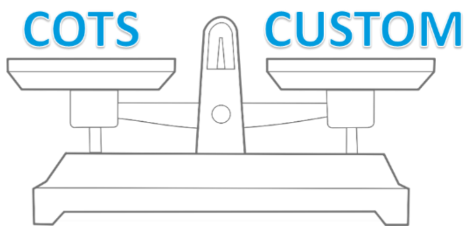
VIEWPOINT
SYSTEMS

COMPARING OFF-THE-SHELF TO CUSTOM DESIGNS FOR INDUSTRIAL EMBEDDED SYSTEMS

CONSIDERATIONS FOR PROTOTYPING AND PRODUCTION

When creating a new product or adding an incremental improvement to an existing one, engineers need a prototype to test the concept design's stability, verify key performance metrics, and vet out any bugs. This paper will address the immediate question that arises: Is it better to buy commercial off-the-shelf (COTS) components or design a completely custom printed circuit board to prototype and build your industrial embedded system?



COTS VS. CUSTOM: THE DILEMMA

The types of products we will be considering for this COTS versus Custom debate will contain the following items:

- Programmable controller
- Analog and digital I/O with possible signal conditioning
- Communication ports of some sort
- Possibly a display (i.e., HMI)
- Possibly some packaging

Obviously, the complete product will contain several subsystems, such as sensors, mechanical components, power supplies, and so on. We are going to focus on prototyping and developing the controller and its I/O because, in our experience, the huge number of controller and I/O options are a source of confusion to many people.

Furthermore, from a business perspective, we also want to be cognizant of unit cost of the end-product and the amount of time before we can start selling the product. If the product costs too much or takes too long to develop, we may spend a lot of time and money with unacceptable payback. Hence, the choice of controller will also be driven by business rather than purely technical decisions.

To weigh the pros and cons of the COTS versus Custom approaches, it helps to understand the options along the COTS – Custom spectrum. Ultimately, your goal is to use prototyping to prove out your product design in advance of creating the product that you will put into production and to do it as quickly and efficiently as possible.

THE COTS APPROACH

COTS is an acronym for Commercial-Off-The-Shelf. A COTS prototype is entirely comprised of items that can be ordered from various vendors. Obviously some integration effort and software development are required to create your product concept: if you could buy your product as COTS, you already have a competitor.

Using COTS speeds development of your prototype since you have less to build, can interface to your sensors and actuators with purchasable signal conditioning, and both write and debug your application in a development environment that offers high-level software development tools. The resulting shortened timeline is traded against either high unit cost or high follow-on non-recurring engineering (NRE). The high unit cost happens is you use the prototype design for your end-product. The high NRE happens if you need to convert your COTS design to Custom after you've vetted the product design.

THE CUSTOM APPROACH

Building a Custom prototype involves selecting a controller, laying out the circuitry for the I/O signal conditioning and connectivity, writing the application using the tools available for that controller, manufacturing the printed circuit board (PCB), debugging the PCB and app software, and building an enclosure if necessary.

Clearly, this approach takes significantly more effort than the COTS approach before the prototype is available for the intended purpose of evaluating your product idea or upgrade. In our experience, it is common for this effort to take 4 to even 10 times the COTS prototyping effort.

The Hybrid Approach

Your prototype could also be part COTS and part Custom. Such a "Hybrid" approach can offer advantages by combining the increased speed of prototyping with COTS with the reuse of the same hardware and software in both prototype and end-product. By carrying over the effort spent to design and build the prototype and reusing it in the end-product, any NRE is spent once.

The amount of NRE savings as compared to the COTS or Custom approach generally scales with the amount of reuse in software (for the application and drivers) and hardware (for the signal conditioning I/O and connectivity). The ideal situation utilizes the same controller for both prototype and product and couples that with I/O designs that can relatively easily be laid out into a custom I/O PCB that connects with the chosen controller.

Let's explore these three approaches in more detail by covering three important decision factors that you will face when creating a working prototype and subsequent end-product.

COTS
HYBRID
CUSTOM

CONSIDERATION #1: Time and Money

Cost is almost always the number one concern for companies when building a product: cost in time and cost in money. In terms of the bottom line, the smaller the costs accrued in prototyping and the faster the new revenue appears, the better the company cash flow position.

Let's look at some of the largest costs for creating the prototype and end-product.

Custom

Since the Custom path requires that the prototype be developed from scratch, the non-recurring engineering (NRE) effort is very significant right from the start.

This large NRE arises from a development flow similar to the following list.

- A hardware designer must first design the PCB schematic and then layout the board to specification, since the board is being made from scratch.

