The Value of Low Power

February 2018

This Thread Technical white paper is provided for reference purposes only. The full technical specification is available publicly. To join gain access, please follow this link: https://www.threadgroup.org/ThreadSpec.

If there are questions or comments on these technical papers, please send them to help@threadgroup.org.

Table of Contents

I. Introduction

An increasing amount of Internet-connected services are now available to homeowners. These services are designed to protect homes by granting restricted access rights; automating operations like heating and cooling to lower energy bills; and controlling lighting, luminance, closures, window covering or ventilation systems.

Home automation systems with products that can deliver these features are developed with the idea that the individual devices can be installed easily and securely, talk to each other seamlessly to enhance the service value they enable, and act/respond instantaneously. Consumers expect these abilities even though the products operate in different ways; for example, some issue mission-critical events (like reporting a home intruder) while others execute an actuating command (like dimming the level of a light bulb) sent from another device within the home or an Internet-connected node outside the home.

Wireless connectivity technologies are the backbone of these home automation systems. Wireless offers ease of installation and maintenance, and reduces redeployment costs compared to wired systems. Communication within the network remains secure, reliable and scalable when using protocols like Thread.

The products that make up a home automation service are normally battery operated and frequently built with a small form factor. Alarms, intrusion and environmental sensors, switches, electric shades or curtains and access control devices like door locks are just some examples of battery-sourced products within this ecosystem.

In many cases, a node must operate from a small coin-cell battery because of its size constraints, like a door contact. Homeowners' value as well extended battery life for such products; no one likes getting on a ladder to handle a smoke detector's low-battery beep. Wire-free battery powered shades and curtains make the installation of such products an easy task. Better power performance also reduces overall system costs. Imagine if homeowners had to replace the batteries every three months for each door and window sensor or garage door remote control: they want instead seamless connectivity and multiple years of coin-cell battery life.

Low-power performance is a fundamental value not only for smart home networks but for smart building applications. In the commercial space, light switches, environmental sensors, carbon monoxide detectors and smoke detectors may need to be torn down and reinstalled regularly to adapt to new layouts. Commercial building networks can be very large (thousands of devices per installation) compared to traditional home automation installations, thus increasing the overall system cost for device maintenance.

The portability of battery-powered sensors and actuators streamlines office retrofitting operations, and keeps costs down. Extended battery life maximizes system reliability by minimizing out-of-service occurrences when one node in the network has a dead battery.

In this paper, we'll describe how the Thread network protocol was designed with low-power performance in mind from the ground up. We'll explain how technological choices translate into extended battery life, and how to analyze Thread-based sensor usage. In the ending section, we'll conclude summarizing how the low power capabilities of Thread provide tangible benefits for home and commercial deployments, thus making it an attractive technology for such applications.

II. How Thread is designed for Low Power performance

Figure 1 illustrates the Thread network protocol architecture, with its corresponding breakdown of layers following the Open Systems Interconnection (OSI) model.

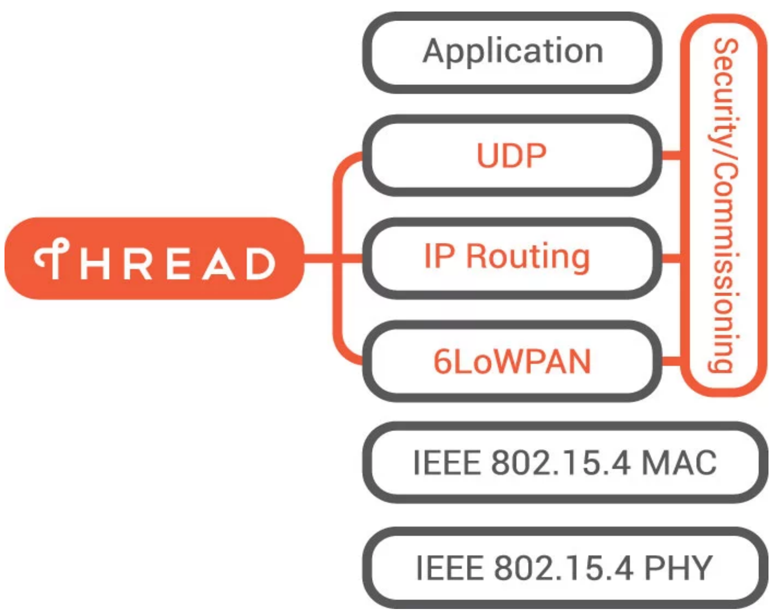

Figure 1: Thread protocol layering

In Thread, every device establishes a link to its one-hop peer, following the Institute of Electrical and Electronic Engineers (IEEE) 802.15.4 standard, a widely adopted and mature technology for low-power local networking communication used in home and building automation networks, as well as retail and factory automation.

