EMBEDDED IN THIN SLICES

The Internet of Things (Part 1)

Options for Connecting Wirelessly to the Internet

COLUMNS

Over the years, Bob has seen a few his customers who had struggled with wired connectivity switch to mobile broadband modem-based connectivity to the Internet. This month, he covers four ways that he has connected customers' products to the Internet via cell modem technology.

By Bob Japenga (US)

As I started writing this column, I decided to Google the "Internet of Things" (IoT) and see what comes up. Simply stated, the IoT is the world of information connectivity. It is a world where Things are connected in some way to the cloud of available information via the Internet. The first article that came up in my search was published eight hours earlier in the day. BBC news reported the availability of a starter kit produced by both IBM and a major British manufacturer of an ARM Cortex-M4 processor.[1] The idea behind this offering was to provide a platform to spark people's imagination about what they can do with the IoT. Market research sources quoted in the article predicted that over 5 billion devices will be "on-line" by the end of 2015. There will be up to 20 billion by 2020. Our thin slice of experience says that if the 2015 estimate is correct, the 2020 estimate is very low. This is because, in our business, the IoT is exploding.

In the last two years, many of our existing customers are trying to turn their products into "always-connected" devices. They want their devices to be one of the Things in the world of the IoT. They want to send logs for diagnostics. They want software updates. They want to provide real-time interaction with their devices. They want to provide their users with more information and more control. So I think this is going to explode worldwide.

As I drilled down into the BBC article, I saw that to connect the starter kit to the Internet you needed an Ethernet cable, a router, and an Internet Service Provider (ISP). Once those were in place you could connect to IBM's cloud server. Ethernet cable? That's not what I think of when I think of the IoT. I think of wireless connectivity. I think of toilets that monitor the water for blood and wirelessly send me an email that I should make a doctor appointment. I think of my basement floor notifying me that it is damp while I am away on vacation. I'm thinking of my pill bottles at home letting me know that I already took one pill at work. We are not going to be connecting our mousetraps and sprinklers to the Internet via Ethernet. It will be wireless. We could use wires. But we won't.

We have seen four of our customers struggle with Ethernet or other wired connectivity over the years with their products. All of them have gone to a wireless connection. And by wireless, I am not talking about ZigBee, Bluetooth, or Wi-Fi. I am referring to a mobile broadband modem-based (also known as a cell modem) connectivity to the Internet.

So, this month, I would like to briefly

introduce the four ways that we have connected our customer's products to the Internet using cell modem technology. Each customer has totally different requirements and cost sensitivities. There are many other options. But as always, we will take this in thin slices. In coming months I will tackle other issues dealing with the Internet of Things.

EXTERNAL CELL MODEM

Our first foray into the IoT was with a customer who had a device that went into homes and small businesses. The device had to be located in a specific location in the home. The connection to the Internet was through Wi-Fi on our device to the customer's router (see **Figure 1**). This did not work well. We had problems with Wi-Fi drivers. We had problems with Wi-Fi connectivity in the home. The router was not close enough to the device and neither could be moved. We had customer's routers that would lock up. We didn't have control of the router to keep our data flowing to the cloud. We had trouble with customer not maintaining their ISP. Demand for the product was good but there were problems with this approach.

The next option was to use Ethernet over Power (EoP). An Ethernet cable went from our device to the customer's router via the power lines (see **Figure 2**). This kind of worked. But we still had issues with the customer's routers, the customer's Internet Service Provider (ISP), and even the EoP devices themselves. The EoP devices are complex and can lock up. Since they are powered when they are plugged in, there was no way to remotely reset them. Another problem we had was that sometimes they didn't work very well if the router and the device were on different legs of the 120-VAC line. How well they work depended upon how far they were from the transformer on the pole. We had thousands of units that needed 24/7 connectivity. We couldn't afford having the customer resetting their router or plugging and unplugging their EoP.

All of these were relatively low-cost solutions ($30). But they relied on our customer. The next step was to put an external cell modem. We used a Spider SA-G by Enfora. This was expensive and added the cost of the monthly data plan. But it was incredibly versatile and eliminated the need for access to the customer's router and ISP. It was pre-certified by the FCC and the cell network provider (AT&T and Rogers). It was extremely easy to interface to. We provided power and an RS-232 serial interface (we had one spare in the device). Our internal software had Linux in it so it supported PPP (see **Figure 3**). For our purposes, our application did not need to change. The infrastructure of PPP and Linux gave us a seamless connection to the Internet.

One caveat: Once deployed by the thousands, the Spider was found to lock up and not talk to our device. The only solution was to power cycle the device. A service call! Ouch. We retrofitted a little USB power control to overcome this problem. Memo to self: Always give your designs control of the power (not soft power) to any moderately complex device if you want 24/7 operability. Most complex devices we interface with have software in them that can and will lock up sometime. Only a power cycle will recover the unit.

There are a large number of pre-certified cell modem devices that are relatively inexpensive (less than $100). In addition to the Spider we have used a stand-alone device from Sierra Wireless. We have used a PCB module from Multi-Tech. You can add them to your device and not worry about carrier, network, or FCC certification. It's done for you.

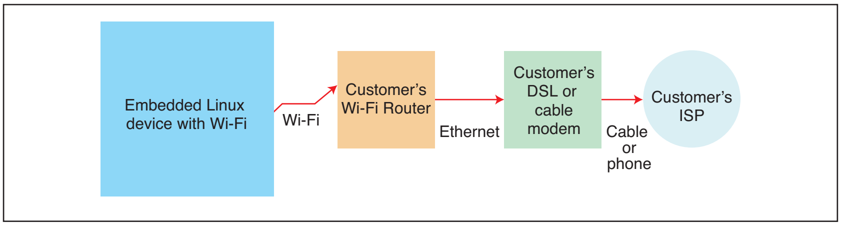

FIGURE 1
Need Caption

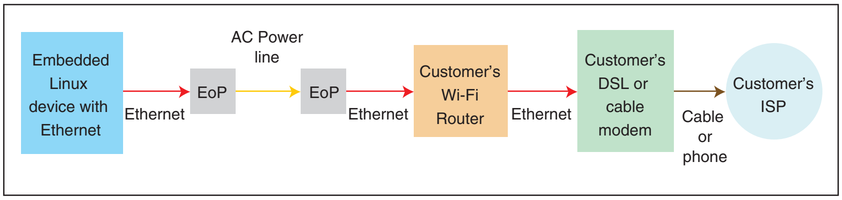

FIGURE 2
Need Caption

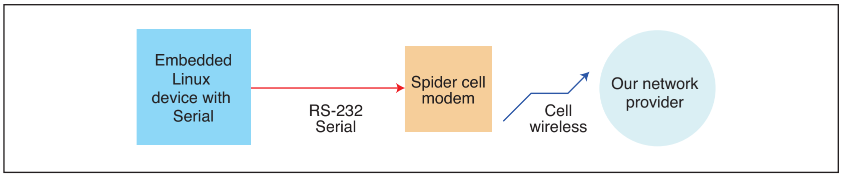

FIGURE 3
Need Caption

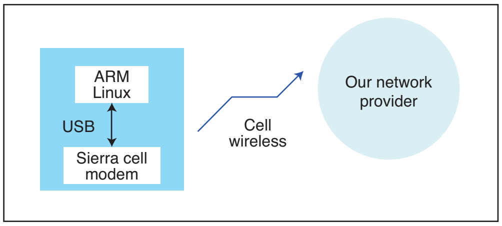

FIGURE 4
Need Caption

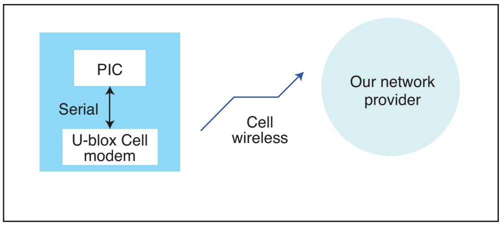

FIGURE 5
Need Caption

ABOUT THE AUTHOR

Bob Japenga has been designing embedded systems since 1973. In 1988, along with his best friend, he started MicroTools, which specializes in creating a variety of real-time embedded systems. With a combined embedded systems experience base of more than 200 years, they love to tackle impossible problems together. Bob has been awarded 11 patents in many areas of embedded systems and motion control. You can reach him at rjapenga@microtoolsinc.com.

Internal Cell Modem

This device became so successful that the customer started looking for ways to cut costs and make even more money. The cost of the Spider was an easy target. Why not put a cell modem chip in our device? They are much cheaper than the pre-certified modules or stand-alone units. We chose a Sierra Wireless SL8081 communicating over the AT&T network. We used the USB interface to the chip and used PPP to connect to the device (see **Figure 4**). Again, Linux made the connectivity both flexible and relatively easy. Sierra had the necessary driver and it played with all of the versions of the Linux kernel we tried. We went through FCC and PTCRB certification. This is a story in and of itself. The customer cut his recurring cost in half. Product sales dramatically increased. He sold more in the first year than he did in the previous three years.

INTERNAL CELL MODEM WITHOUT LINUX

Most of the devices we design don't support Linux. They didn't require connectivity to the Internet when first designed. But now, the demand to be first on the block with wireless connectivity in your particular field is putting immense pressure on manufacturers to make their device part of the IoT revolution. Or the competition is already there and you must play catch up so as to not lose your edge in the market. We have customers in both positions.

Most of our non-Linux designs use Microchip Technology PIC microcontrollers. It is not uncommon for our more complex designs to use as many as 10 PIC microcontrollers. But when I have a successful design and a successful product that is making our customer a lot of money, I cannot just redesign the entire processor infrastructure to give it cell capability. How do I provide wireless connectivity in this device?

One thing we learned from our experience with the SL8081 was that it had a complex operating system inside it. In fact, some of the Sierra parts let you write your own applications inside their cell modem chip. You can create your own Thing to be part of the IoT using just their chip. But in our case we couldn't design our device around the Sierra chip or actually any other manufacturer's cell modem. But we could use some of the complexity of the chip to simplify our interface.

We chose a u-blox family of modules (LISA-C200) to add to one of our designs. It had both a USB and serial interface (so did the Sierra SL8081). It provided AT commands that allowed us to communicate over the Internet using either HTTP, TCP/IP, or UDP. How cool is that? With one simple AT command, my PIC serial interface could connect to a URL, authenticate, and command an HTTP POST or HTTP GET command. All with about 10 AT commands. And the modules are relatively inexpensive (see **Figure 5**).

So using our tiny little PIC, we were able to add a single module with an antenna and be able to send HTTP, TCP/IP or UDP protocol data over the Internet. (Refer to Figure 4. Again, we had a spare serial port.) We send data logs to our server. We have a real-time interactive mode where a remote tech support person can actually see what is on the LCD graphics screen 12,000 miles away. We can update software from the cloud for three of the four PIC processors in the device. All with just a handful of AT commands. And just for kicks, the module gives us about a meg of flash disk storage also accessible through the AT commands. Wow!

Since the module is embedded into our device, even though the module was certified, we were required to get FCC and Network certification for our device. On this project we chose Verizon as the network provider. As a technology we chose to use 3G instead of 4G for cost reasons and the fact that the units do not have a 10-year life.

MIFI SOLUTION

We have one customer who has a very expensive device that has Wi-Fi built in. It

circuitcellar.com/ccmaterials

REFERENCE

[1] "Internet of Things Starter Kit Unveiled by ARM and IBM," BBC, www.bbc.com/news/technology-31584546.

SOURCES

SL8081 Module
Sierra Wireless | www.sierrawireless.com

LISA-C200 Modules
u-blox | www.u-blox.com

is a Linux-based system with both Wi-Fi and Ethernet connectivity. In most cases, the user has three or more of these units. The original idea was that this device would be used in a home or office. The devices would connect to a Wi-Fi router in that office for Internet connectivity. Even before we deployed this, we ran into snags getting onto the Wi-Fi networks or onto the Ethernet networks in the homes and medical institutions. We were experiencing déjà vu all over again. Our device became dependent upon the users' Wi-Fi router, modem, and ISP.

In this case, the mobile Wi-Fi hotspot (like the MiFi marketed by Novatel) solved the problem. The home or medical institution would have one mobile Wi-Fi device and multiple units would connect to the Internet wirelessly. Since the devices are going to be deployed in both the US and Canada, the network provider needed to be chosen to support that. But mobile Wi-Fi hot spots are available from a wide variety of network providers. And they are relatively inexpensive compared to the cost of our device.

Basically, this solution (see **Figure 6**) replaces the need to interface with the home or office's router or ISP and sets your product free from any problems this may bring. And in this special case of multiple units, we don't need to have separate cell modems (separate data plans, etc.) in each device (which take significantly more power than Wi-Fi).

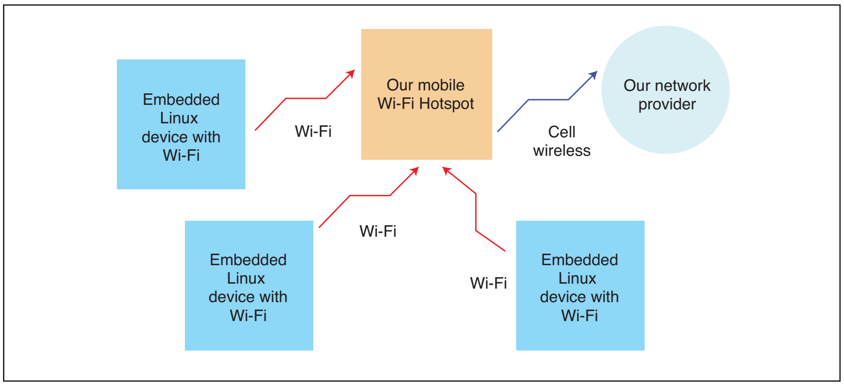

FIGURE 4
Need Caption

CONCLUSION

I predict that the number of devices interconnected on the Internet will grow more than four fold in the next four years. This month we looked at some of the ways your devices can become one of those Things connected to the Internet. Next time, we will look at more of the issues required for making one of your designs a member of the IoT community

COLUMNS