long time period - with active connections, differently from typical devices with average Bluetooth. The fact the possible network size is undefined also makes network system projects more flexible, with possibilities of network rearrangement, as such as increase or decrease of nodes.

TABLE I
Comparison between average Bluetooth and BLE technologies [1]

	Average Bluetooth	**BLE**
Physical Layer	IEEE 802.15.1	GFSK
Frequencies	2.4 GHz	2.4 GHz
Maximum Bit Rate	1-3 Mbps	1 Mbps
Maximum Distance	10-100 meters	50 meters
Power Consumption	High	Very Low
Life Battery	Days	Months to Years
Network Size	max. 7 devices	Undefined

A device equipped with BLE technology opens up several possibilities. In practical exemples, we can highlight devices such as thermometers, speedometers, clocks, pressure, level, presence or proximity sensors, medical devices such as cardiac, physical activity, blood pressure and glucose content monitors, as well as applications such as remote controls and industrial automation [4]. BLE technology also offers the option to add functionalities to smart devices. A smartphone with BLE, for example, can serve as an intermediate device, or a gateway, and provide something like a communication tunnel to any device, making Internet access viable [4].

This paper's purpose is to demonstrate the use of BLE technology applied to IoT, with a SAMD21 development kit integrated with the ATBTLC1000 peripheral, both owned by Microchip Technologies and donated by the company itself to Federal University of Santa Maria. This will be shown by running a program in the C language, containing a series of subroutines that deal with the communication between the embedded device and a second BLE device, in this case, a cell phone. The results of the interaction between them will be analyzed through a terminal on a computer connected to the board.

II. Hardware Components

A. SAMD21 Development Kit

The SAMD21 development kit was originally manufactured by *Atmel Corporation*, today called Microchip Technologies®, in the year of 2014. Using the ATSAMD21J18A microcontroller, its organization allows not only the coupling of several peripherals, but also makes easier the understanding of its functions [5].

Among the possible applications, there are audio processing, graphing, encryption, Pulse Width Modulation (PWM) and BLE communication. For data storage, the kit emulates an EEPROM-type memory, used to store small amounts of data that must be saved when power is cut [6]. Tab.2 makes a brief comparison between ATSAMD21J18A microcontroller, which is used in SAMD21 development kit, and AVRATmega328, a microcontroller with similar purpose, used in Arduino Uno [8].

TABLE II
Comparison between ATSAMD21J18A and AVRATmega328 [7] [8]

	ATSAMD21J18A	AVRATmega328
Operation voltage	3.3V	5V
I/O Electric Current	40mA	10mA
Clock Frequency	48MHz	16MHz
Flash Memory Size	256KB	32KB
Protection against over-current	No	Yes

In this paper, a version of SAMD21 kit called *Xplained Pro* was used, provided by the company itself to the Federal University of Santa Maria, through donation, as previously stated. The Xplained Pro version makes good programming practice possible, used by several embedded systems manufactured by Microchip [5], even though it is a compact version.

This version supports a wide range of extension cards, which can be connected to the SAMD21 by standard headers and connectors. It also allows the installation of serial communication drivers on a PC, supporting data transmission between two or more cards, or between the card and the computer. In addition, it provides sample codes that demonstrate its features, compiled through the *Atmel Studio* programming environment and executed in the kit [4]. Fig. 1 shows a top view of the SAMD21 boarder.

B. ATBTLC1000 Peripheral

The ATBTLC1000 peripheral is a transceiver device (enabled for transmission and reception) of BLE signals. It is a ARM Cortex-M0 type microcontroller chip, integrated with a modem, a medium access control layer (MAC), a power amplifier, a duplexer and a power management unit (PMU). It can be used as a BLE transmission link or data pump with the microcontroller from an external server [9].

ATBTLC1000's Bluetooth Smart protocols are stored in a dedicated ROM memory. Its firmware includes service layer protocols L2CAP, Security Manager, Attribute Protocol (ATT), GATT (Generic Attribute Profile) and GAP (Generic Access

