ISSN 1870-4069

Energy Consumption of an Internal CRC Module in a Microcontroller

Mario Alberto Camarillo-Ramos[1], Roberto López-Avitia[1], Miguel Bravo-Zanoguera[2], Verónica Quintero-Rosas[1], Apolonio Castro-Corral Reyes[1], Andrea Magaly Alvarado-Álvarez [1]

[1] Tecnológico Nacional de México, ITMexicali, Mexico

[2] Universidad Autónoma de Baja California (UABC), Mexico

{mario.camarillo, avitia.roberto, veronicaquintero, reyes, a11490776}@itmexicali.edu.mx, mbravo@uabc.edu.mx

Abstract. In the current scenario with the Internet of Things looming on every aspect and activity of daily life, the need for the interconnection with information grids is inevitable. Information from one device to another, to others, is always happening and thus, are prone to transmission errors. To mitigate these errors, CRC can be used. The added error checking feat is also an issue since the device will need to code and decode the checksum. This research presents the energy consumption of a microcontroller with an internal Cyclic Redundancy Check module. Such microcontroller is the PIC24FV32KA302 from Microchip. An energy profile was used to determine the behavior of microcontroller using a CRC-16 configuration.

Keywords: energy consumption, microcontroller, CRC, cyclic redundancy check.

1 Introduction

With the raise of the Internet of Things the number of devices that communicate with each other over wireless mediums have increased. It is not limited to the Internet, it is also true for Bluetooth, Zigbee, and other communication schemes of wireless data transmission where different applications can be realized. Those applications have a wide spectrum of uses, ranging from parking spaces managers [1], which uses sensors to provide accurate information to users to select an empty parking slot, to aid in healthcare training for future surgeons [2].

In [3], an overview of how connected devices can achieve meaningful intelligent information is presented; it shows how different architectures and algorithms to implement this information exchange between such devices can be realized.

Whether the device is used for any application or if the information is intelligent, it must guarantee the integrity of such transmitted information. One technique used for this purpose is Cyclic Redundancy Check or CRC. In [4], this technique is used in the Internet of Things and in [5], such error correction technique is used in Bluetooth.

Another advantage of applying CRC to the transmitted information, is that by minimizing the errors in the receiver, the need for retransmission lessens, thus lowering the energy needed by the transmitter. But implementing CRC also generates a burden in energy consumption in the device, due to the information codification. This paper analyzes the energy consumption of an internal CRC module in a microcontroller using a 16 bit encoded message using CRC-16.

2 Background and Related Work

In data communications, where devices continuously exchange information in the form of bits, ones and zeros, errors are prone to happen. Whether those errors are caused by electrical distortion or signal attenuation, one way to reduce these errors is to use a checksum. This technique is used to detect errors in a data transmission and was first explored in [6], as a polynomial equation. The theory of operation is similar to that of a checksum but instead of using addition for the bits, division is used.

In [7], the algorithm is described as a number with the appended checksum divided by another fixed number. The division uses polynomial arithmetic which is simplified by applying the Boolean operation of Exclusive OR (XOR).

For instance, let C be the code checked with the algorithm, M the message (information), G the generator or the divisor (polynom), Q the quotient and R the reminder of the operation. One more characteristic is to append n zeros (X^n), shifted to the left or to the right to complete the CRC operations [8]. If a transmission is issued:

$$MX^n = [QG] + R, \tag{1}$$

$$C = MX^n + R. \tag{2}$$

Eq. 2 can be rearranged in the form of:

$$C = MX^n - R. \tag{3}$$

The receiver can be described as follows:

$$C = QG. \tag{4}$$

Combining Eq. 3 and 4:

$$MX^n = (QG) + R. \tag{5}$$

Using another combination of Eq. 3, 4 and 5:

$$(MX^n - R)/(G) = (QG)/G = Q. \quad (6)$$

Eq. 6 describes what happens if the message integrity is not compromised, there should be no remainder, the CRC value should be zero.

Although the added benefit for the information integrity is that it will be intact when it is received, there is extra computation within the application to achieve this feat, whether the CRC is done in software or hardware. In [9], a data transmission of 16 bits and 8 bits is used to demonstrate the computation requirements in clock cycles for hardware and software using four bytes of data with two CRC bytes. The implementation in software yields 6400 clock cycles; in hardware the required clock cycles are 800. This represents a 700% in time reduction from software to hardware and in terms of energy consumption, it should also require less.

