Considerations in the Build vs Buy Decision-Making Process for SDRs

EPIQ SOLUTIONS

Table of Contents

The flexibility and enhanced performance offered by software-defined radios (SDRs) in RF transceiver applications is driving an increased demand for their use across many industries such as defense, telecom, aerospace, and government. Coupled with the rise of 5G, it is no surprise that the projected global market for SDRs is expected to top $39 billion by 2027. As market demand grows for SDRs, many engineering organizations are wrestling with the age-old question: build or buy? With SDRs, the answer is not straightforward.

Even with industry standards like SOSA and CMOSS helping to move SDRs toward a single architecture while reducing size, weight, and power consumption (SWaP), the flexibility promised by software keeps pushing technical and hardware requirements for SDR modules to handle ever-wider frequency ranges and increasingly complex protocols in the development, deployment, testing and demonstrations of wireless systems. SDRs at their simplest are complex digital circuits that provide a combination of RF, data conversion, and digital signal processing. Building them has traditionally not been easy, but evaluating a commercial-off-the-shelf (COTS) SDR is no walk in the park, either.

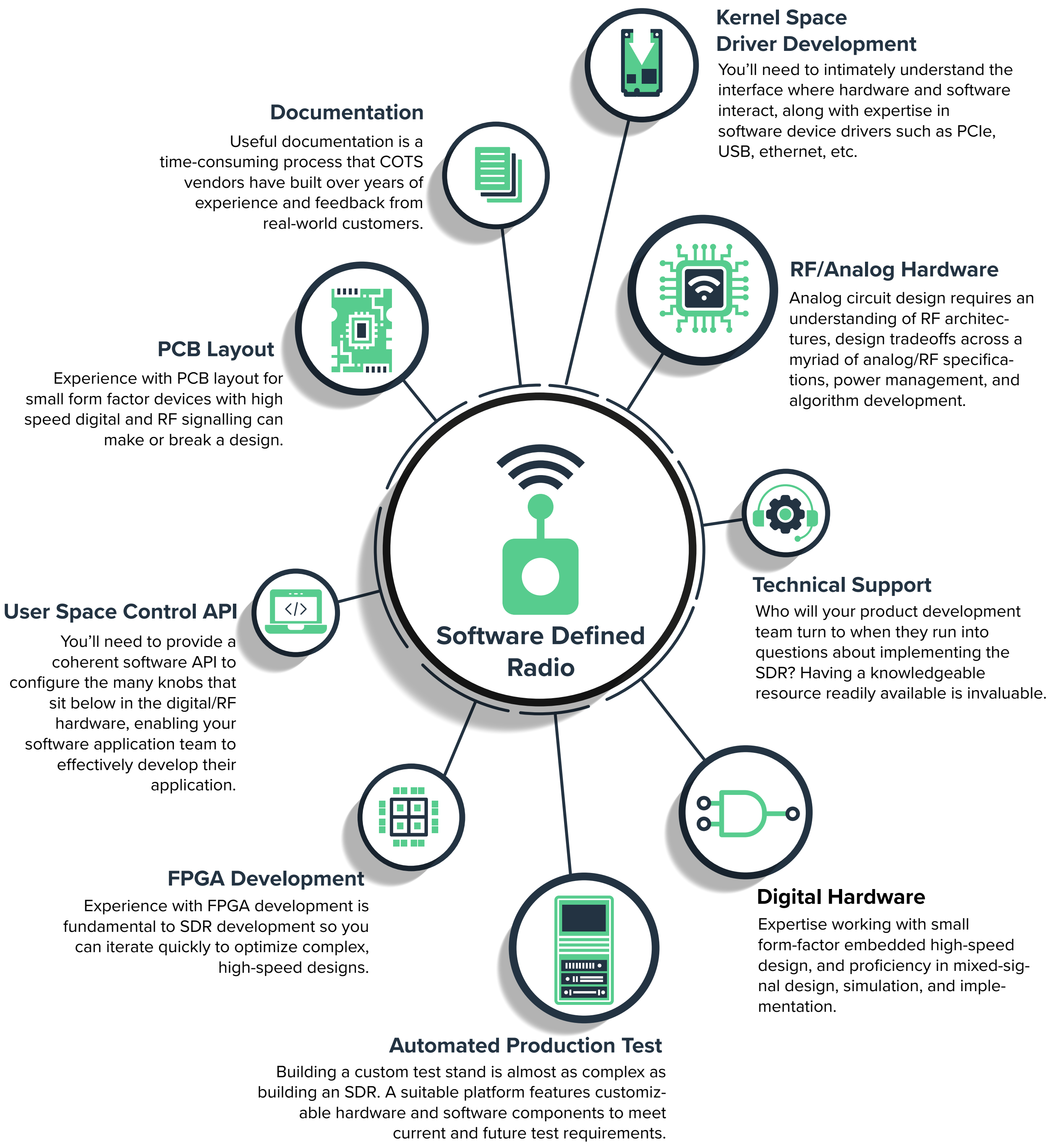
With numerous considerations and decisions needing to be made, from hardware selection to software development and programming, from device configuration to testing, the path to arrive at a final build or buy answer for an SDR can be complicated. This white paper explores these and other considerations throughout the RF transceiver development process to help your organization decide which avenue is best suited for your needs.