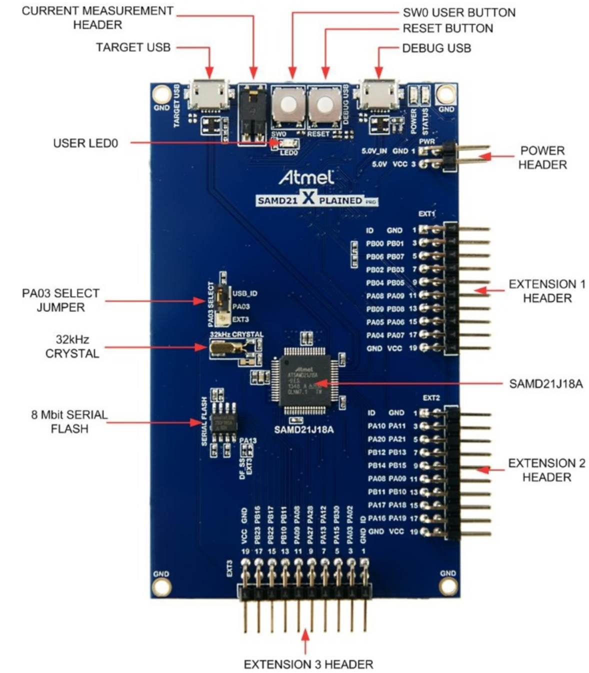

Fig. 1. SAMD21 development kit, *Xplained Pro* version [5].

Profile). Supported applications, such as the demonstration of signal transmission studied in this paper, are also included in the protocol stack [9]. Fig. 2 shows a top view of the ATBTLC1000 peripheral.

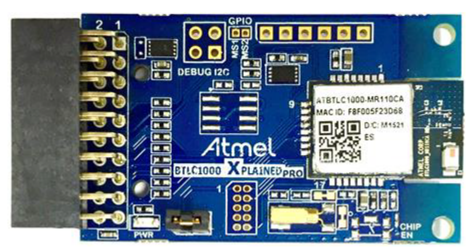

Fig. 2. ATBTLC1000 Peripheral [6].

III. Software Component

A. Structure and Methodology

The source code used in the paper was based on one of Microchip's examples, called FIND_ME_XPLAINED_PRO [10]. It configures SAMD21 and ATBTLC1000 to receive and transmit BLE signals of three different strengths, enabling communication with an Android® smartphone through the Atmel SMART Connect® application. The code was restructured by the authors in protothreads, which are threads designed for embedded systems or nodes of wireless sensor networks with restricted memory. This code structure provides sequential flow control without complex state machines or full multi-threading [11].

The development methodology used was Test Driven Development (TDD). Each new feature starts with a test, which should fail because it is written before the functionality to be implemented. To write a test, the developer must clearly understand the specifications and requirements of the functionality. TDD makes the developer focused on the requirements before the code, which is a subtle but important difference [12].

B. Code Description

This subsection has a description of the most relevant parts in terms of configuring and executing of the C source code executed by SAMD21. The complete code is available in a GitHub repository [13]. Beyond the re-structuring, the source code was optimized and instructions for recording and reading of EEPROM memory were also inserted in the C code.

The *main* function initializes the program and executes the protothread through the PT_INIT () function. The program performs 10 steps, controlled by a counter and a loop. Each increment of the counter indicates the step that must be performed by the program on protothread. Fig. 3 shows the flowchart of a successful program execution.

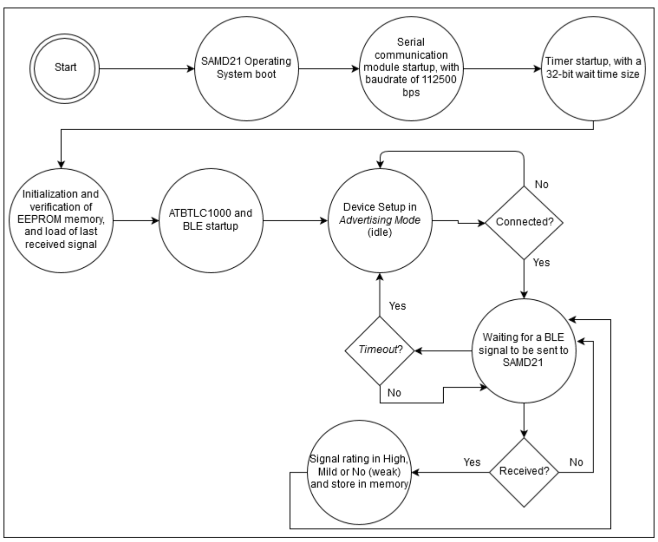

Fig. 3. Flowchart of the program's step-by-step execution.