Thread uses the asynchronous mode of operation within the 802.15.4 specification – making it a good fit for battery-operated peripheral devices (also known as "sleepy end devices"). These devices keep their radios mostly off and generate data only when an event occurs. For examples, door and window sensors or smoke detectors wake up only when there is a triggering event, like an intrusion. Lighting/shade actuators and remote controls efficiently implement operations through an asynchronous network, waking up only to send commands (like open/close) to other devices one or multiple hops away in response to user-triggered operations such as a button press.

Peripheral devices only communicate with their one-hop-away parent, irrespective of the location of the node for which the application frame is destined. As Figure 2 illustrates, the originator can go back to sleep right away once the parent (the end device proxy) successfully receives the packet emitted by the peripheral device, regardless of whether the data reached its destination or not. The parent delivers the packet to the target node through a distributed network routing system and mesh fabric. Regardless of whether the target is one or several hops away, the originator doesn't need to be awake to facilitate routing while the frame travels through the network.

Thread's 802.15.4 mode offers an effective power solution to support this type of traffic profile, eliminating the need to wake up regularly for housekeeping functions, or maintain a synchronous connection and adjust the device clock with the rest of the network.

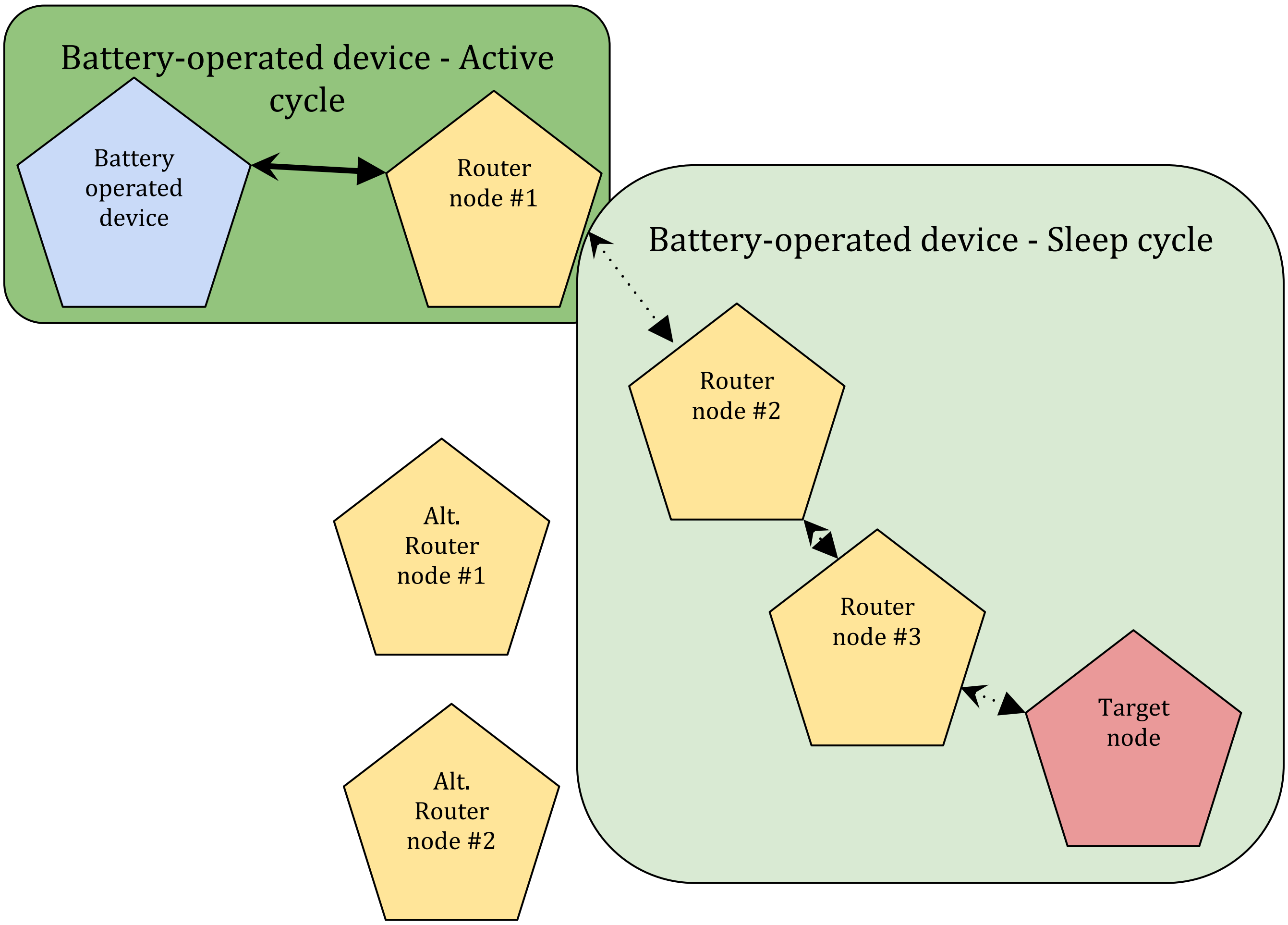

Figure 2: Routing of packet generated by a battery-operated device in a Thread network

