wirelessCONGRESS
systems & applications

Low-Power Wireless – What is possible with Wi-Fi?

Andreas Müller and Andreas Rüst
Zurich University of Applied Science (ZHAW)
Institute of Embedded Systems (InES)
Winterthur, Switzerland
andreas.ruest@zhaw.ch

***Abstract* — Various chip vendors are offering low-power Wi-Fi solutions for embedded applications. These systems contain integrated TCP/IP stacks. They feature specific operation modes to minimize energy consumption of battery operated devices targeting applications in the internet-of-things area. The paper presents measurement results with regard to energy consumption. The results are used to compare the solutions of several chip vendors for selected use cases.**

Keywords — Wi-Fi, WLAN, low power wireless, embedded systems

I. INTRODUCTION

The wide spread deployment of access points makes the Wi-Fi technology attractive for many applications. However requirements of embedded devices differ from consumer devices. In many embedded use cases data has to be transmitted only during short time frames. The energy requirement of a continuously operated Wi-Fi connection is much too high for such applications. In such cases the specific use of miscellaneous vendor-specific sleep and power down modes allows to significantly reduce energy consumption.

Although the datasheets of the devices are providing information on power consumption for the supported operation modes, it is difficult for the user to calculate and compare the amount of energy required. On one side this depends on many application specific parameters such as amount of data, frequency and direction of data transmission, as well as usage of the medium. On the other hand energy consumption is influenced by vendor specific parameters. In addition to the power consumption itself these are especially the times required to enter into and to wake up from power down modes.

Selected use cases have been implemented on Wi-Fi platforms of different vendors. The resulting energy consumptions have been measured using a power analyzer [1]. The results show that the choice of platform depends on the use case.

This paper is structured accordingly. We begin by describing the two use cases that have been used for the power measurements. We continue by outlining the three different categories of Wi-Fi devices that have been evaluated. In the forth section we present the results of our power measurements and in the fifth section we summarize key findings discovered while performing the measurements. We end with appropriate conclusions.

II. USE CASES

An integral part of the power consumption is defined by the use case of the application. The length of the communication sequence as well as the amount of exchanged data has an impact on the required energy. Two different use cases have been defined for the power measurements on the selected Wi-Fi modules. Both use cases involve an access point and a central server. To minimize potential variation caused by external influences the following constraints are applied for all measurements unless stated otherwise.

1. The evaluated Wi-Fi module shall be the only device currently associated with the access point.
2. The distance between the Wi-Fi module and the access point shall be one meter.
3. All transmissions shall be WPA2-PSK encrypted.

A. *Upload Sensor Data to Server*

In this use case the embedded system with the Wi-Fi module acts as a client and wants to upload a relatively small amount of data to a central server. This is a typical scenario when a sensor wants to send either an individual measurement value or several measurement values grouped together. Fig. 1 shows this use case.

Fig. 1. Use case: "Upload Sensor Data to Server".

As soon as the measurement data is available, the client system applies power to the Wi-Fi module. The Wi-Fi module powers up, associates itself with an available access point and connects to a server. Next the client uploads 20 bytes of data to the server before the Wi-Fi module disconnects and powers down.

B. *Download Data from Server*

While the first use case uploads a small amount of data this second use case downloads a larger amount of data from a central server. Typical applications are the download of a new firmware image, of a larger configuration file or of information to be displayed. Fig. 2 shows this use case.

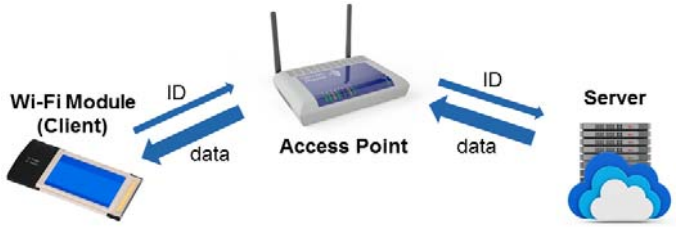

Fig. 2. Use case: "Download Data from Server".