Getting Started with SDR Development

It is easy to underestimate the amount of time and the breadth of technical skills required to develop an SDR for a particular use-case. The advances in both RFIC technology and FPGA technology have definitely simplified the development process, but this doesn’t mean that development is simple. It is not uncommon for companies to have the engineering talent on staff to develop a flexible SDR transceiver on their own. The question is whether or not this is an effective and appropriate use of engineering resources.

For a snapshot of just how much goes into SDR development, the diagram on page 2 provides a high-level overview of the wide range of expertise required.

To help make an informed decision on build versus buy, let’s look more in depth at the development effort and skills needed for each.



Kernel Space Driver Development

You'll need to intimately understand the interface where hardware and software interact, along with expertise in software device drivers such as PCIe, USB, ethernet, etc.

Documentation

Useful documentation is a time-consuming process that COTS vendors have built over years of experience and feedback from real-world customers.

RF/Analog Hardware

Analog circuit design requires an understanding of RF architectures, design tradeoffs across a myriad of analog/RF specifications, power management, and algorithm development.

PCB Layout

Experience with PCB layout for small form factor devices with high speed digital and RF signalling can make or break a design.

Technical Support

Who will your product development team turn to when they run into questions about implementing the SDR? Having a knowledgeable resource readily available is invaluable.

User Space Control API

You'll need to provide a coherent software API to configure the many knobs that sit below in the digital/RF hardware, enabling your software application team to effectively develop their application.

FPGA Development

Experience with FPGA development is fundamental to SDR development so you can iterate quickly to optimize complex, high-speed designs.

Digital Hardware

Expertise working with small form-factor embedded high-speed design, and proficiency in mixed-signal design, simulation, and implementation.

Automated Production Test

Building a custom test stand is almost as complex as building an SDR. A suitable platform features customizable hardware and software components to meet current and future test requirements.

Figure 1: The diagram above provides a high-level overview of the different engineering disciplines required to develop, manufacture, and support an SDR.

RF Hardware Engineering

The advancement of RFIC technology has single-handedly been one of the most enabling factors in recent years to facilitate SDR development. The integration of RF synthesizers as local oscillator (LO) sources, quadrature mixers, configurable baseband filters, and data converters in a single chip helps to minimize the “black magic” of RF circuit design. But even with these integrated RFICs, there is often a significant amount of RF engineering still required between the antenna and the digitized samples. Low noise amplifiers, RF filtering, careful power supply design, pristine reference clocks, and more are still needed for a successful design that is ready to be used in the real world. It can be easy to assume that the RF performance of two solutions using the same RFIC will be identical, when in reality the supporting RF hardware around the RFIC makes a significant difference. And of course, it usually takes a few iterations of the RF hardware design to get to the desired level of performance.

Digital Hardware Engineering

In addition to integrating the RFIC hardware, there is a substantial amount of digital hardware engineering needed for SDRs as well. At the start of this process, a digital hardware engineer needs to make the following major considerations:

- FPGA device selection (almost all SDRs have an FPGA these days)
- Boot memory for the FPGA
- Local compute element (CPU, memory, etc.), if needed
- High-speed digital interfacing, which often reaches into the many GB/sec range
- Power supplies, which need to be sized to support a wide range of operational scenarios

These items must be decided on before you start developing the FPGA code necessary to do something meaningful with the digitized RF spectrum.