But what happens when battery-operated sensors receive unsolicited data, like a sensor reporting a re-configuration command, and must send back their state (like battery information) for diagnostic operations? In 802.15.4's asynchronous mode, the parents are buffering the data for their peripheral device children, and the children extract the data when polling their parent. 802.15.4 poll frames are very short and, in the closed-loop acknowledgment packet sent back to the child, the parent indicates toggling a single bit if there is data addressed to the peripheral device waiting to be downloaded. This scheme keeps the over-the-air message exchange short and minimizes power consumption. If an application frame destined for the peripheral device is pending, the packet can be extracted right away in order to maintain processing and communication time. An asynchronous architecture gives you the flexibility to adjust poll frequency, trading-off battery life and downlink latency depending on the application needs.

In applications like environmental or security sensing, these downlink messages are very sporadic (like report re-configuration commands). Therefore, the latency requirements for downlink extraction are typically relaxed and designers can tune systems for the best power-consumption performance.

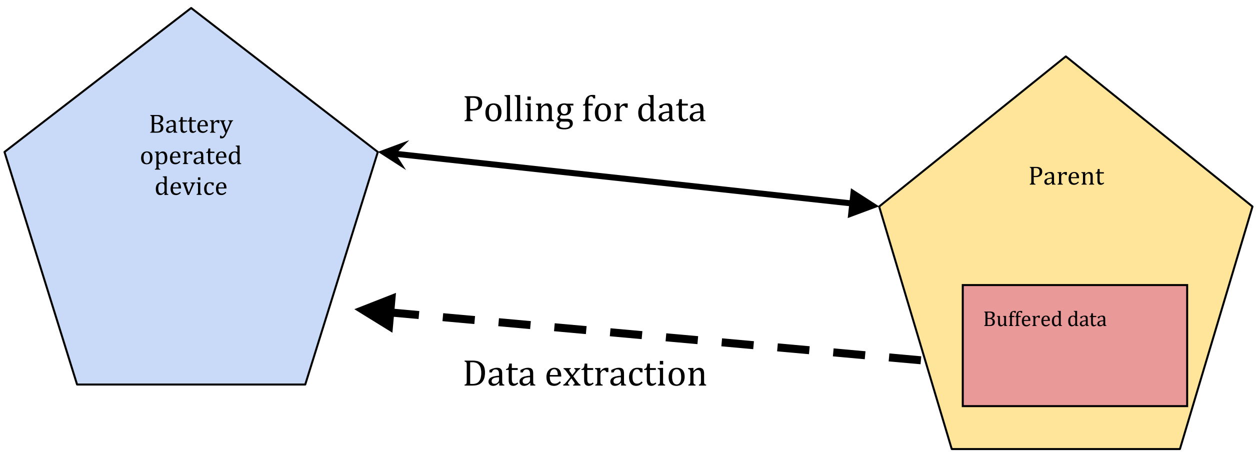

Figure 3: Downlink data extraction from the parent of a battery-powered device

The active communication cycle of a battery-operated Thread device is the major contributor to power consumption. Understanding the data points of operations during this phase helps illustrate how the Thread protocol keeps system power consumption at low levels. The duration of a single 802.15.4 frame can be very short (as little as a fraction of a millisecond), and with current technology for wireless microcontrollers (MCUs), the current drained during the reception and transmission phases can be contained to the single-digit milli-amperes.

In order to maintain low overall energy consumption, it is critical to control the power envelope (or “area under the curve”). This requires keeping the active cycle time very short. Having fewer active cycle occurrences to efficiently send the same amount of information wastes less energy.

Thread minimizes the overhead of the upper networking layers, efficiently encapsulating Internet Protocol (IP) packets and transporting them over the 802.15.4 data link. Thread uses IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) adaptation, a standards-based technology that compresses in-mesh communication IP headers efficiently and gives its best performance in terms of bytes reduction when the connectionless User Datagram Protocol (UDP) lies on top.

