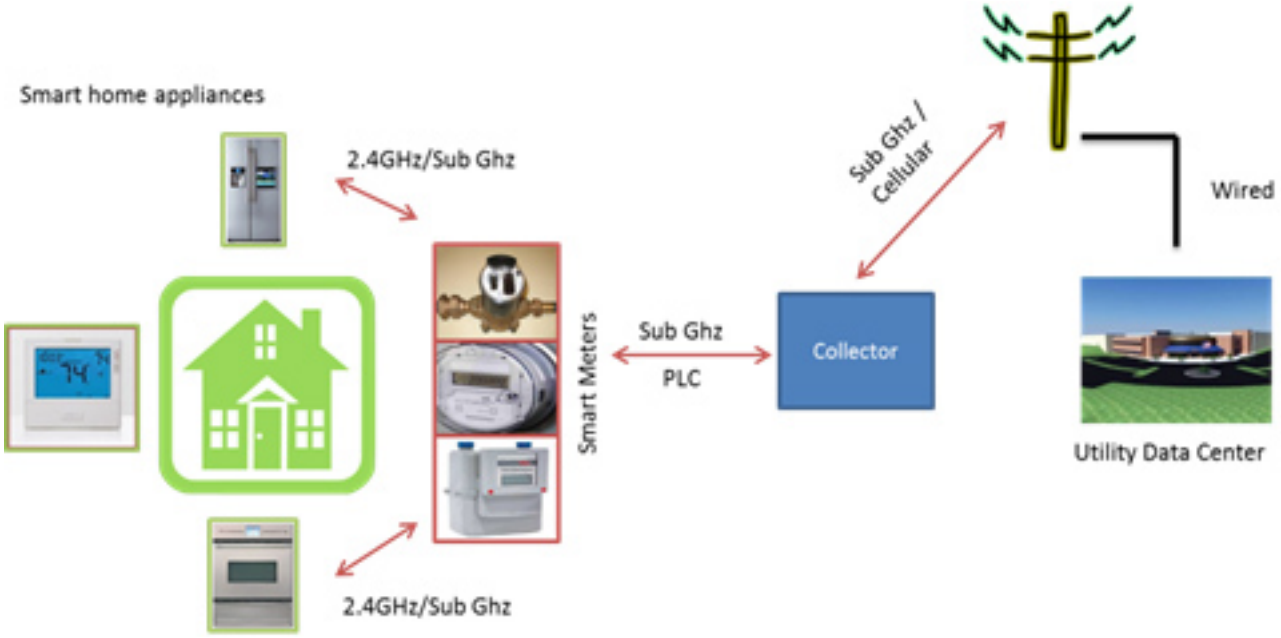
Implementing Sub-GHz Wireless Connectivity in Embedded Devices

Matt Maupin, SoC Marketing Manager, Silicon Labs

Wireless connectivity continues to expand across all markets and industries as growing numbers of electronic devices and systems add wireless controls and communications. Consumers desire the convenience of ubiquitous communication so wireless functionality is becoming a fundamental requirement for many electronics products. The rapid growth of the Internet of Things (IoT) is further driving the proliferation of wirelessly connected devices.

Industry leaders predict that the number of connected devices for the IoT will surpass 15 billion nodes by 2015 and top 50 billion by 2020. This phenomenal growth includes the more visible Internet-connected devices such as smartphones, tablets, wearable computing devices, and even TVs and set-top boxes. However, there are large numbers of proprietary wireless devices behind the scenes, such as security and surveillance systems, lighting control systems, remote keyless entry, smart thermostats and smart meters, and a host of other connected home and building automation devices.

The choice of wireless connectivity technology for embedded devices is a decision that requires careful consideration of various design choices. Primary factors in choosing the optimal communications technology include cost of deployment, security, regulatory compliance, range, and power consumption. Several communications technologies are available for wireless connectivity such as Wi-Fi, Bluetooth, ZigBee, and sub-GHz wireless. For many applications, such as backhaul communications from a smart meter into the home and to a data collector and utility data center (see **Figure 1**), sub-GHz technology is a popular choice because of its long-range performance, superior propagation, low-power operation (enabling longer battery life), and broad access to unlicensed sub-GHz spectrum around the world.



In fact, recent activity in IEEE 802.15 around smart utility networks and IEEE 802.11 for sub-GHz Wi-Fi shows continued industry support for sub-GHz wireless solutions. The increased demand for sub-GHz wireless connectivity has driven a variety of applications and with it a need for different wireless communications and control architectures.