In this use case the Wi-Fi module powers up, associates itself with an available access point and initiates the download by transmitting an ID of 6 bytes to the server. The server responds by sending 64 Kbytes of data to the Wi-Fi module. Finally the Wi-Fi module forwards the received data using a UART and powers down.

C. *Test Hardware*

For Atmel WINC1500A [2], Gainspan GS2011 [3], Microchip RN171 [4] and TI CC3100 [5] the measurements have been done using the evaluation boards of the manufacturers. For the Espressif ESP8622 [6] the evaluation board of Olimex has been used.

The power measurements only included the Wi-Fi module itself except for TI and Espressif where additionally the external flash memory has been included.

As access point a Fritz!Box Fon WLAN 7170 [7] has been used. The server side was an own Java software running on a Dell Latitude E6540 with Windows 7.

III. EVALUATED WI-FI DEVICES

According to Fig. 3 the evaluated Wi-Fi devices can be classified into three categories based on their type of interface. The type of interface and especially the available interface speed have a direct impact on the overall power consumption. Therefore it is not always possible to directly compare the absolute power consumption values between the devices.

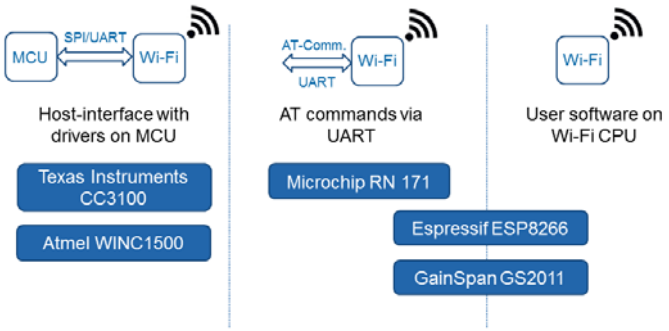

Fig. 3. Categories of Wi-Fi devices

A. *Host-interface with Drivers on External MCU*

For the devices in this category an external MCU uses an SPI or UART to directly read and write from/to registers on the Wi-Fi module. The register interface is used to configure and monitor the Wi-Fi module as well as to transmit and receive data. Representatives of this category are the Texas Instruments CC3100 and the Atmel WINC1500. From TI a silicon version with an additional user programmable CPU is available. However for these measurements the chip version with the standalone radio has been used to avoid biasing the measurement results with the individual power consumption of the additional CPU.

B. *AT Commands via UART*

In this category an external MCU connects via a UART and uses short text strings, so-called AT commands to control the Wi-Fi module. Similarly AT commands are used to transmit and receive data. This causes overhead and may lead to longer transmission times. The Microchip RN 171 device belongs to this category. For the devices Espressif ESP8266 and GainSpan GS2011 this is one of two possible operating modes.

C. *User Software on Wi-Fi CPU*

The third category provides the user with the possibility to load his software onto the same CPU that

runs the Wi-Fi/TCP/IP stack. This allows building systems where only a single CPU is required. The devices Espressif ESP8266 and GainSpan GS2011 support this mode.

IV. Measurement Results

The measurement results for the individual devices are presented in alphabetical order.

A. Atmel WINC1500A

The device from Atmel features a fast cold-start as well as short times for association with an access point and for establishing a connection to a server. It is the only chip that automatically adapts the transmission power to the required signal strength. This allows saving additional energy in cases where the access point is close, i.e. as positioned in this evaluation. The device is controlled through a host-interface using call-backs. Measured peak currents are around 220 mA at 3.0 V.

1) Use Case "Upload Sensor Data to Server"

Fig. 4 shows the dynamic energy consumption of the Atmel device in the first use case. The whole sequence lasts only 1.7 seconds, which is very short compared to the other devices. Total energy consumed is a low 132 mJ. Using all the energy stored in an AA battery the process could be repeated about 94'000 times.

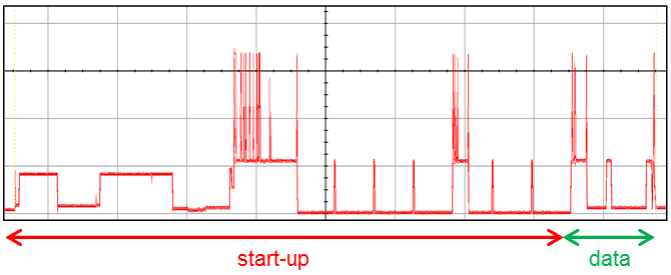

Fig. 4. Energy consumption of Atmel WINC1500A in use case "Upload 20 byte"

2) Use Case " Download Data from Server"

Fig. 5 shows the dynamic energy consumption of the Atmel device in the second use case. The complete sequence takes 4.7 seconds and uses a total energy of 280 mJ. This could be repeated about 44'000 times on an AA battery.

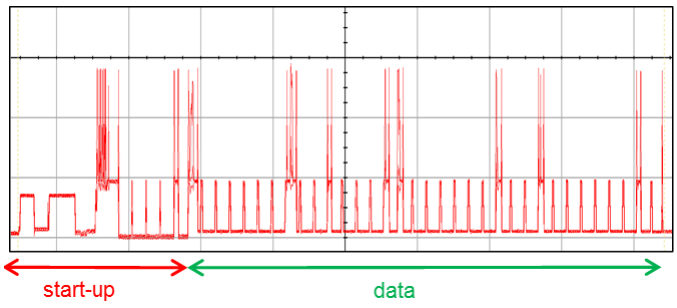

Fig. 5. Energy consumption of Atmel WINC1500A in use case "Download 64 kbyte"

B. Espressif ESP8266

The Espressif device sticks out with its low cost of under 4 USD. It requires only very few external components. With its integrated Tensilica Xtensa core it features a powerful internal CPU which can be utilized by the user for application programs. Alternatively it can be programmed through AT commands. Unfortunately the device has a rather energy intensive start-up. Measured peak currents are around 210 mA at 3.0 V.

1) Use Case "Upload Sensor Data to Server"

Fig. 6 shows the dynamic energy consumption of the Espressif device in the first use case. The whole sequence lasts 6.1 seconds. Total energy consumed is 1032 mJ, which is almost 8 times more than the Atmel device. Using all the energy stored in an AA battery the process could be repeated about 12'000 times.

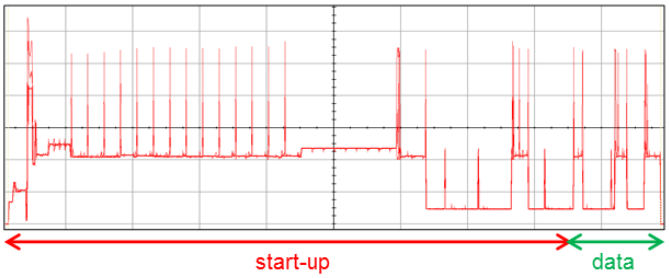

Fig. 6. Energy consumption of Espressif ESP8266 in use case "Upload 20 byte"

2) Use Case " Download Data from Server"

Fig. 7 shows the dynamic energy consumption of the Espressif device in the second use case. The complete sequence takes 8.6 seconds and uses a total energy of 1580 mJ. This could be repeated about 7'800 times on an AA battery.

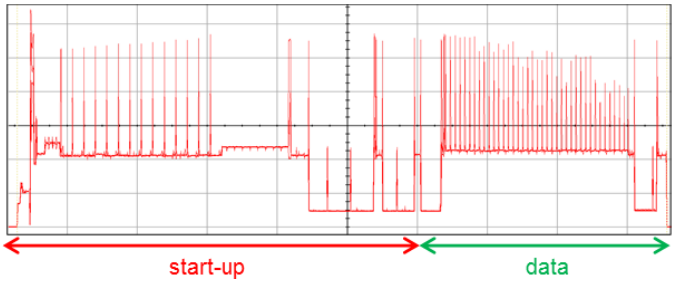

Fig. 7. Energy consumption of Espressif ESP8266 in use case "Download 64 kbyte"

C. Gainspan GS2011

The Gainspan device displays a very solid and good overall performance but does not show outstanding highlights. It features a huge AT command set as well as native programming on the integrated Cortex-M3 processor. An active connection can be held with as low as 15 mA at 3.0 V. Measured peak currents are around 350 mA at 3.0 V.

1) Use Case "Upload Sensor Data to Server"

Fig. 8 shows the dynamic energy consumption of the Gainspan device in the first use case. The whole sequence lasts 4.6 seconds. Total energy consumed is 696 mJ. Using all the energy stored in an AA battery the process could be repeated about 17'800 times.

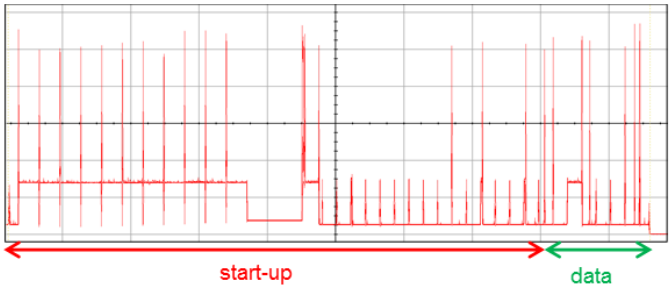

Fig. 8. Energy consumption of Gainspan GS2011 in use case "Upload 20 byte"

2) Use Case " Download Data from Server"

Fig. 9 shows the dynamic energy consumption of the Gainspan device in the second use case. The complete sequence takes 8.2 seconds and uses a total energy of 931 mJ. This could be repeated about 13'300 times on an AA battery.

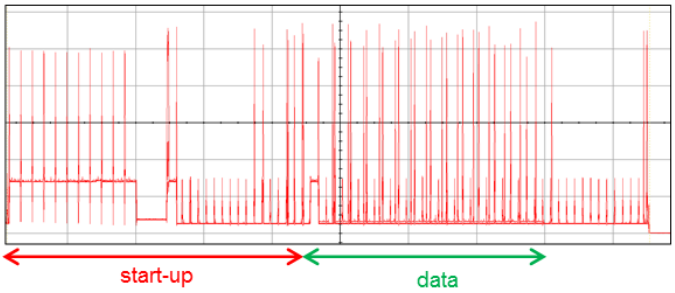

Fig. 9. Energy consumption of Gainspan GS2011 in use case "Download 64 kbyte"

D. Microchip RN171

The device from Microchip provides an extremely fast association with the access point but it takes a long time to obtain an IP address through DHCP and to open/close a TCP connection. The receiver draws little current but unfortunately it cannot be disabled so there is a constant current draw of about 40 mA. Measured peak currents are around 280 mA at 3.0 V.

1) Use Case "Upload Sensor Data to Server"

Fig. 10 shows the dynamic energy consumption of the Microchip device in the first use case. The whole sequence lasts 5.5 seconds. Total energy consumed is 703 mJ. Using all the energy stored in an AA battery the process could be repeated about 17'700 times.

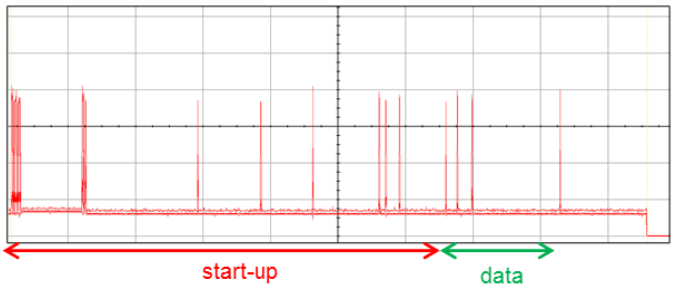

Fig. 10. Energy consumption of Microchip RN171 in use case "Upload 20 byte"

2) Use Case " Download Data from Server"

Fig. 11 shows the dynamic energy consumption of the Microchip device in the second use case. The complete sequence takes 8.5 seconds and uses a total energy of 1098 mJ. This could be repeated about 11'300 times on an AA battery.

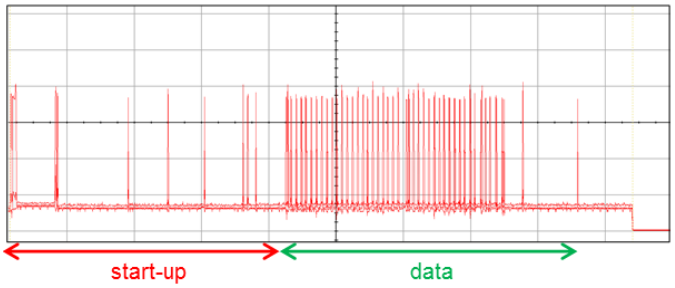

Fig. 11. Energy consumption of Microchip RN171 in use case "Download 64 kbyte"

E. Texas Instruments CC3100

The Texas Instruments device features an extremely fast switching between its individual power modes. An additional highlight is the low energy consumption during an inactive connection. Unfortunately it employs a long and energy consuming cold start. Measured peak currents are around 350 mA at 3.0 V.

1) Use Case "Upload Sensor Data to Server"

Fig. 12 shows the dynamic energy consumption of the Texas Instruments device in the first use case. The whole sequence lasts 3.2 seconds. Total energy consumed is 264 mJ. Using all the energy stored in an AA battery the process could be repeated about 47'000 times.

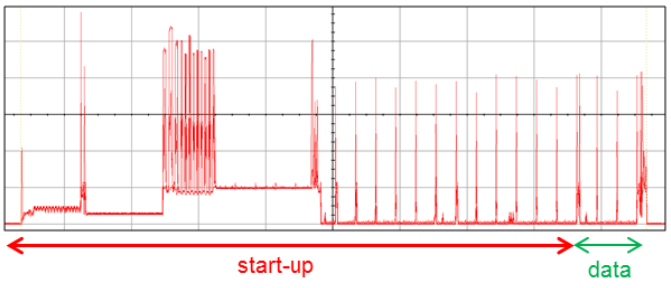

Fig. 12. Energy consumption of TI CC3100 in use case "Upload 20 byte"

2) Use Case " Download Data from Server"

Fig. 13 shows the dynamic energy consumption of the Texas Instruments device in the second use case. The complete sequence takes 6.1 seconds and uses a total energy of 498 mJ. This could be repeated about 24'900 times on an AA battery.

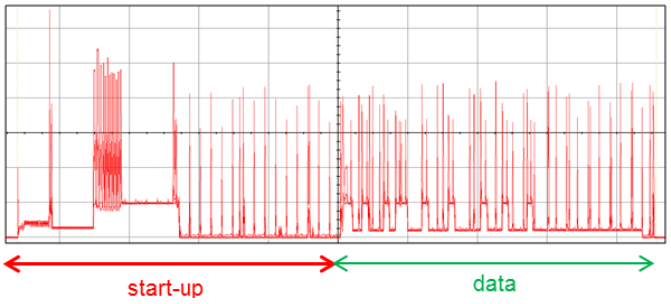

Fig. 13. Energy consumption of TI CC3100 in use case "Download 64 kbyte"

F. Texas Instruments WL1835

Measurements of this device [8] have been included as a reference. Unlike the other devices this device is not targeted at low power and low cost MCU applications but rather at applications that require higher Wi-Fi performance. It can reach a throughput of up to 100 Mbit/s. The measurements have been done using a BeagleBone Black board with Debian 8, Kernel 4.1.

Fig. 14 shows the dynamic energy consumption of the device in the second use case. The complete sequence takes 5.6 seconds and uses a total energy of 2.07 J. This is about 7 times higher than the Atmel device but only about 1.3 times higher than the device from Espressif. Measured peak currents are around 426 mA at 5.0 V.