Inside the protothread, the ATSAMD21J18A microcontroller on SAMD21 is initialized. If the initialization occurs without problems and if the return conditions are checked, the serial communication module of the shipment is initialized with a baud rate of 115200 bits per second. The timer is initialized with a maximum 32-bit wait time for a successful connection since the BLE signal is unstable.

When the ATBTLC1000 peripheral is initialized, the program loads from the memory the last signal sent to the board during the last well-succeeded communication. The device is then placed in *Advertising Mode* (idle), until a cell phone is successfully connected. If this connection is successful, the SAMD21 starts waiting for a signal to be sent. When this happens, it is classified according to its intensity. If the time limit is reached without a device sending a signal, it returns to Advertising Mode, forcing a new connection attempt.

IV. Practical Tests

A series of twenty tests were carried out over two weeks to verify the viability of the algorithm developed. Each test started by configuring the TeraTerm serial terminal with a baudrate of 115200 bits per second. On the cell phone, the kit was tracked and paired by Atmel SMART Connect to make the connection. If pairing was successful, the cell phone would send signals to the SAMD21 that were shown on the terminal. The software was tested on three computers, with a Samsung Galaxy SM-G316ML® smartphone. Fig. 4 shows the SAMD21 kit connected to a computer.

Fig. 4. SAMD21 with the ATBTLC1000 peripheral connected to a computer.

During the tests, it was verified in stopwatch that the BLE signal took in average 12 seconds to be effectively paired by the ATMEL Smart Connect on the smartphone. With this, the test became difficult, as several times the time limit was reached. At other times, the connection was abruptly interrupted during signal detection. In both situations, new connection attempt was the only solution. The distance between the kit and the smartphone was also verified but didn't have significant influence. The example code also doesn't make clear the used power value or consumption during the operation, with this data being abstracted from the programmer.

Each time a connection was successful, the signals sent by the cell phone have been shown on the TeraTerm terminal and recorded in the EEPROM memory as soon as detected. When the program was restarted, the last received signal was shown on the screen. In Fig. 5, taken from one of the last tests, a series of signals received by SAMD21 are shown, and it is also shown that the last received signal was a *Mild* signal. When the program was restarted then this *Mild* signal was read from the EEPROM and shown on the terminal, as shown in fig. 6.

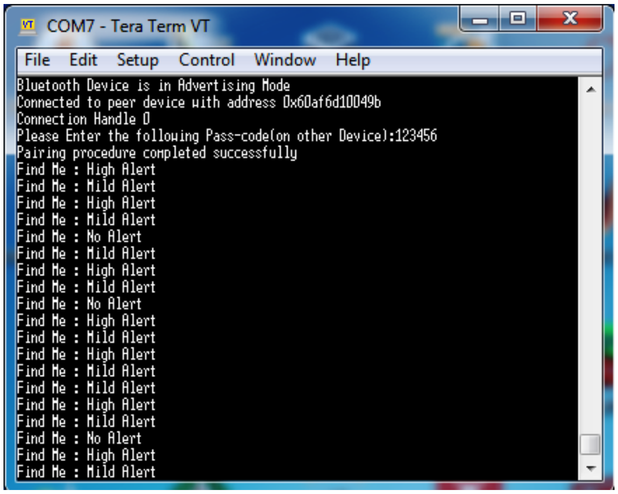

Fig. 5. Signals received by SAMD21 displayed on the terminal.

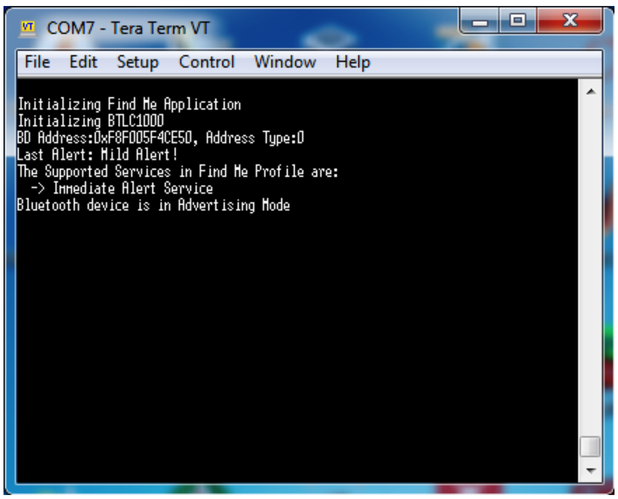

Fig. 6. Following program execution, showing the last signal saved.

In a general way, the results have remained the same. Only small variations in connection time were detected, but within the average.

V. Final Considerations

The SAMD21 development kit, together with the ATBLTC1000 peripheral, was successful in the task of receiving and handling BLE signals. The performance of the TeraTerm program was also satisfactory, and the EEPROM memory functioned effectively.

The biggest problems came from the time it took for a connection to be made. This time does not depend on the developed source code but rather on the smartphone, making the data collection slower, since there is no way to change it. Another difficulty found during the research and during this paper's writing was due to the few projects or papers developed in the SAMD21 available in the Internet, probably due to the little diffusion of this and other embedded systems of the same family. This also made the comparison with work already done with the same hardware difficult.

Despite that, SAMD21 is a useful and functional embedded system, following the current demand for wireless and IoT devices in general. Then, this paper can be used as a stimulus for further research using SAMD21 and its functionalities.

Acknowledgment

The authors would like to thank the Nucleus of Research and Development in Electrical Engineering (NUPEDEE) of the Federal University of Santa Maria.

References

[1] L. Pessoa, "Introdução ao Bluetooth Smart (BLE)," *Embarcados*, 2016. [Online]. Available: https://www.embarcados.com.br/bluetooth-smart-ble. [Accessed: 15-Apr-2018].

[2] P. M. G. Alves, "Funcionamento, vantagens e desvantagens das diversas tecnologias," *Instituto Politécnico de Tomar*, 2012. [Online]. Available: https://alumni.ipt.pt/pmgalves/Funcionamento_Tecnologias.pdf.[Accessed: 16-Jul-2018].

[3] A.S de Araújo and P. de Vasconcellos, "Bluetooth Low Energy," *Universidade Federal do Rio de Janeiro*, 2012. [Online]. Available: https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2012_2/bluetooth/ble.htm. [Accessed: 15-Apr-2018].

[4] J. Rochol, "Bluetooth 4.0 ou Bluetooth Low Energy (BLE)," in *Sistemas de Comunicação Sem Fio: Conceitos e Aplicações*, Porto Alegre, Brazil: Bookman, 2018, ch. 4, pp.128.

[5] Atmel Corporation, "Sam D21 Xplained Pro User Guide," *Microchip Technologies*, 2016. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42220-SAMD21-Xplained-Pro_User-Guide.pdf. [Accessed: 16-Apr-2018].

[6] Arduino AG, "EEPROM Library," *Arduino Reference*, 2018. [Online]. Available: https://www.arduino.cc/en/Reference/EEPROM. [Accessed: 16-Apr-2018].

[7] G. Kenshima, "Review Arduino M0 x Arduino M0 Pro," *Embarcados*, 2016. [Online]. Available: https://www.embarcados.com.br/review-arduino-m0-e-arduino-m0-pro/. [Accessed: 16-Jul-2018].

[8] Atmel Corporation, "Atmel SAM D21E / SAM D21G / SAM D21J Datasheet," *Microchip Technologies*, 2016. [Online]. Available: https://cdn.sparkfun.com/datasheets/Dev/Arduino/Boards/Atmel-42181-SAM-D21_Datasheet.pdf. [Accessed: 16-Jul-2018].

[9] Atmel Corporation, "ATBTLC1000 Xplained Pro User Guide" *Microchip Technologies*, 2016. [Online]. Available: http://ww1.microchip.com/downloads/en/devicedoc/Atmel-42538-ATBTLC1000-Xplained-Pro_UserGuide.pdf. [Accessed: 16-Apr-2018].

[10] Microchip Advanced Software Framework, "FIND ME example," *ASF Source Code Documentation*, 2016. [Online]. Available: http://asf.atmel.com/docs/latest/thirdparty.wireless.ble_sdk.apps.find_me.samd21_xplained_pro/html/index.html. [Accessed: 16-Apr-2018].

[11] A. Dunkels, "Protothreads," *Swedish Institute of Computer Science*, 2005. [Online]. Available: http://dunkels.com/adam/pt/. [Accessed: 18-Apr-2018].

[12] M. Aniche, "Test-Driven Development," *Caelum*, 2014. [Online]. Available: http://tdd.caelum.com.br/. [Accessed: 19-Apr-2018].

[13] F. Seitenfus and L. F. Saccol, "SAMD21_BLE," *GitHub*, 2017. [Online]. Available: https://github.com/FelipeSeitenfus/Dispositivo_Bluetooth. [Accessed: 21-Apr-2018].