- After the layout is finished, it can take approximately two weeks to spin the PCB. For expedited delivery times, an additional fee must be paid.
- If a bug is found on the PCB, it will take additional design effort and another two weeks to fabricate the revised board.
- If there are mechanical issues, noise problems, backward connections, or any number of bugs, reworking the hardware can significantly delay the project timeline and considerably increase NRE design costs.
- The software development effort can start in parallel with the design and build of the prototype PCB. However, as all software developers can attest, the real work begins only when the hardware is available and debugging in real-world situations can occur.
- Additional tweaks to the PCB and software can occur during the application development, when the application doesn’t perform as expected.

COTS

COTS solutions tend to have a shorter prototyping phase, since the hardware is ready-to-use out of the box and troubleshooting involves only rewiring, reprogramming, or changing the types and quantities of COTS I/O modules. Hence, most of the effort is in software development, and the NRE in prototyping will be smaller than for the Custom approach.

However, you may need to consider converting your COTS prototype into a Custom design for various reasons such as size, power, unit cost, and maintainability, reasons that are likely driven by market desires that force you to remake your prototype into something that your potential customers will buy.

If you do have to rework your COTS prototype, then the NRE will grow, possibly even larger than the NRE in the Custom approach.

The best argument for staying with COTS for your end-product is when either:

1) You are not sure your product idea is viable (you can “fail fast” with COTS)
2) You don’t think you will be selling many units (so “low” unit cost is not the main driver of sales) and form factor is not an important sales attribute.

If neither of these reasons apply, the COTS approach will very likely end up converted to a Custom approach and hence as expensive as the Custom route.

Hybrid

Since a Hybrid approach tries to balance the benefits of the COTS and Custom approaches, the NRE can beabout the same for prototyping as the COTS approach. Further, you likely will start with close to the same form, fit, and function as the prototype when it’s time to move to production, leaving total NRE lower than either COTS or Custom.

Specifically, this reuse benefit appears when you select the same COTS embedded controller that would be used in your end-product and coupling it with COTS I/O. This hybrid combination brings you closer to the form factor your end-product needs than would COTS alone. However, note that, if you need some I/O that is not available as COTS, then some Custom I/O will need to be developed and the NRE will increase but only proportionally to the amount of COTS compared to Custom I/O you require.

Approach	Pros	Cons
Custom	Develop for the intended target right from the start. Best if you have a solid idea of your goals, such as an upgrade for an older (possibly obsolete) controller.	Highest initial and total NRE investment due to design and build from scratch. And, if your idea fails, you unfortunately have spent extensive time and money.
COTS	Lowest initial NRE. Complete the prototype stage as quickly as possible. Best if you need to do some R&D in order to validate your idea or will have "low" sales volume. Also good if this form factor works for your end-product.	If the COTS hardware needs to be reworked on a different platform to build the end-product, you will spend significant NRE on the end-product design and development.
Hybrid	"Low" to "medium" initial and total NRE. Fastest development. COTS controller enables the software to be developed on the same controller used in the end-product. I/O modules of acceptable form factor or access to I/O designs allow the end-product to be designed quickly.	If the prototype I/O form factor is not suitable, you will either need to design custom I/O or rework the prototyping vendor I/O boards. Look for vendors that have open or licensed schematics so the I/O layout can be moved to alternate forms.

CONSIDERATION #2: Performance and I/O

Your prototype and end-product need to be capable of reacting to inputs, processing data, crunching algorithms, and updating outputs as quickly as needed for your product's application. Achieving this throughput requires both a controller powerful enough to handle all the tasks and I/O with enough bandwidth, precision, accuracy, and damage protection from the outside world.

The controller market is very mature and offers a broad range of products, from tiny, low-power, very simple microcontrollers to high-speed, fully-featured 32-bit computers. The on-chip peripherals can include:

- Memory (EEPROM, RAM, and Flash)
- Communications (CAN, Ethernet, RF, USB, I^{2}C, SPI, and UART)
- Timers/counters
- PWM channels
- Encoder interfaces
- A/D convertors
- Graphical and touchscreen interfaces
- FPGA

Besides the controller, analog and digital I/O capabilities will be needed. Not only does the controller need to support the appropriate channel count and speeds, you will need signal conditioning between the controller chip and external sensors and actuators.

Also, you may wish to consider an FPGA to offload time-critical acquisition and processing from the controller. A Field Programmable Gate Array (FPGA) is a special integrated circuit consisting of logic blocks with interconnects that can be programmatically "rewired" via VHDL (VHSIC Hardware

Description Language) to create different complex applications. Moreover, FPGAs can execute algorithms in parallel with no task switching.

Often, we find that domain experts, while expert in their field, have little experience with embedded technology, and they can be overwhelmed with so many options when trying to choose an embedded controller and I/O for their product concept. Let's review some of these options.

Custom

Embedded controller companies offer many reference designs that can be leveraged for use in your prototyping circuitry. Many of these reference designs use support ICs made by the same company that makes the controller, so you can gain the benefit of additional components being from a one-stop-shop.

To assist, many companies offer pre-assembled development kits (also called eval boards) that have the controller and a number of features already implemented to help streamline prototyping for certain types of applications. Keep in mind that these starter boards are created as demo versions of the manufacturers' product, and will often still require heavy customization to become prototype-ready.

For software development, controller companies often provide in-house C and assembler compilers, along with many third-party vendor toolsets. Any FPGA component is usually programmed in VHDL. The application software typically runs on either Linux or some real-time operating system (RTOS). Several RTOS choices are available, including real-time versions of Linux.

For the I/O in a custom prototype, the channel granularity matches the requirements precisely, and you pay only for the necessary features, thus minimizing price per unit of your product. Furthermore, when the system requires support for special sensors and signals, signal conditioning, or signal connections that are not readily or inexpensively available as COTS, or may require many intermediate COTS products, then building a custom solution is more reasonable than buying a COTS solution. It will likely be more straightforward to build the prototype from ground up so that the specifications – such as noise performance, voltage levels, or dynamic range – are not compromised by the availability of COTS products on the market and instead are exactly tailored and optimized to the product's needs.

Additional Information: As an example of the advantages of utilizing a one-stop-shop, Microchip (www.microchip.com/) has a very broad set of pre-coded libraries that simplifies the use of their on-chip peripherals, such as software stacks for TCP/IP and Bluetooth, as well as algorithms for DSP-based products, graphics libraries for displays, and much more. This example serves as motivation for you to explore options.

COTS

With the COTS approach, you will want to consider a COTS embedded controller that is also ready to accept I/O modules without the need for any customization. Also called Embedded CPUs or Embedded PCs, these controllers will be plug-and-play compatible with COTS I/O modules. Some examples of such systems are the National Instruments CompactRIO and systems based on single board computers utilizing PCI plug-ins for the I/O, offered by such vendors as ADLINK, Advantech, Technologic Systems, and many others. These systems often run a version of Linux for the operating system and some are real-time.

With all these COTS components readily available for purchase, the COTS solution is generally more adaptable to change and the COTS solution can usually be easily modified during multiple prototype revisions.

For instance, the prototype may need just one current input channel, but the COTS supplier only sells modules with four current inputs each. The downside is that three extra channels were purchased and may never be used, taking up both space and budget. The upside is that if the prototype does need more than one current input, the additional channels or modules are immediately available for use and little time is lost. Also, the extra I/O can come in handy for monitoring or troubleshooting an issue on the prototype.

As another example, if the prototype unexpectedly needs an analog output, the COTS supplier likely already has a module for sale. Again, the downside is that additional budget and space have been consumed, but the upside is that a solution was readily and rather inexpensively available.

However, COTS analog or digital I/O boards may not offer adequate connection schemes because of signal speed, channel-to-channel isolation, signal conditioning, or connector requirements. In those cases, custom circuitry and hardware may need to be built in order to add special functionality to COTS systems, which can increase time and cost.

Additional Information: Some examples of COTS systems are the National Instruments CompactRIO (www.ni.com) and systems based on single board computers utilizing PCI plug-ins for the I/O, offered by such vendors as ADLINK (www.adlinktech.com), Advantech (www.advantech.com), Technologic Systems (www.embeddedarm.com), and many others. These examples should serve as motivation for you to explore options.

Hybrid

The hybrid approach can utilize both a COTS embedded controller and pre-developed COTS I/O. Connecting this COTS hardware into a prototype system is certainly simpler than the Custom approach and is only slightly harder than offered by the COTS approach. The benefit of the Hybrid approach stems from the ability to develop the prototype close to the same timeline as the COTS prototype and minimal post-prototype customization since so much of the COTS hardware can be reused.

It is important to note that ease of rework to convert the prototype into a ready-to-sell product implies either that 1) the COTS I/O is directly suitable for your product or 2) you have access to the COTS I/O hardware designs so that you can (or the supplier of the COTS I/O will) re-layout only the necessary I/O components into your own specialized I/O board.

Additional Information: Examples of vendors that offer such a hybrid approach are those that are built on the PC104 standard, such as WinSystems (www.winsystems.com) and RTD Embedded Technologies (www.rtd.com). And see www.pc104.com for a longer list of PC104 suppliers. Also look at systems based on the National Instruments RIO standard, such as the VERDI platform from Viewpoint Systems (info.viewpointusa.com/verdi).

Approach	Pros	Cons
Custom	Optimally chosen components deliver only the performance and I/O that you need.	All the components have to be made from scratch and to work properly together.
COTS	Wide choice of options.	The chosen configuration likely will have more performance capability and I/O than you actually need.
Hybrid	Balance between smaller choice of performance options than either Custom or COTS and optimal I/O selections.	If the I/O available for prototyping is insufficient or overly capable, an NRE effort is needed, but the outcome is an optimal selection of just the I/O you need.