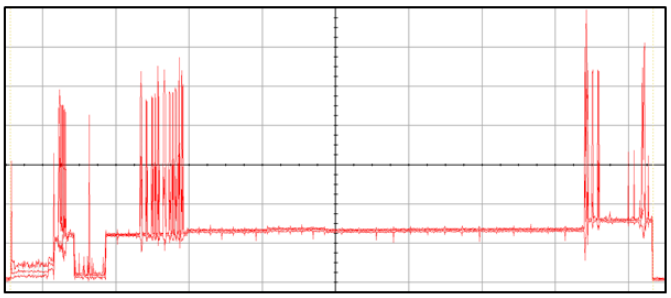

Fig. 14. Energy consumption of TI WL1835 in use case "Download 64 kbyte"

V. Key Findings

This section highlights several key findings discovered during the measurements. The described points shall illustrate some trade-offs faced by system designers.

A. Stay Connected

Many embedded systems spend most of their time in an energy optimized sleep state. They only need to transmit data at certain discrete points in time. So far we have seen that each association with an access point requires a relatively high amount of energy. On the other hand some of the evaluated chips have dedicated low power modes where they can remain associated with the access point and connected to the network (e.g. keep their assigned IP address). Now, is it more efficient to power-down the chip and re-connect in case data needs to be transmitted or to stay associated and connected in the dedicated low power mode? The answer depends on the time interval between two transmissions. TABLE I. provides measurement data based on the TI CC3100 in the use case "Upload Sensor Data to Server". While the energy for each connection is independent of the transmission interval, the energy to stay connected directly depends on the transmission interval.

TABLE I. Energy Needed to Stay Connected in Low Power Mode

Transmission Interval [minutes]	Energy for connecting and sending 20 bytes [mJ]	Energy for staying connected and sending 20 bytes [mJ]
1	264	94
5	264	412
10	264	793
15	264	1185

As a result if we want to send two packets of 20 bytes each and one minute apart then this will require a total of 2 * 264 mJ = 528 mJ in case we use the reconnection option. Whereas if we stay connected we will consume 264 mJ + 94 mJ = 358 mJ. So for a time interval of one minute it is significantly better to stay connected. However broadcast traffic may cause additional wake-ups and therefore increase the power consumption for the case where the Wi-Fi module stays connected.

Yet in a case where the sending of the two packets is 5 minutes apart the energy need will be lower if we power-down and reconnect. The energy requirements for a time interval of 5 minutes are 2 * 264 mJ = 528 mJ and 264 mJ + 412 mJ = 676 mJ for reconnection and staying connected respectively.

B. Programming

How much does the energy consumption depend on the type of programming interface used? To evaluate this question the two use cases have both been implemented in two different ways on the Espressif ESP8266: Once using AT commands and once as native code on the internal CPU. TABLE II. summarizes the measurement results.

TABLE II. ENERGY NEEDED: AT COMMANDS VS. NATIVE CODE ON ESPRESSIF ESP8266

Use Case	AT Commands		Native Code	
	[mJ]	*[s]*	*[mJ]*	*[s]*
Upload	1032	6.04	938	5.82
Download	1580	8.59	1520	8.13

Sending AT commands from an external microcontroller through a serial interface requires slightly more time compared to the native solution. The native code requires 9.1 % less energy for the first use case (Upload 20 bytes) and 3.8 % less energy for the second use case (Download 64 Kbytes). Thus using native code allows a slight reduction of energy consumption. However in case of the AT commands the power consumption of the required external microcontroller has to be added as well, shifting the results even more in favor of the native code solution.

C. *Difference TCP/UDP*

Can the use of UDP instead of TCP help reduce energy? By design TCP has a higher overhead used to establish its connection than the connection-less UDP. TABLE III. shows measurement results for the use case "Upload Sensor Data to Server". On the Atmel device UDP reduces the overall energy consumption by 28.6 % whereas on the TI device the reduction amounts to 18.6 % compared to TCP.

TABLE III. ENERGY NEEDED: TCP VS. UDP FOR USE CASE "UPLOAD SENSOR DATA"

Device	TCPs		UDP	
	[mJ]	*[s]*	*[mJ]*	*[s]*
Atmel WINC1500	175	2.03	125	1.04
TI CC3100	598	5.92	497	5.24

Even though UDP significantly reduces energy consumption it is no silver bullet. The evaluated Wi-Fi modules have rather small receive buffer sizes compared to the high through-put capacity on the server side. As UDP does not provide flow control, chances are that the powerful server side overloads the low power client side. As a result data packets may be lost.

D. *Access Points*

So far the energy measurements have been performed using an access point without any load, i.e. the evaluated Wi-Fi modules were the only clients connected through the access point. This allows reproducible measurement results. Although in real world situations the load on access point and server can have a significant influence on the connection time and therefore on the required energy. A simple experiment using the TI CC3100 in an office environment with a typical network load illustrates the effect. An adapted use case has been used. The Wi-Fi module connects to the server and sends 6 bytes. In response it receives 1400 bytes from the server. The radio remains initialized, so the measured values do not include the radio start-up. Fig. 15 shows the measured distribution of 80 transfers. The histogram shows that the load can influence the required energy by more than a factor 2.

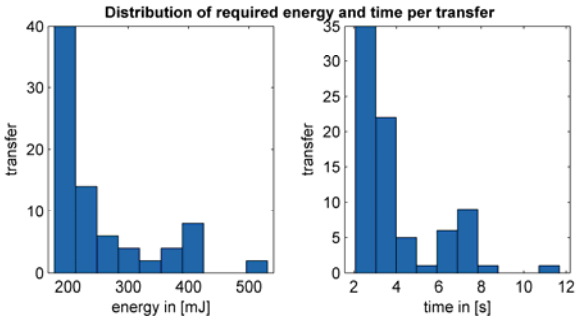

Fig. 15. Influence of load on energy and connection time for the TI CC3100

E. *Miscellaneous*

The application of the low power sleep modes does not always reduce energy consumption. Wake up time from sleep can slow down data transmission. As a result in some cases the use of a sleep mode can therefore effectively increase the required energy. Sometimes it is more effective not to use them.

From an energy point of view it is generally better to send fewer but larger packets. The optimal packet size depends on the Maximum Transfer Unit (MTU) of the MAC layer, which often amounts to 1460 bytes.

VI. CONCLUSIONS

This paper presents energy measurement results for five commonly available low-power Wi-Fi modules for embedded applications. All modules feature an integrated TCP/IP stack. Two distinct use cases have been implemented on all modules and have subsequently been measured with regard to energy consumption. The provided data illustrates involved trade-offs and facilitates selection of an appropriate device. In addition several important aspects that have to be kept in mind during the design of a low power Wi-Fi embedded system are highlighted. Even though Wi-Fi is still quite energy hungry compared to other wireless technologies, it opens interesting opportunities for specific applications in the embedded systems area.

REFERENCES

[1] Keysight Technologies N6700, Modular Power System Family, Data Sheet, Keysight Technologies, 2014

[2] WINC1500-MR210PA Datasheet, Atmel Corporation, 2015

[3] GS2011M Low Power WiFi Module, Data Sheet, GainSpan Corporation, 2015

[4] RN171, 2.4 GHz IEEE Std. 802.11 b/g Wireless LAN Module, Microchip Technology Inc., 2014

[5] CC3100 SimpleLink™ Wi-Fi® Network Processor, Internet-of-Things Solution for MCU Applications, Texas Instruments Incoporated, 2015

[6] ESP8266EX Datasheet, Version 4.3, Espressif Systems IOT Team, 2015

[7] Fritz!Box Fon WLAN 7170, Installation, Einrichtung und Bedienung, AVM Computer Systeme GmbH, 2010

[8] WL18xxMOD WiLink™ 8 Single-Band Combo Module – Wi-Fi®, Bluetooth®, and Bluetooth Low Energy (BLE), Texas Instruments Incoporated, 2014