Research have shown that dealing with CRC implementation in IOT applications can result in retransmission issues regarding energy consumption. They try to reduce the error correction by implementing novel approaches for Bluetooth Low Energy CRC-24 [10]. By doing this, the transmitter will send less corrupted information thus reducing the required energy for its transmission. In [11], classification methods are used to improve the efficiency of electronic devices in standby mode by analyzing the time they are doing their activity and the time they are in sleep mode. It is an interesting scheme to predict when those devices will not be used so they can be turned off.

The approaches to efficient energy consumption in IOT devices stated earlier focus on the act of predicting the errors within the transmission or the use of the devices already in communication. We present another layer of analysis but at a lower level of abstraction, in the device itself. By measuring the energy consumed in the act of generating the CRC within the device, the energy will be determined at the clock cycles level. The latter will be accomplished by measuring the current drawn by the device.

3 Methodology

Energy measurement is described in [12], [13] and [14]. This paper uses the approach of [15]. Current is measured by a shunt resistor in a low side configuration. After the current of a period is measured, an integration of the period is required to calculate the energy consumed.

3.1 Algorithm

In the algorithm department, activity is initiated in the device regarding the CRC module initialization and codification. Fig.1 provides a block diagram of the algorithm, as such, a message is issued and it is comprised of an array of sixteen unsigned integer elements, each 16 bits wide; these elements are used to apply the CRC codification. This operation is generated with the internal CRC module of the microcontroller. It does so by applying a pre-defined polynomial for CRC-16 calculation for the 16 bit message. The value for the operation is calculated by the module and it is 0x8005. The Checksum value is also generated for the data sequence been sent.

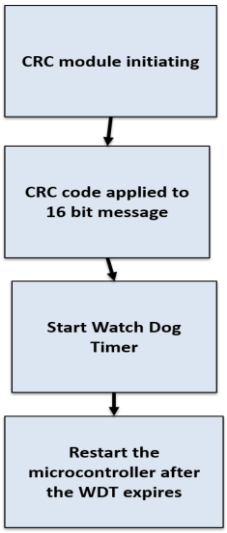

Fig. 1. Block diagram of the algorithm.

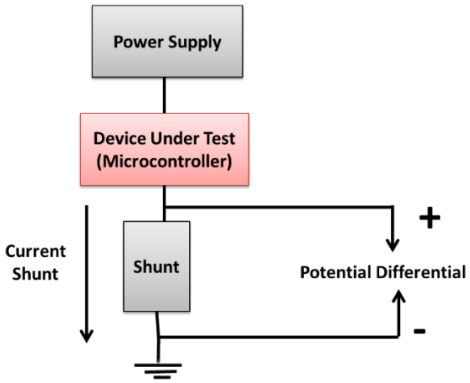

Fig. 2 Measurement setup.

Such data starts with 0x0000 and culminates with 0x000F, which comprise the array of sixteen elements; the checksum value is 0x1A0C. The algorithm is written in C and uses the free license for the XC16 compiler from Microchip in conjunction with the Microchip´s Code Configurator plugin. All the necessary functions required for the initialization and codification in hardware for the CRC-16 of the message are provided by the compiler.

3.2 Energy Measurement

Fig. 2 provides the measurement configuration by which the signal is acquired for its processing. A 10Ω@1% resistor is used as the shunt to provide the necessary voltage drop for the current to be calculated.

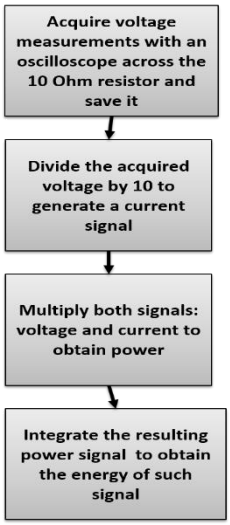

Fig. 3. Block diagram for the measurement and processing of the energy signal.

After the last element of the message is coded, the watchdog timer initiates operation; when such timer ends, it resets the device. The watchdog timer is configured to trigger every 1 ms. Since the device resets itself, no loops are required.

Fig. 3 illustrates a block diagram with the steps by which the signal is taken and then processed. A voltage reading is taken with a 10 Ohm resistor and is then saved for further analysis. The signal is then divided by 10 to generate the current passing through the shunt resistor. A power signal is then produced by multiplying the initial voltage signal with the new calculated current one. Once the power signal is known, an integration over one period is needed to calculate the energy consumption every time the device enters and exists the main program.

4 Results

Fig. 4, shows the energy profile [16], of the device. The red graph displays the voltage dropped across the 10 Ohm resistor with an oscilloscope.