FPGA Development

Field Programmable Gate Arrays (FPGAs) play a central role in creating a flexible RF frontend within SDRs. They provide the high speed, low latency, digital infrastructure needed to enable substantial amounts of processing and data transport. This all starts with the ability to interface to the high speed A/D and D/A converters utilized on the SDR, which often requires management of high speed parallel data busses or SERDES protocols such as JESD204b and JESD204c. The FPGA also manages the high rate of digitized RF spectrum that needs to be carried to a CPU for processing using a transport bus such as USB, Ethernet, or PCIe. Finally, lower speed interface busses such as SPI and I^2C are often managed through an FPGA to control hardware peripherals on the SDR. All of these FPGA functions are required as part of the typical SDR infrastructure even before any actual signal processing takes place. In many cases, there is application-specific signal processing that needs to happen in the FPGA, as well.

Developing the Software Infrastructure

Once the steps are taken to create a flexible RF frontend with baseband A/D and D/A converters that interface to an FPGA with a high-speed transport up to a CPU, the real programming fun with the software stack can begin. There are multiple layers of software required as part of the SDR infrastructure to make this type of system work. An example software/FPGA stack is shown below, based on Epiq Solutions' SDR portfolio.

At the lowest level, there are typically software device drivers that run in kernel space to interface with the transport bus supported by the FPGA. Since an SDR is often dependent on continuously streaming a high rate of digitized radio spectrum to the CPU where the core of the signal processing can take place, these kernel space device drivers are critical in supporting efficient, high-rate data streaming between the FPGA and the CPU. The next level above the kernel space device drivers is a user space API which is typically required to expose a control layer for the underlying hardware. This control layer is the foundation upon which all application-specific software is developed. It allows a user of an application to perform functions such as changing the RF center frequency, reducing the RF receiver gain, setting the A/D converter's sample rate, and starting or stopping streaming of the digitized radio spectrum to the CPU. These types of functions are usually included among dozens of other parameters that must often be exposed to application software. Each parameter needs to have