CONSIDERATION 3: Product Life Cycle

Selling a product is the ultimate goal. Consequently, consideration must be given to the number of units you expect to sell and the timeframe in which they will sell: do you have a large backlog of customers anxiously waiting to purchase your product or will the product sales ramp over several years? We want to understand the impact of NRE and future product updates on the unit costs of the product over the several years it will be sold, simply because you will want to recoup your NRE investment and quickly.

Let’s explore the COTS versus Custom options to understand the impact of sales on the production life cycle. For the purposes of scale, let’s say that “high” volume sales mean anything larger than 50 units per year. In our experience, the crossover of unit cost between the various COTS, Custom, and Hybrid approaches is often between 10 and 100 units (total, not per year).

Custom

If you know or suspect that you will have “high” volume sales, then starting with a Custom approach is going to give you the lowest unit cost as quickly as possible. The NRE will be highest for this approach, due to all the custom development work and any possible certifications you might need (such as a CE Mark), but the Cost Of Goods Sold (COGS) will be as small as possible. Spreading the NRE costs across each unit sold for this “high” volume situation allows you to keep unit pricing low because you’ll recover your NRE “quickly”.

However, if you do not expect to have “high volume” sales, then this Custom approach is going to make the unit price high for a “long” time. Unless you have a strong reason to follow this approach, such as size or power constraints that your customers demand, use one of the other approaches.

An important benefit of the Custom approach is that longer term modifications, warranty support, and parts life cycle management will be in your direct control, since you have access to all the designs and parts lists in your product build.

COTS

The COTS approach is best suited to “low” volume sales, especially if the customer places minimal constraints on the product design, such as size and power. For example, if your product already has a large enclosure for motor controllers and power supplies, an embedded PC or embedded controller can be very effective. Furthermore, if controller and I/O are a small fraction of the cost for all the remaining components in your product, then unit costs for the embedded system may be less important than reducing costs elsewhere, and you may be able to tolerate using COTS, even for “high” volume sales.

Furthermore, if you are using these initial unit sales as a way to build understanding of your customers through the early adopters of your product, then the COTS approach is especially beneficial, since you can make an updated version quickly compared to the other approaches discussed in this paper.

The life cycle management of a COTS approach offers the advantage of pushing the management of the parts life cycle onto the supplier of the COTS components. Warranty handling can be simplified when using COTS, since the supplier takes much of the risk in product failures. Worldwide support networks often exist for the larger COTS suppliers, and your customer halfway around the world can get a replacement part delivered and possibly installed by the supplier instead of needing to ship parts around the world and talking (or traveling) to the customer for the replacement procedure.

Hybrid
The Hybrid approach combines the benefits of a “low” unit COGS with a “medium” NRE to enable a rapid time to market with reasonably “low” unit costs. This benefit appears because the COTS controller is merged with some combination of COTS and Custom I/O. Thus, this approach can be effective for both “low” and “high” volume sales.

Longer term modifications, warranty support, and parts life cycle management are partly in your control and partly controlled by the vendor of your COTS components. In our experience, your custom parts are often tailored to your product in ways that would make a COTS approach too expensive or even not available. Often, these custom tailored parts are a differentiator for your product (easier field connections, higher signal to noise, etc.), and you would prefer to keep life cycle control over these custom parts, rather than turn over that life cycle management to a COTS supplier. Thus, the Hybrid approach places life cycle management of specific components with the most appropriate people.

Approach	Pros	Cons
Custom	Best for highest volume sales or when needing lowest unit costs. Offers most control over changes to product components, since you are in control of more aspects than any other approach.	You have to manage the entire life cycle yourself from warranty returns to end-of-life components.
COTS	Highest leverage of vendors to help manage the product life cycle, since most of the product is COTS.	Least control over component obsolescence. Part can go obsolete too late for you to react.
Hybrid	Balance between vendor and your management of component life cycle. Since I/O is often your design, you keep control over your IP and leave controller life cycle management to a COTS vendor.	The COTS controller vendor may obsolete the controller with little time for you to react.

Summary

The choice between using Custom components for prototyping and product build versus COTS is balanced with a Hybrid approach that utilizes a COTS controller with as much COTS I/O as possible but in the same form factor used in your end-product. The initial NRE for Custom is highest, but COTS may end up having almost as high an NRE too if you have to redesign COTS into a custom solution after prototyping for reasons of unusable form factor, excessive unit cost, or concerns about lack of control in the product life cycle. The Hybrid approach balances many of the NRE costs and life cycle issues by letting you focus on your IP in the form of software and I/O while letting a COTS vendor manage the controller and its standard peripherals.