Wireless System Design Considerations

The growth in the demand for connectivity has resulted in traditionally non-connected devices suddenly requiring wireless connectivity. In many cases the embedded developers creating these products do not have experience designing wireless products. The demand for wireless connectivity, together with the diversity of wireless-enabled products, is driving the need for flexible hardware and software architectures for wireless applications. Let’s examine three distinct wireless architectures for sub-GHz connectivity and how each approach provides an optimal solution that fits the specific needs of embedded device developers. There are numerous factors to consider, such as cost, form-factor constraints, design constraints, and time-to-market requirements, when selecting the right wireless architecture to implement in an embedded device.

- ***Cost***: To deploy 35 billion new connected devices by 2020, the total cost of ownership must be low. Cost of ownership includes not only the bill of materials (BOM) but also the product development cost and the ongoing cost of supply. Using the least expensive hardware solution will result in the lowest BOM cost, but may add significant development cost due to network and software development. The least expensive hardware solution may also require more support including field upgrades. Selecting the right architecture can minimize the total cost incurred and maximize profits.
- ***Product Size Constraints***: For the Internet of Things to be successful, connected devices need to be useful without being intrusive. Consumers want data from hundreds of devices in their homes, but they do not always

want to see them, so the physical size of the device is an important design consideration. The most common constraints that can impact the system architecture include, the physical size of the embedded device including the printed circuit board (PCB) area that can be used to implement the wireless feature and the size of the battery. If the product design is space-constrained, it will be necessary to limit the size impact of adding wireless connectivity. Factors that can impact space requirements include IC footprint, antenna size, the total number of discrete components used in the wireless solution and power consumption. Minimizing power consumption can reduce the size of the battery required. Reducing the number and/or size of any of the system components can have a direct effect on the overall product size and form factor.

- ***Time-to-Market***: New Internet of Things applications are emerging daily and evolving quickly so fast time to market is required. The choice of wireless architecture can directly affect time to market. A key area often overlooked is the time required to integrate wireless software into an application. In addition, wireless specifications are still evolving and may require updates to the wireless software more often than the core product code. Mixing stable product code with evolving wireless code may increase the amount of test and validation, increasing time to market. In some cases, separating the wireless software from the application software can reduce this risk and accelerate time to market.
- ***Other Design Constraints***: Embedded developers face numerous other design constraints such as power consumption, wireless range, and radiated emissions when specifying wireless solutions for their applications. While the wireless architecture may not directly affect all of these constraints, the product’s requirements may not be met by all architectures. For example, if the embedded device needs specific user-interface features such as an LCD or touchscreen, a single wireless IC may not offer all of the capabilities required for the end product, and the designer must look for multi-chip solutions to meet the product requirements.

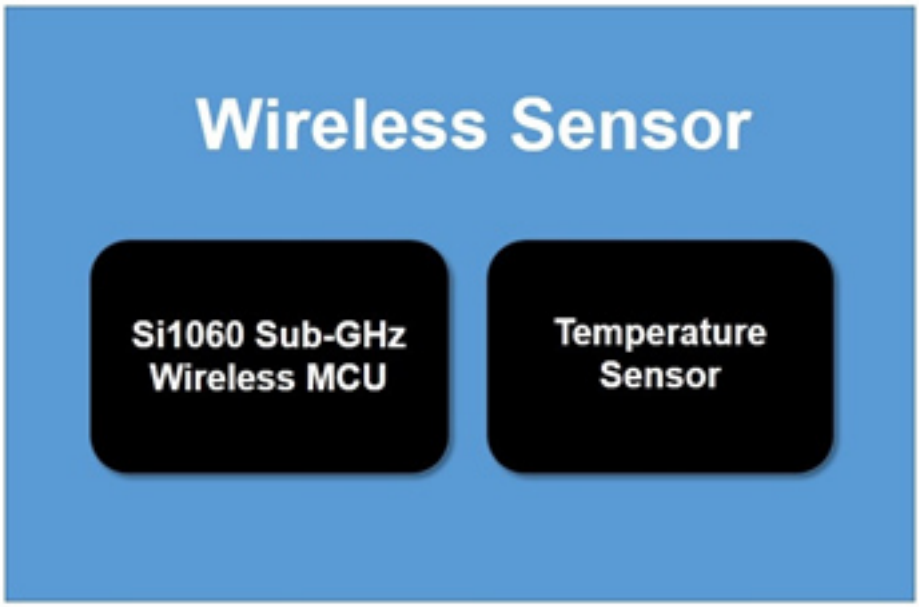
Wireless Connectivity Solutions

Now let’s examine three commonly used system architectures for embedded wireless solutions. The system architecture determines the hardware and software partitioning including the choice of IC components. **Figure 2** shows three options, based on the OSI model, for partitioning the software and hardware blocks in an embedded system design.

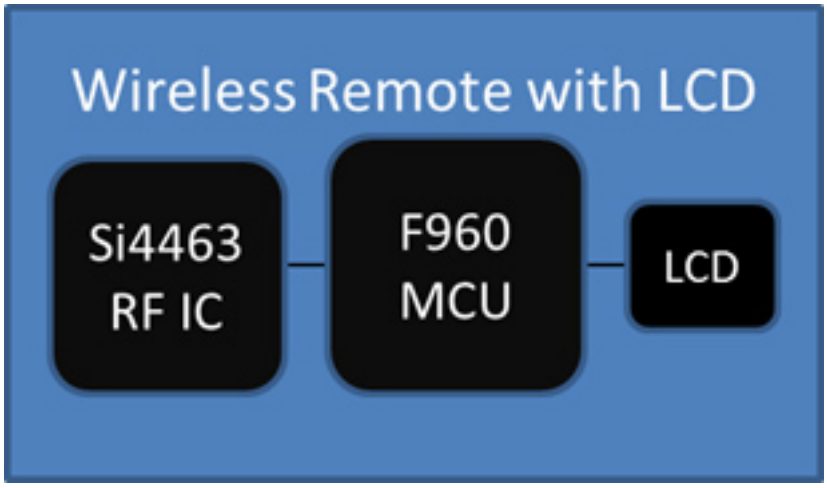
<table>
<tr><th>OSI Layer</th><th>SoC</th><th>MCU+Transceiver</th><th>NCP</th></tr>
<tr><td>Application</td><td rowspan="4">Wireless MCU (Si1060)</td><td rowspan="4">MCU (F930)</td><td>Host MCU/MPU (EFM32WG)</td></tr>
<tr><td>Presentation</td><td rowspan="4">Wireless MCU (Si1082)</td></tr>
<tr><td>Session
Transport</td></tr>
<tr><td>Network</td></tr>
<tr><td>Data
Physical</td><td colspan="2">Radio (Si4460)</td><td></td></tr>
</table>

SoC: A system-on-chip (SoC) device, such as Silicon Labs' Si1060 wireless microcontroller (MCU) product, combines an MCU and a transceiver into a single-chip solution that runs the wireless software stack and the application software. The SoC must have enough functionality to support the embedded device such as the necessary I/O for push-button controls and analog-to-digital-converters (ADCs) for sensing conditions such as temperature and humidity. In addition, SoCs are typically ultra-low-power, small-form-factor devices that are designed to enable extremely long battery life. This option often provides the most cost-effective solution and smallest physical size.

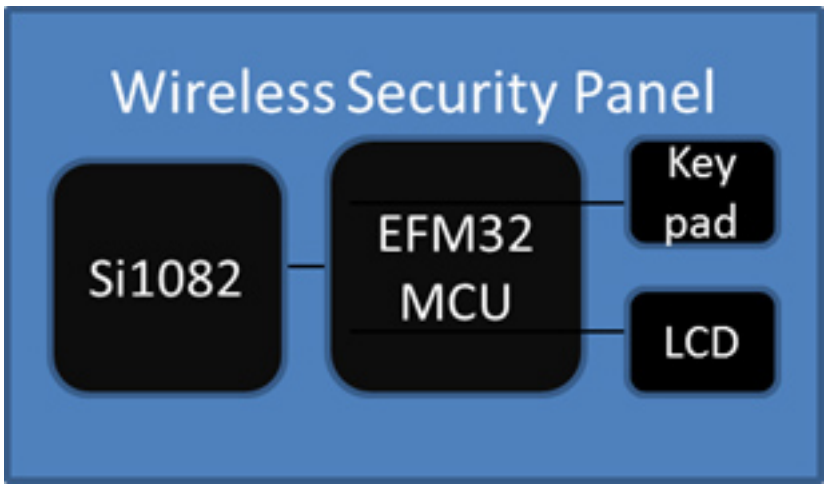
Typical applications for SoCs include fixed-function devices with a simple user interface or in many cases no user interface at all such as remote controls, key fobs for locking/unlocking car doors or home door/window security sensors.