A distinction should be made regarding the energy profile: it is not the actual energy consumed, it is the voltage. The energy of a device is described as the integral of the power over a certain time as stated in [17]. The latter is the reason why it is necessary to process the initial voltage signal through the energy measurements steps. Fig. 5, represents the power dissipated by the shunt resistor; this is the signal of interest for the integration in order to derive the energy consumption.

By integrating the power signal over a certain amount of time (period), we obtain the cumulative energy during the code execution. The operation of the initial configuration and the CRC calculation are 1238 cycles, plus 4 more for the sleep mode at 1 MHz or 1 microsecond for each instruction cycle.

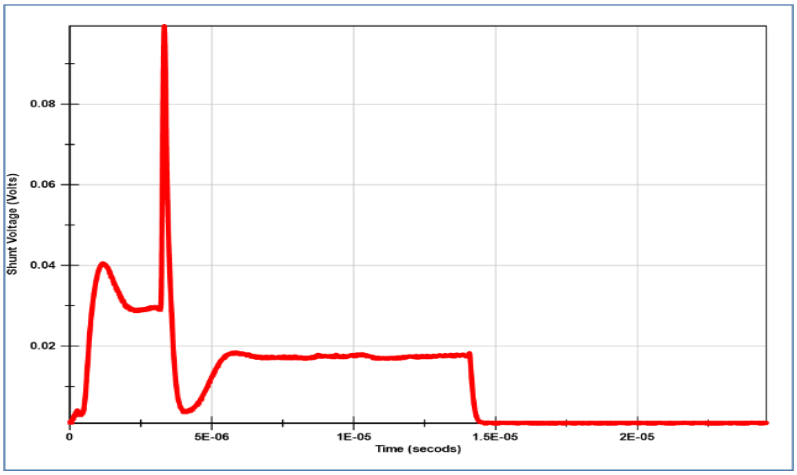

Fig. 4. Voltage across the shunt resistor.

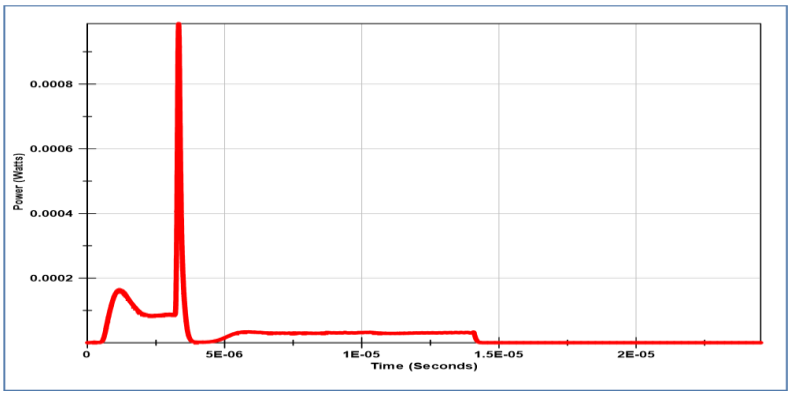

Fig. 5. Power dissipated in a period of the signal.

These clock cycles added determine the time the device is executing a task. The result is 1242 clock cycles, which translate to 1.242 seconds. The watchdog timer is configured to reset the device every 1mS after the last instruction has executed. Since the clock used for the watchdog timer is the internal RC module, it has a maximum value of 1.5 ms [18]. The actual time it takes to reset it is 1.238 ms, well within the specifications.

Fig. 6 displays the energy consumed by the device over one period. The cumulative energy consumption is 0.7203 nano Joules.

5 Conclusions

The energy consumption of a microcontroller with an internal CRC module is evaluated executing its operations at a clock cycle rate of 1 microsecond. The internal oscillator

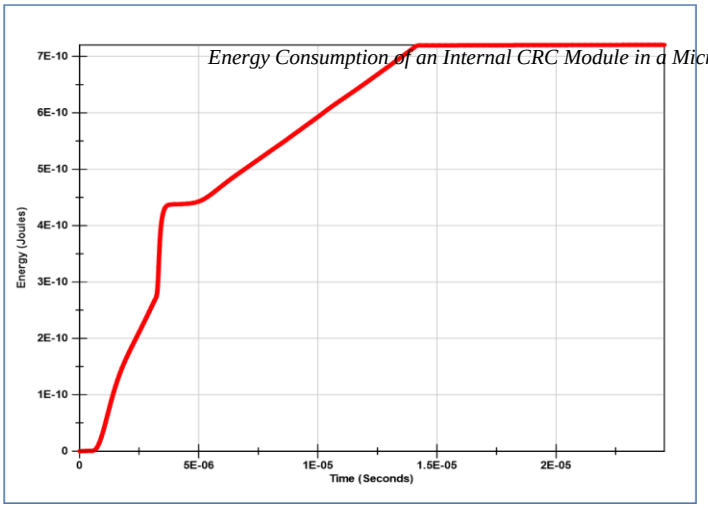

Fig. 6. Energy consumption over a period.

runs at 2MHz but the frequency of the instruction cycle runs at half that frequency. The computational power required for doing a CRC-16 with 16 bit, width data over an array of 16 unsigned integer variables is 1242 clock cycles with an energy consumption of 0.7203 nano Joules. The majority of the energy is drawn by the initial setup, which include the microcontroller wake up from reset. The initialization code form the XC16 compiler with no optimizations shows a steady power consumption up until the CRC module execution, as shown in Fig. 2 and 3. There is an increase in energy demand by way of an overshoot of current after the initial setup. This demand is most likely generated by the CRC module preparing to do the codification for the message.

No loops where used in the algorithm as the watchdog timer resets the microcontroller so it can repeat the operation form the power up state and not the evaluated while or for loop. Further investigation could be done using those loops to measure the current consumed by entering and exiting such loops.

The measurement was conducted for only one frequency; further analysis can be made by changing the frequencies of operation of the microcontroller.

References

1. Farhad, A. I.: Internet of Things Based Free Parking Space Management System. International Conference on Cloud Computing Research and Innovation (ICCCRI), 1(1), pp. 1–6 (2017)
2. De la Borbolla, C.T.: Applying the Internet of Things (IoT) to biomedical development for surgical research and healthcare professional training. IEEE Technology & Engineering Management Conference (TEMSCON), 1(1), pp. 335–341 (2017)
3. Bello, Z.: Intelligent Device-to-Device Communication in the Internet of Things. IEEE Systems Journal, 10(3), pp. 1172–1182 (2016)
4. Tsimbalo, F.P.: CRC Error Correction in IoT Applications, IEEE Transactions on Industrial Informatics, 13(1), pp. 361–369 (2017)
5. Tsimbalo, F.P.: Fix it, don't bin it -CRC error correction in Bluetooth Low Energy. IEEE 2nd World Forum on Internet of Things (WF-IoT), 1(1), pp. 286–290 (2015)

6. Peterson, B.: Cyclic Codes for Error Detection. (IRE), 49(1), pp. 228–235 (1961)
7. ATMEL: www.atmel.com (2017)
8. Nkom: Concise schemes for realizing 1-Wire® cyclic redundancy checks. 3rd IEEE International Conference on Adaptive Science and Technology, 1(1), pp. 70—79 (2011)
9. Microchip: www.microchip.com (2008)
10. Tsimbalo, F.P.: CRC error correction for energy-constrained transmission, IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), 1(1), pp. 430–434 (2015)
11. Andrade, R.N.P.: Applying classification methods to model standby power consumption in the Internet of Things, IEEE 14th International Conference on Networking, Sensing and Control (ICNSC), 1(1) pp. 537–542 (2017)
12. Tiwari, M.W.: Power Analysis Of Embedded Software: A First Step Towards Software Power Minimization. Computer-Aided Design, IEEE/ACM International Conference, 1(1), pp. 384–390 (1994)
13. Luo, G.S.L.R.: Analysis and Optimization of Embedded Software Energy Consumption on the Source Code and Algorithm Level. Embedded and Multimedia Computing, 1(1), pp. 1–5 (2009)
14. Ortiz, S.: Impact of Source Code Optimizations on Power Consumption of Embedded Systems. Circuits and Systems and TAISA Conference, 1(1), pp. 133–136 (2008)
15. Camarillo-Ramos, L.A.B.Z.: Medición del consumo de energía en un microcontrolador de 16 bits en operación y reposo. Memoria del 1er. y 2do. seminario de investigación de la facultad de ingeniería, 1(1), pp. 20–23 (2014)
16. Renesas:http://www.techonline.com/electrical-engineers/education-trainig/webinars/4420344/Implementing-Ultra-low-Power-Design-Techniques-for-High-performance-32-bit-MCU-based applications (2013)
17. Vey: AN1416 Low Power Design Guide, Microchip. http://ww1.microchip.com/downloads/en/AppNotes/01416a.pdf. (2014)
18. Microchip:http://ww1.microchip.com/downloads/en/DeviceDoc/39995_b.pdf. (2017)