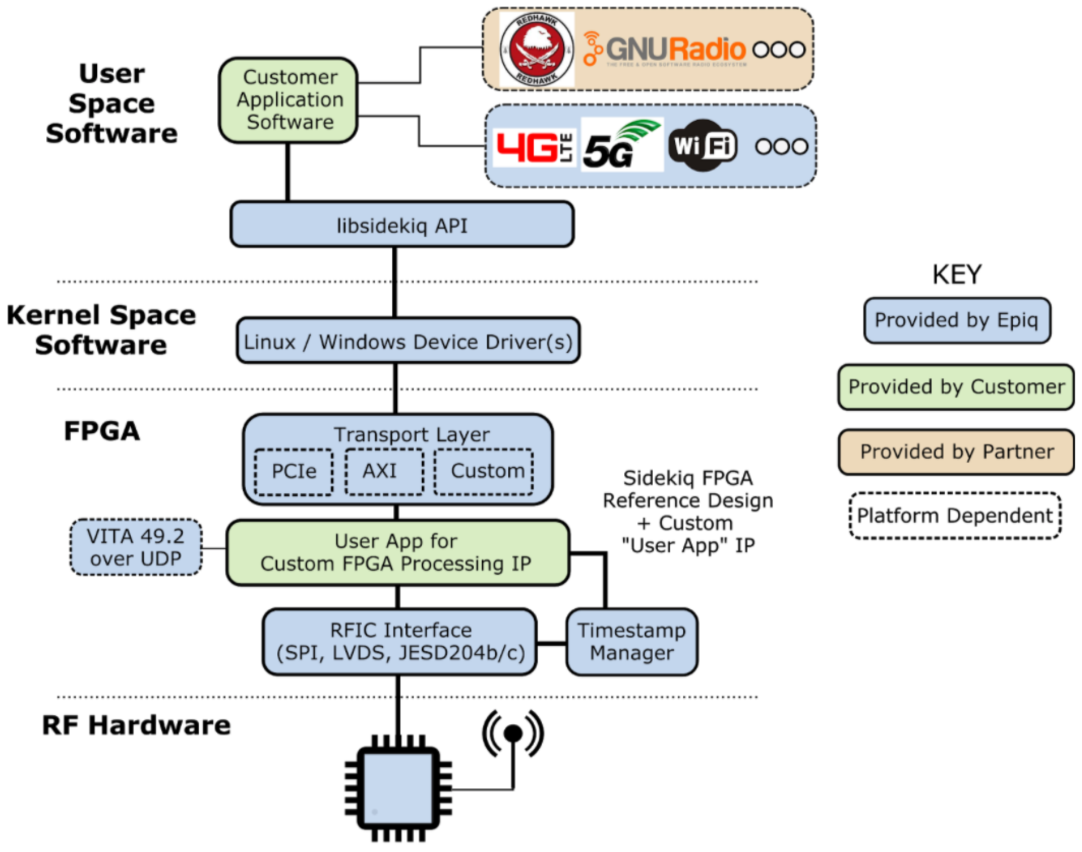


Figure 2: An example of a typical SDR technology stack, based on Epiq's software defined radios

its own set of underlying control functions, which are typically then split between user space libraries, kernel space drivers, and the FPGA.

Of course, all of this SDR-specific software may need to operate on a variety of different host operating systems and/or CPUs. This could include different versions of Windows or Linux, 32-bit vs 64-bit operating systems, and all of the detailed nuances that come along with the host CPU where the software will run. Each option dictates the flexibility of the SDR which ultimately impacts decisions that will ripple through the entire development life cycle to the end application.

Design Iteration to Ensure Optimal Performance

Once all the above pieces of hardware and software are developed and integrated, it is time to ensure the radio performance meets the requirements of the application, which can often take multiple hardware design iterations. This is often the most underestimated part of the process in terms of time and cost. After each iteration that does not result in the desired performance requirements or acheive the size, weight, and power objectives, the process starts over. Each revision typically includes making nuanced adjustments to the RF front end and/or digital hardware to minimize interference and spurious signals. Each hardware iteration typically requires a full cycle of schematic adjustments, PCB layout updates, PCB fabrication, and assembly at a contract manufacturer before finally getting the latest iteration of the hardware back in your hands. This process can take eight to ten weeks per iteration beyond the application development and engineering time required to assess performance each time. Investment in a highly experienced engineering team will keep iterations (and headaches) to a minimum and help keep projects on track.

