RIGADO

White Paper

A Rigado Best Practices Guide

De-risking IoT Wireless Design

Leveraging certified modules and rapid iteration

Foreword

When we partner with a new IoT client at Rigado, the first question we ask is "What will make this product unique in the marketplace?" It's unlikely to be the radio, but too many companies underestimate the time and effort required to build a robust wireless infrastructure before they even start the challenging process of developing their unique, category-killing product.

This paper covers two key strategies for IoT wireless design: leveraging pre-certified modules to reduce risk and time to market and planning for continued iteration via remote updates.

Low power IoT wireless design is challenging because the very concept of where a connected product begins and ends gets stretched across hardware, firmware, local mesh and device operations up in the cloud. Add in IoT's constraints around size, unit cost, mass distribution and power consumption, and it's clear that for a successful product, every aspect of the wireless design must be carefully considered and optimized.

Having successfully deployed hundreds of wireless projects across consumer, commercial and industrial markets, we are happy to share our perspectives on how to get products to market as quickly as possible with maximum flexibility for the future and minimum risk to both schedule and budget.

Who should read this paper and why

CEO	Read this paper to learn about the business opportunities of remote updates allowing "planned iteration" and how to get to market quickly while minimizing risk to your company.
Product Manager	Read this paper to understand the risk factors of wireless design and how to design for product roll-outs that scale massively with minimal risk. You will also learn how thinking of connectivity as a platform for a family of devices can change your ROI calculus and how best to evaluate if and when to migrate from modules to chip-down designs.
Design Engineer	Read this paper for specific best practices on how to avoid the most common mistakes when designing robust infrastructure for over-the-air updates.

Designing for iteration

One of the biggest advantages of IoT is the ability to update configurations or add new functionality after the device has left the factory, but you have to do some work upfront. For example, a well designed bootloader enables iterative updates, but requires a robust process for managing those updates in the field. You also need to mitigate the risks of frequent iteration by minimizing downtime and designing fail-safes against bricking the device during updates.

A Smart Bootloader

All wireless SoCs will come with some stock Device Firmware Update (DFU) functionality, but adding a couple of features to your bootloader will cut your update risk factors by 90%.

- First, the bootloader DFU should exist outside of the main application firmware. This ensures that the device can start up regardless of the condition of the application firmware.
- Second, you should always include a double buffer. This allows you to verify updates before they're loaded. Be sure that the buffer is large enough: 500K should be sufficient.
- Finally, end-to-end encryption protects you from both theft of IP and malicious code being inserted onto your devices. In this case, end-to-end means encrypting the update package all the way from a system you totally control to the IoT device itself. Don't rely on the Bluetooth connection's default encryption; instead you should share the keys at the point of manufacture via a hardwire connection at the factory. This ensures that even if the connection is compromised, no rogue firmware can be loaded on the device.

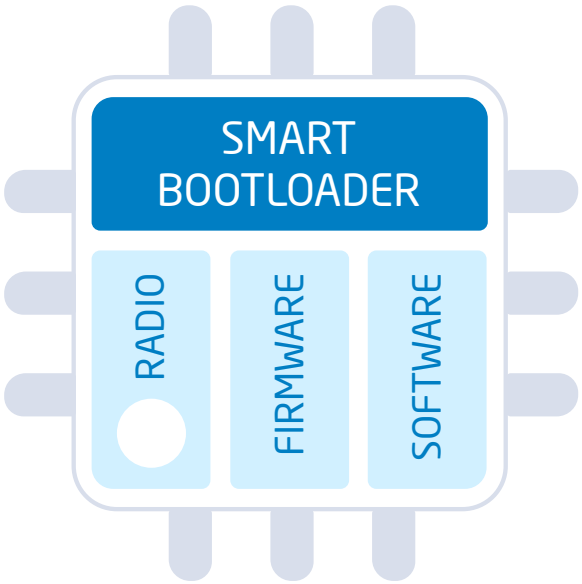

Figure 1. Smart Bootloader Make sure your bootloader is independent of your firmware.

PROTIP

To save time and minimize the risk of human error, any ongoing update management process should be as automated as possible, so make sure you have a robust and well-documented API.

Unique Key Encryption: Only the paranoid survive

If someone has physical access to a device and they are determined enough, you should assume the device can be hacked. So the question to answer is, "How exposed am I if someone hacks one device?"

Rigado uses 128-bit AES-EAX encryption to secure the transfer of firmware images to our modules, but we also employ device-specific encryption keys to reduce the threat of a single successful hack turning into a major Mirai botnet-level catastrophe.

This requires some extra development work upfront (or use a module that has secure bootloader already integrated), but with your company's entire reputation at stake plus the potential expense of a recall, the risk-reward calculation skews highly in favor of playing it safe.

Managing updates at scale

To update the devices in the field, you will need a central device operations application that is in communication with each device. This cloud application will probably have other functionality (e.g. processing data from device sensors or issuing instructions to the device), but we're focusing here on update management.

The first requirement of the device operations app is to identify and group devices. Your device operations system needs to track which unique devices have what features enabled, and ideally where they are deployed (e.g. to roll out updates at off-peak hours time zone by time zone).

By adding support for device 'tags' in both your firmware design and your cloud app, you can assign specific devices to one or more groups, which makes feature management a lot easier. Tags can help manage hardware variations too, making it easier to deploy different versions of an update to support different hardware without having to add serial number-level logic into your updates.

Finally, having granular command of device selection enables staggered roll-outs, which dramatically reduces the risk of catastrophic update failure. A staggered roll-out updates a small percentage of devices first, waits for an acknowledgment that the update went well, and then rolls out the update in larger and larger cohorts, each time checking for a heartbeat before proceeding to the next cohort.

Minimize your risk with differential updates and roll-back failsafe

Consumers regularly complain about automated updates interrupting their use of their digital devices. In industrial applications, you can actually measure the cost of downtime due to updates. In either case, downtime translates to dissatisfaction. At the other end of the frustration spectrum, there is the possibility of a bad update totally bricking the device.

Rigado has developed a unique solution for minimizing the size and time of over-the-air (OTA) updating through the use of differential updating. A differential update contains only the changed parts of the firmware, not the entire image.

There are different degrees of differential: you might only update at the library level, or with the right firmware in the bootloader, you can diff down to a single line of code. At Rigado, we often reduce the update size by 80% or more when we use differential updates.

What's more, smaller update packages means less data, which also means less power required for transmitting updates.

Differential updating also results in a better customer experience as downtime during update can decrease from minutes to seconds. This can really add up for devices where updates are pushed frequently, especially at the beginning of the product lifecycle where the owner is evaluating the total cost of ownership of the device (which includes the frustration factor) versus its value.

Good update design includes building in failsafes. Add logic to your bootloader DFU to do a checksum on the update and verify it transmitted successfully before you write over the firmware image. Then move a copy of the original firmware into the buffer while the device reboots. If the new firmware fails to boot successfully, then have the bootloader swap out the firmware images and roll-back to the earlier version that was working.

You will need to add logic so you don't get caught in an update and roll-back loop, and make sure the bootloader reports back to the device management application with details about the failed update. At that point your operations software can send instructions to try again or alert you to fix the issue and start again with a new version.

Figure 2. Managing updates Your Device Operations cloud app communicates with and manages devices via a gateway.

Getting to market faster & planning for scale

If time to market is important to your product's success, then leveraging a pre-certified module should be your default strategy. While pre-certified modules carry a slightly higher per-unit price than a custom designed wireless (or 'chip-down') solution, they typically save far more in design & certification costs for new products. A good rule of thumb is to leverage the speed and implementation advantages of module solutions until you exceed 100,000 units, and then evaluate migrating to a custom chip-down solution.

This section will look at the time savings of leveraging a module solution and the advantages of designing your connectivity solution as a platform for all future members of the product family.

De-Risking Part 1:
Testing the product value proposal

Every product manager's goal is to bring their product to market as quickly as possible at the lowest possible cost so they can test their market assumptions and see if they truly have a winner.

Wireless modules often allow you to bring connectivity to a product for less than half the cost of designing your own wireless solution, so you can evaluate the product's potential before laying out a huge investment in wireless engineering expertise and global certifications.

With so much engineering work completed in advance, you get to market faster, giving you more time to iterate based on customer feedback and first mover advantage over your competitors.

De-Risking Part 2:
Minimizing regulatory risk & certification costs

When you design your own radio, you are considered an "intentional radiator" by the FCC and as such must comply with CFR 15.249. These requirements have hundreds of tests which can take months and cost tens of thousands of dollars. What's more, changing the antenna, ground plane, RF firmware, board layout or even the casing can all trigger re-certification. Getting these all perfect the first time is unlikely, so if you decide to do it yourself then you should factor in at least two rounds of certification before going to market at scale.

In contrast, leveraging a pre-certified radio module makes you an "unintentional radiator" which has a much easier certification. You also have the benefit of a market-tested design, where any re-designs and re-certifications will be the responsibility of the module vendor.

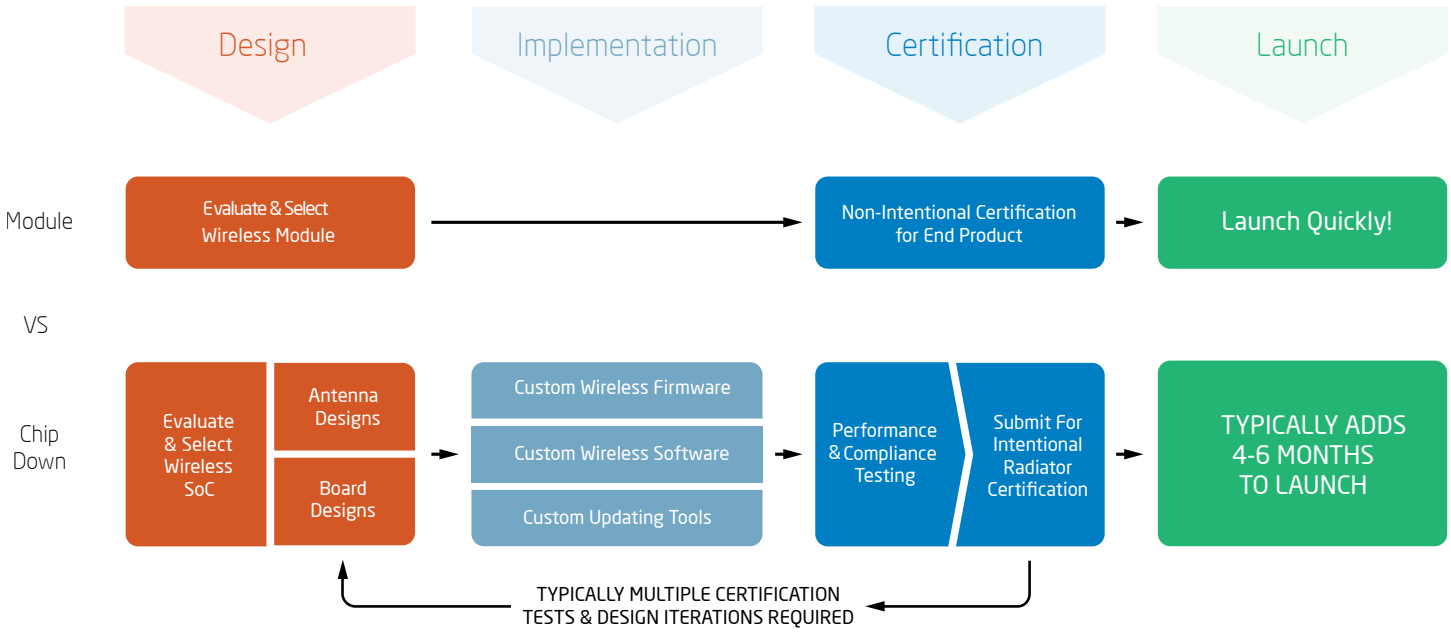

Figure 3. Timeline of two engineering teams Modules reduce risk, cost & time to market.

Connectivity as a platform

Given the cost and complexity of designing a wireless connectivity solution, it makes sense to reuse your designs as often as possible.

When we're talking with our clients, we strongly recommend they think of connectivity as a "platform" for an entire family of products rather than as a one-off design exercise for a single device. With a platform in place, new product family members can get to market with near-zero engineering time for connectivity. Sharing parts across a family also makes it easier to buy in bulk which reduces BOM cost and inventory management, always a concern when scaling.

When looking at connectivity as a platform, taking steps to "future proof" that platform is often a wise investment. For example, you may want to select a dual mode wireless SoC that supports both Bluetooth and Thread, even if only one will be used initially. You will also want to ensure you have adequate RAM, memory, sensors and I/O for future apps.

Finally, you should consider where you'll be selling as each country has its own certification requirements. At Rigado, our modules are all FCC, IC and Japan certified, and CE and Australia/New Zealand compliant.

Making the move from module to custom design

Modules incorporate the tools you need for fast iteration, but they're typically a bit more expensive, so at some point you may need to make a decision about if and when to move from modules to a chip-down design.

From an engineering perspective, the process is just a part swap: you can use the same SoC in your custom design that the module uses, and you can probably license any firmware or libraries that the module provider included. But as the designer of a radio transmitter, you will now need a different type of certification; and by stripping out extra features in favor of a "bare-bones" design, you may be limiting your options to add new features in the future.

Evaluating the move from Module to Chip Down
Are expected BOM savings > cost of design, test & certification?

You should also consider engineering opportunity cost. Say a chip-down design saves $1.25 per unit sold, but it will take 4 months of engineering time to get there.

Would you generate more revenue if the team used that time to work on new innovations that leverage your company's unique skill set?

Ultimately, migrating to a custom chip-down design may be the right move once a device reaches volumes of 100,000 or more. That's also a transition that your module provider should be able to facilitate.

PROTIP

Make sure your module provider has a well documented process to migrate your firmware and app code to a custom design. For example, will they license any proprietary firmware to your custom design?

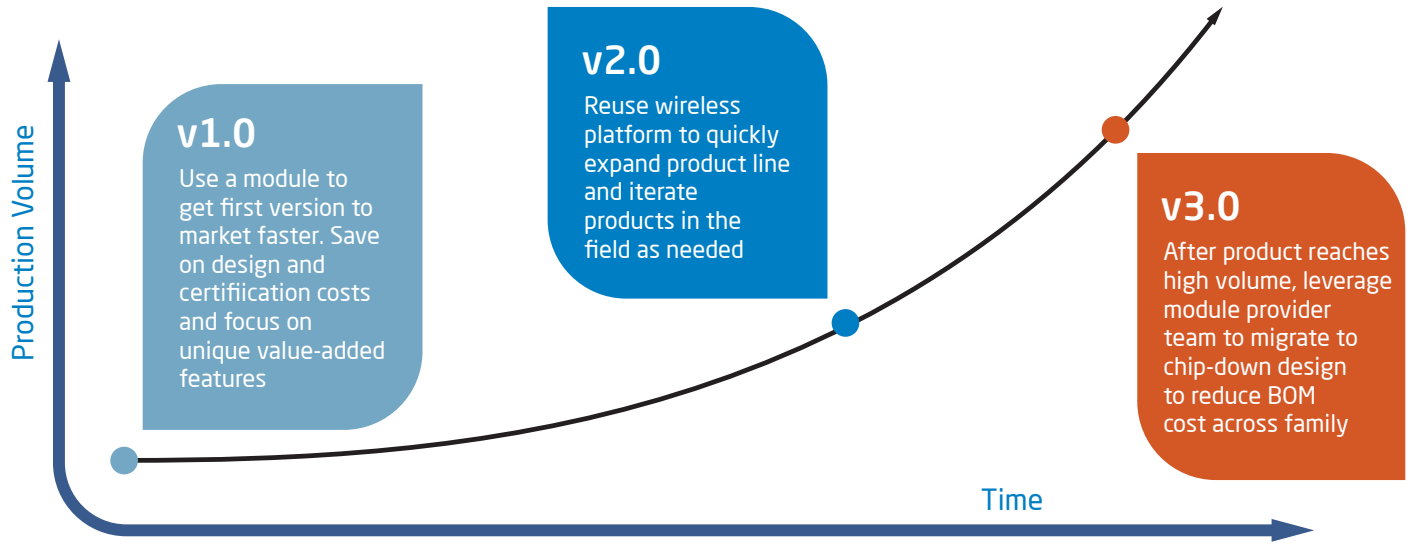

Figure 4. Getting to market & when to scale: Look at sales volume to decide if and when to move to a custom wireless design.

Summary: Wireless design is challenging, but best practices help

IoT revolutionized product design with the ability to update devices remotely, opening the door to a design philosophy of planned iteration and ongoing revenue models with monthly subscriptions.

It's all about your product's unique value-add.

As long as you design with security in mind from device firmware to cloud operations, and you invest resources in robust over-the-air updates, your products can continue to evolve long after they've left the factory, generating new revenue streams and delighting users for years.

As engineers ourselves, we recommend you consider a module-based design or look at partnering with a wireless design house. It allows your team to focus on what makes your product unique, minimizes risk and dramatically decreases time to market.

To take a closer look at the most likely risk factors facing your wireless project, contact us at info@rigado.com.

About the author

Justin Rigling, *Co-Founder and Chief Technology Officer, Rigado*

As CTO at Rigado, Justin has designed literally hundreds of wireless modules for every application imaginable, from bike computers to heavy industry to golf course watering sensors.

Prior to co-founding Rigado in 2010, Justin was a design engineer at Garmin International. He specializes in low-power wireless systems, and holds a B.S. in Electrical Engineering from MIT.

Justin is invited to speak regularly at industry events and he is a member of the Bluetooth Special Interest Group. At 124 miles, he also set the Guinness book world record (alongside Rigado co-founder Ben Corrado) for the world's longest wi-fi connection.

+1 (866) 674-4236
www.rigado.com