MCU + Transceiver: If a suitable SoC is not available, system architectures that combine a separate MCU and wireless transceiver allow developers to select the optimal MCU for the application. MCU options include not only MCU architecture (8-bit versus 32-bit including a choice of ARM Cortex-M cores) but also on-chip features such as an LCD controller, USB support, multiple I/Os, timers, comparators, ADCs, and a range of flash memory sizes. The developer can then add the best transceiver solution to meet the application's wireless requirements. From an application standpoint, an MCU + transceiver system is similar to an SoC in that the network is typically fairly simple, and there are relatively few timing concerns in the system. The MCU runs all of the software including the application and wireless software. However, the physical layer (PHY) and data link layer are often integrated on the transceiver. The end product will often include a simple user interface for setup and control.



NCP: A network coprocessor (NCP) maybe used when the complexity and performance needs of the end product entail the use of a high-end MCU or microprocessor (MPU). Similar to a wireless SoC, an NCP combines an MCU and transceiver into a single-chip solution, but an NCP does not have the functionality to support the full application. The NCP runs the entire communications stack and interfaces to the host processor via a serial interface such as a UART or SPI. The NCP architecture separates the application's software complexity from the communications software, which is critical in applications where the network stack timing is critical or there are high throughput requirements. While this approach is the most expensive system solution option, product cost is often offset by reduced development complexity and faster time to market. Common devices suitable for an NCP architecture include gateways, security panels and devices running multiple protocol stacks.



Addressing Design Considerations

These three wireless architecture options help to address the design considerations of embedded wireless applications. The developer must weigh the tradeoffs and advantages of each approach to determine which architecture best fits the application requirements.

System Cost: As we advance from SoC to MCU+Transceiver to NCP architectures, hardware cost often increases. However, increased hardware cost can be offset by lower development and support cost due to reduced software complexity and faster time to market. The NCP architecture is typically the lowest cost solution to develop and maintain because the wireless software is isolated from the rest of the system so it can be upgraded without disturbing the rest of the application code. As the application complexity increases, the developer should consider NCP architectures to address the risk of higher total cost of ownership that could result with the SoC and MCU+Transceiver options.

Product Size and Design Constraints: While an SoC offers integration of the MCU and radio and generally results in smaller size, there may be tradeoffs in functionality that are required by the end product. In many cases the size advantages of an SoC may be outweighed by the feature requirements of the end product. If the applications requires a large host processor, a transceiver may have the least impact on the size of the product. When the size of the battery dominates, an NCP may be the optimal choice since handles all of the network traffic without waking the power hungry host processor. Designers must understand the end product requirements when looking at product size and design constraints.

Time-to-Market: Time to market, or “time to revenue,” is an important factor in deciding on how to implement a wireless architecture. Developers must consider many aspects that affect time to market. As noted earlier, software complexity is often overlooked in embedded designs. In a simple unidirectional wireless network, software complexity is usually not an issue, but as a network becomes more complex with the addition of bidirectional communications, more connected devices and especially mesh networks, this added complexity can impact the application. Wireless applications are often timing-constrained, requiring a certain amount of latency, and applications that are processor or interrupt-intensive can cause issues

when running the application on the same CPU as the network stack. NCP architectures can best address these types of issues when the complexity of the application and network increase.

Design Considerations	SoC	MCU+ Transceiver	NCP
Lower Product Cost	✓	✓	
Lower Development Cost	✓		✓
Smallest Hardware Area	✓		
More Product Features		✓	✓
Quickest Time to Market	✓		✓
Simplicity of Design	✓		✓

Conclusion

It is important to consider the optimal wireless architecture when designing wireless embedded systems for the Internet of Things. Making the right decisions up front will help minimize the overall system cost, product size and time to market. Simple applications can benefit from the lower cost of an SoC architecture, while complex applications may require an NCP architecture to reduce the total cost of ownership. Whatever architecture is chosen, broad portfolios of wireless ICs, MCUs and SoCs are available to meet the designer's needs. In addition, comprehensive hardware and software tools, reference designs and protocol stacks are available to simplify the effort of adding sub-GHz wireless connectivity to embedded devices.

This article originally appeared in the January/February print issue. Click here to read the full issue [1].

Source URL (retrieved on *07/30/2014 - 1:40am*):
http://www.wirelessdesignmag.com/articles/2014/03/implementing-sub-ghz-wireless-connectivity-embedded-devices

Links:
[1] http://digital.wirelessdesignmag.com/wirelessdesignanddevelopment/may_june_2014#pg1