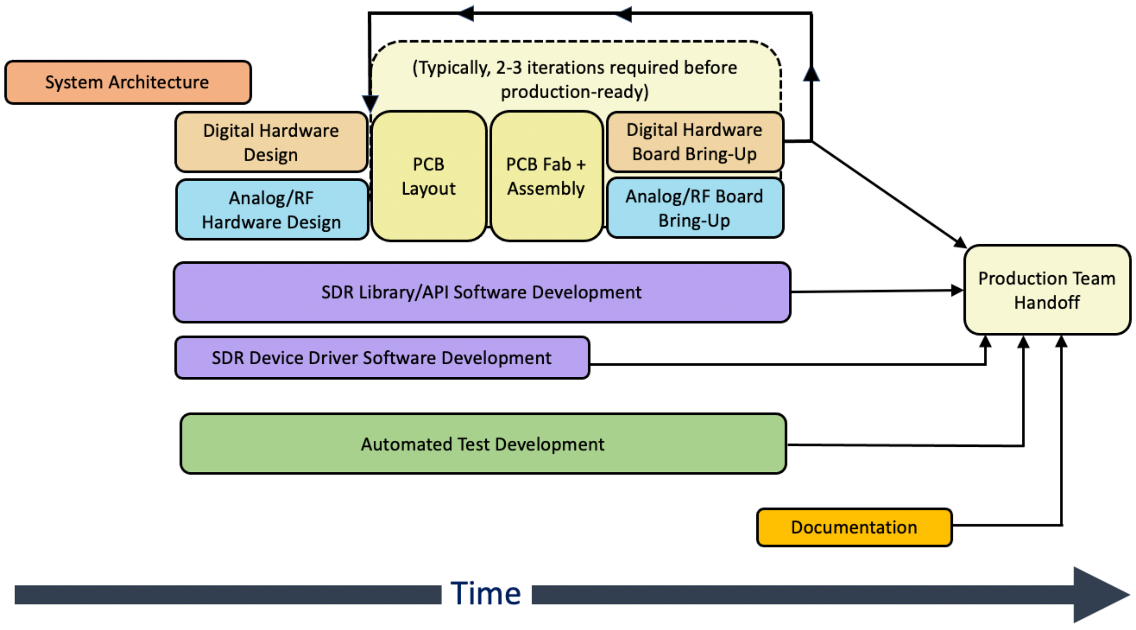

System Architecture
(Typically, 2-3 iterations required before production-ready)
Digital Hardware Design
Analog/RF Hardware Design
PCB Layout
PCB Fab + Assembly
Digital Hardware Board Bring-Up
Analog/RF Board Bring-Up
SDR Library/API Software Development
SDR Device Driver Software Development
Automated Test Development
Documentation
Production Team Handoff
Time

Figure 3: SDR development is an iterative process and how effectively each element is executed will impact the number of design cycles required to bring a design to life.

Functional Test

Upon completion of the final design iteration, the SDR hardware will need to be functionally tested. Ideally, the development of the testing infrastructure is happening in parallel with the development of the SDR along the way. Setting up an automated functional test environment to perform the necessary digital and RF validation of the SDR can be its own significant undertaking. Performing functional testing, even if automated, often requires additional hardware resources to develop the test pods for the SDR, software to control the RF test equipment such as the RF signal generators and spectrum analyzers, and software to measure the performance of the SDR. This testing and validation step is critical to ensure the quality of the SDR module itself, and often requires a team dedicated to only supporting this type of test development separate from the engineering team.

Beyond Development Logistics

It is also important to note that simply having the technical capabilities to develop the hardware and software components discussed above is not enough. For many applications, RF engineers are constantly being asked to improve the size, weight, and power (SWaP) of all components, and SDRs are no exception.

Having a requirement to optimize an SDR for SWaP can extend development time and also further complicate many development stages of the build process for a couple reasons. To start, when building the SDR, the first iteration or two will generally be much larger than what is needed, resulting in additional time and effort required to miniaturize the SDR. Second, there are unique technical challenges associated with RF devices as size is reduced, that will require additional hardware cycles to be built in as part of the optimization process. Careful part placement, component shielding, and PCB stackup management can make a significant difference when it comes to the performance of the final solution.

Supply Chain Reliability and Management

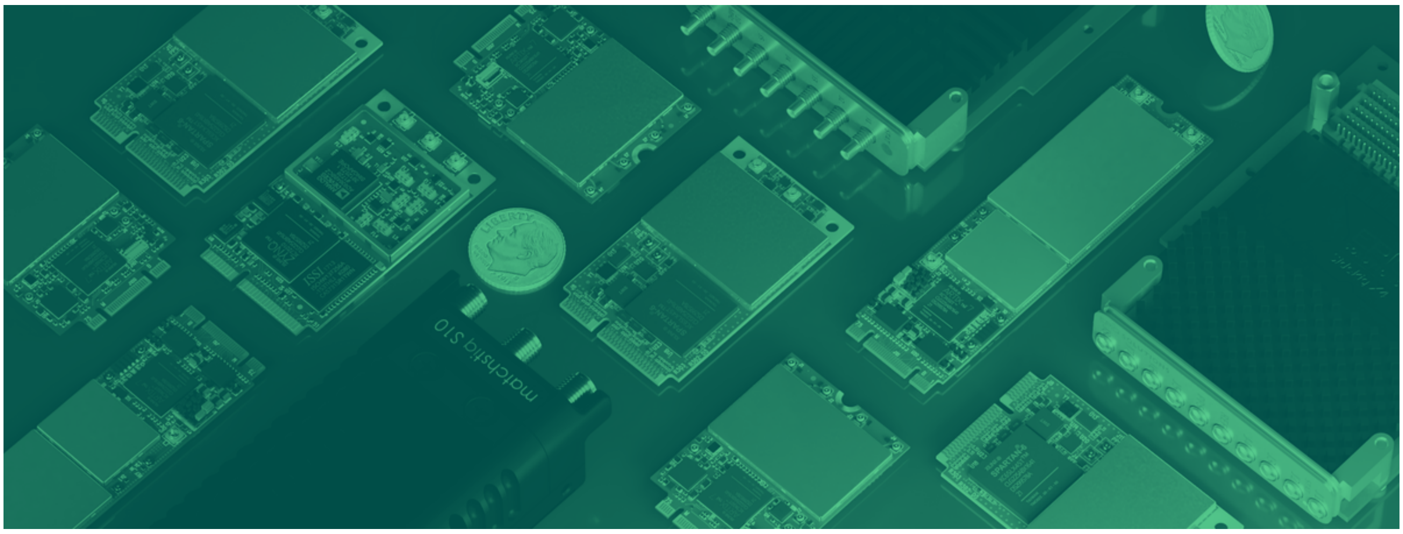
Managing component availability and lead times can also be an unforeseen obstacle when building a custom solution. For example, in 2018, the lead-time for capacitors (yes, a simple passive component) extended up to 50 weeks due to unforeseen, unpredictable market forces including raw material shortages and increased tariffs on steel and aluminum. Compounding these factors was the expansion of electronic technologies into new markets like electric vehicles (EV) and hybrid electric vehicles (HEV). EVs use a lot more capacitors than their gas-powered counterparts, so the global demand for most electronic components, and capacitors in particular, increased demand which then reduced supply. More recently, during the first half of 2021, the semiconductor industry has seen a surge in component lead times from RFICs to FPGAs to simple passive components such as inductors. It isn't uncommon to see lead times in the range of 40-50 weeks, significantly extending the hardware development cycle. Given the current shortage of components, minimal availability of raw material, and dependence on manufacturing in other countries, challenges with component availability is likely to extend well into 2023.

Besides market forces, natural forces can cause sudden and unexpected delays as well. In late 2020, a fire devastated the Asahi Kasei Microsystem (AKM) semiconductor factory in Nobeoka, Japan. The facility was known for producing components for high-end audio devices and high-stability reference clocks often used in SDR development. Production was expected to be delayed for at least six months, potentially crippling global semiconductor supply and forcing buyers to find other suppliers or redesign their applications. A COTS SDR vendor will not be as vulnerable to these events because part of their responsibility is to have inventory on hand to manage production level SDR development, and therefore be in a better position to navigate supply chain shortages.

The Importance of Flexibility and Long-Term Support in the Build Versus Buy Riddle

Now that you have a high-level understanding of what is involved in developing an SDR all the way from chip selection to functional test, it is time to think more in depth about what is feasible for your organization. Putting together a dedicated team with the diverse skills required to develop all the necessary hardware and software components is an expensive and time-consuming proposition, so it may be that heading down this road is clearly not viable. However, it's not always so clear. In general, if the requirements for your SDR are highly specific, if the flexibility required from your SDR platform is not a necessity, or if schedule demands or time-to-market pressures are not a concern, then building an internal team to custom design a software defined radio system may very well be the best option. But if development time is tight or budgets aren't flexible, selecting a COTS SDR should be considered.

Supporting the SDR once it is shipped to the customer is also an important consideration. If an SDR is developed in-house and a product team gets deep into product development, there might not be adequate support for a proprietary SDR if technical issues arise. This applies to the expectations being placed on the product being developed, as well. If you know that the end product will never need or be expected to evolve beyond what it is being designed for, then in-house development of a purpose-build RF transceiver may be perfectly suitable. But chances are, if an SDR is required, it needs to be part of a system that can be enhanced over time as application requirements evolve. One of the primary reasons an SDR would be used in the first place is its ability to be updated and



reconfigured. Without continued innovation or development, an in-house SDR solution will only be as good as the day it was designed. Taking an honest look at the likelihood of shifting requirements can be extremely beneficial for answering the build versus buy question early in the project development cycle. On the other hand, a COTS SDR will have dedicated product teams whose entire jobs are to enhance the functionality of your SDR and provide ongoing support. It is not uncommon for SDR vendors to offer new features as simple software updates as development evolves. This perpetual product innovation can be an easy benefit to overlook when considering building a custom solution, not to mention rigorous support documentation, as well as having a consistent API library and user experience across new radio cards from the same vendor if the need arises for future projects.

Knowing what is required to fully realize a flexible SDR transceiver is important, as is having an idea of where the financial break even point is for going it alone. Our team at Epiq Solutions has been through this process numerous times while developing our portfolio of COTS SDR modules. The financial investment isn't small, extending well into the millions of dollars required in up front engineering/development investment to tackle all of the relevant pieces. Estimating the financial commitment, coupled with the technical expertise required to deliver something that works well, can help you realistically evaluate whether to undertake the development on your own or buy from a trusted vendor.

Finally, time constraints are always a part of the decision-making calculus. An SDR vendor will have the steps we discussed earlier - RF hardware development, digital hardware development, FPGA development, software development, testing, and documentation - down to a science. The time efficiencies realized with a highly-specialized team can be significant when compared to even an experienced company that is starting from scratch.

Selecting the Right Vendor and COTS SDR Solution

Let us assume the big decision has been made to buy instead of build. At this point, you might be thinking the decision-making is at an end. But now is the time for another monumental decision: which SDR vendor and COTS solution to select. If SWaP is also a consideration, the candidate pool of options is fairly small. Evaluating a potential vendor for their ability to support development and evolution of the SDR product is also a priority. Choosing a vendor who deals with standard commercial form factors such as PCIe cards, MiniPCIe cards, M.2 cards, 3U/6U VPX cards, or similar, can radically simplify the interfacing and integration of the end SDR into the intended host system. Further, these commercial form factors typically provide a significant number of options for the system topology in terms of compute resources and peripherals. Standards simplify the system development process, and choosing a COTS SDR aligned to a standards-based form factor can increase the attractiveness of the end product to numerous buyers.

Have Confidence in Your Decision

At Epiq Solutions, developing radically small SDRs that deliver on flexibility, performance, and functionality is one of our core principles. Our portfolio of SDR cards, FPGA reference designs, and software are field-proven to function reliably in mission-critical mil/aero and communication applications. If you find yourself weighing the pros and cons of developing an SDR in house versus

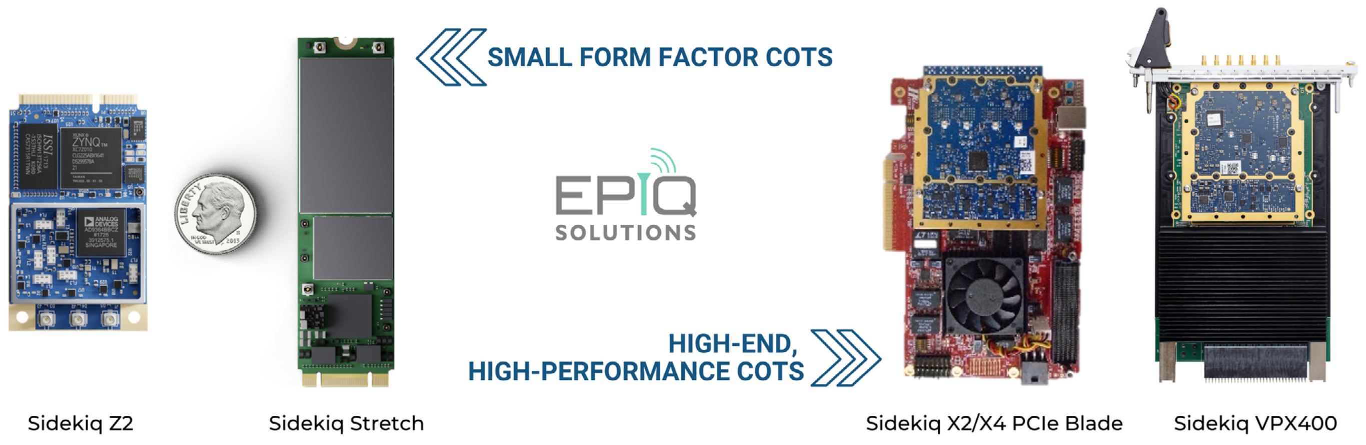


Figure 4: Selecting a COTS SDR from a vendor that has mature, SWaP-optimized and high-performance devices cuts time-to-market and provides peace of mind.

using a COTS SDR from a proven supplier, our team at Epiq Solutions would welcome the chance to discuss your application. You can contact us here to learn more about our SDR portfolio, and everything that goes into making these products possible.

LEARN MORE

EPIQ SOLUTIONS

epiqsolutions.com