As Figure 1 illustrated, Thread chose UDP as the transport layer and selected the Constrained Application Protocol (CoAP) as the application substrate layer to efficiently exchange data and network management packets for operations like device commissioning and network management.

Compression factors for UDP and IP headers can bring down the packet overhead from 40 bytes to 2 bytes for in-mesh local communication, significantly reducing both overall packet processing and air time.

Another important feature of the 6LoWPAN layer is link-layer packet forwarding. Thanks to this feature, nodes can quickly and efficiently route-forward packets within the local mesh network without having to send them up to the network layer for further processing. This saves central processing unit (CPU) cycles and further improves power consumption.

Thread's design choices enable the application and management frames circulating in the network to remain mostly contained within a single 802.15.4 frame (where the maximum size is 127 bytes), thus reducing the over the air time and avoiding lengthy frame processing. This choice also minimizes active cycle instances and eliminates the need for nodes to have to deal with fragmentation and reassembly operations, which consume precious CPU cycles.

Battery-operated devices in a Thread network are directly reachable through their IPv6 address; thus, they can exchange IP packets with any Internet-connected device to balance active and sleep time. Combined with advances in the integration of low-power radio-frequency (RF) silicon technologies, and the ability to efficiently reach out to any point in the network thanks to the mesh topology, Thread can guarantee several years of battery life for typical sensor operations in home and building network environments.

Let's look at a specific use case, analyzing the power envelope during the active time, breaking down the different sections of the power curve and estimating the overall battery life of a Thread-based device in an environmental sensing application.

III. Use case example: a Thread environmental sensor

Imagine a node acting as a Thread peripheral device in a network. The application in this study is an environmental sensor (such as a temperature sensor) generating regular reports sent periodically to a data-aggregator device (such as a thermostat).

Here is the application profile:

- The sensor reports data every 60 seconds to a Thread thermostat, which is one or more hops away.
- The sensor checks in with the parent to see if there's data like a reconfigure report command every 4s. Assume that an actual reconfigure report command is very rare during the product life cycle.
- The application data (UDP payload) is 36 bytes, which is typical payload size for humidity-/temperature-sensing reports.

You can make these assumptions for this profiling study:

- The sensor runs on a coin-cell battery.
- The current consumed during commissioning is negligible and doesn't affect overall battery life. This is a reasonable assumption, since commissioning is typically a one-time operation.
- All packets are successfully received and there are no retransmissions.
- Current consumption is constant at all operational voltages.
- Zero capacity reduction due to (battery/chemical) aging is assumed.

Current profile for a sensor which is polling its parent:

1. Exit sleep.
2. Start and calibrate the crystal oscillator.
3. Initialize the radio.
4. Switch to receive mode.
5. Clear channel assessment (CCA).
6. Switch to transmit mode.
7. Transmit the IEEE 802.15.4 data request frame (22-byte media access control [MAC] frame).
8. Switch to receive mode.
9. Receive the IEEE 802.15.4 acknowledgement frame.
10. Enter sleep.

Current profile for a sensor sending data:

1. Exit sleep.
2. Start and calibrate the crystal oscillator.
3. Initialize the radio.
4. Switch to receive mode.
5. Clear channel assessment (CCA).
6. Switch to transmit mode.

7. Transmit the IEEE 802.15.4 data frame with 36-byte UDP payload (83-byte MAC frame).
8. Switch to receive mode.
9. Receive the IEEE 802.15.4 acknowledgement frame.
10. Enter sleep.

Figure 4 shows the structure of the wakeup and sleep cycle:

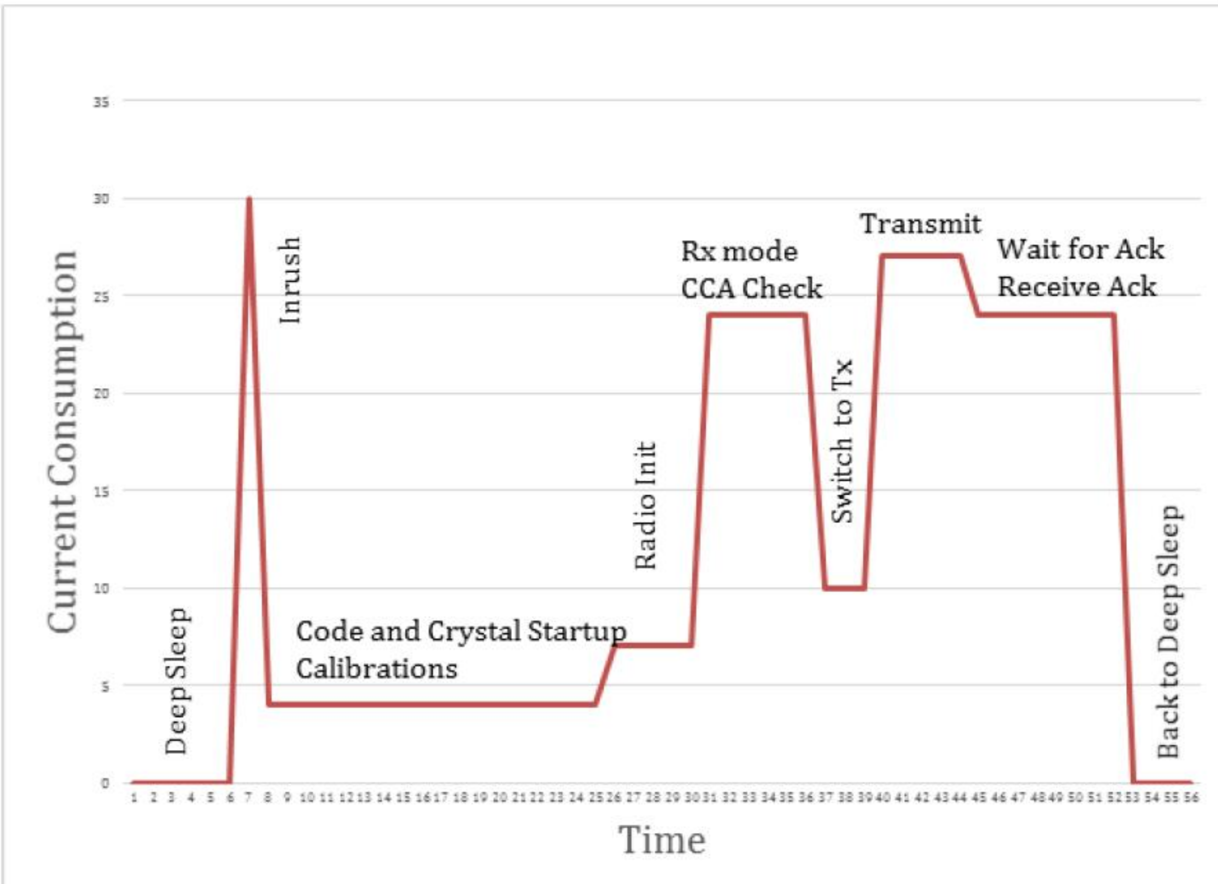

Figure 4: Typical wake-sleep cycle. In Y-axis the scale is in fraction of milliampere, while in the X-axis is fraction of milliseconds.

Current sensitivity to application data size

Thread devices transmit data at 250kbps. As a result, each byte of application payload adds an additional 32 microseconds of transmit time to the data-transmission phase (step No. 7 in sensor polling and sensor reporting data use cases).

Typical average current measurements on existing IEEE 802.15.4 radios

The actual current consumption varies between different IEEE 802.15.4 radios and platforms. The measurements from an existing IEEE 802.15.4 platform are:

- Sleep: 1.6 µA
- Data Poll at 1 second period: 22 µA
- Data Transfer (36-byte UDP payload) at 1 second period: 37 µA

Estimated lifetime of existing IEEE 802.15.4 radios

Three components model the current consumption for a peripheral device:

- Average current for sleep: 1.6 µA
- Average current for reporting data every 60 seconds: (37 µA - 1.6 µA) / 60 seconds = 0.59 µA
- Average current for polling every 4 seconds: (22 µA - 1.6 µA) / 4 seconds = 5.1 µA

Summing the above components, the total average current is 7.29µA.

Assuming a device with a CR2032 battery and a typical energy capacity of 200mAh, the battery lifetime is 200mAh/(7.29µA/1,000)/(24 × 365) = 3.13 years.

IV. Conclusions

Thread technology enables you to build an interoperable network that is secure, easy to deploy and scalable, where products can communicate within the local area and on the Internet using a single unified security and application model, thanks to the native support of the IP protocol suite.

Thread's power-conservative process dramatically extends the battery life of peripheral devices. For several types of end equipment, Thread can even enable direct IP connectivity for products that operate from a coin-cell battery, which in itself constitutes a paradigm shift when comparing Thread to any existing technology that supports IP communication natively.

While Thread is expanding its scope from the home to commercial deployments with its forthcoming additions to the Thread specifications and roadmap, it will keep low-power performance at its very core of operation and continuously improve it to provide enhanced value for battery-operated products, with even longer battery lives and shorter communication latencies.

References

1. IEEE 802.15.4 Specification
2. Thread Technical overview
3. Thread usage of 6LoWPAN
4. Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks