# Building your next killer embedded UI


The no-nonsense guide to creating high volume, high margin consumer goods with sophisticated interfaces.

CLICK TO START

CRANK®
AMETEK®


## INTRODUCTION

The face of consumer goods is quite literally changing. We've come a long way since the VCR, with growing numbers of household appliances boasting easy-to-use interfaces. This is due in no small part to consumers—now accustomed to graphically rich and intuitive interfaces on their phones—no longer satisfied with dumb appliances that blink 12:00. The challenge comes when manufacturers try to meet consumer expectations with slick-looking GUIs on inexpensive hardware to maintain good margins.

This e-book looks at this next wave in the consumer market, the challenges that come from creating products that are at once sexy and cheap, and ways that you can turn your low-end products into ROI superstars.

# What to expect from this e-book

### Four critical consumer expectations that open wallets

Create successful products by taking these consumer preferences into account.

PART 1

### Two ways to meet the relentless pressure to deliver faster

Understand the importance of streamlining the development process and ways to do it.

PART 2

### Selecting silicon and software to differentiate your product

Capitalize on consumer preferences and overcome time-to-market pressures through component selection.

PART 3

### Two sure-fire ways to break into the product sweet spot

Adapt existing designs to create products with mass-market appeal and high-margin pricing.

PART 4

### How to determine if your product is a good candidate

Decide for yourself if these strategies make sense for your product.

PART 5

### Key takeaway

Do you have what it takes to create a consumer-electronics masterpiece?

PART 6

PART 1

# Four critical consumer expectations that open wallets

UX leaders Apple and Google have forever changed consumer expectations. Products must now be simple and user-friendly regardless of device or platform. While these two companies have loyal followers, in general people are beginning to care less about the brand and more about the experience. Data from Forrester shows that improving customer experience improves profitability; in fact, the revenue growth of customer experience leaders is 5.1 times that of laggards. A case in point is Uber: the company has been through one scandal after another yet they remain successful because people love the overall experience. Let's examine four of today's biggest consumer expectations and develop some guidelines to follow in order to make your next consumer electronics product wildly successful.

## 1 Create products with both functionality and design in mind

In the good old days when function trumped form, consumers had to live with difficult-to-use products like the much-maligned VHS recorder. The reason TiVo killed the VHS was it was well-designed and dead simple to use. Today, consumer preference for a product is increasingly determined by the quality of the user experience. Designers and developers who work closely together to create products that are at once attractive, functional, and easy to use are behind some of the world's most successful products.


Apple was not the first MP3 player to hit the market but, with its sleek design and intuitive interface, it quickly dominated all other rivals.

In order to strike that perfect balance between form and function, you need to have designers engaged from the very beginning of the product development process. Since critical design choices may significantly morph your product, you shouldn't wait until the product requirements are done before you start working with the design team or you risk wasting time or building suboptimal products. We also recommend facilitating a smooth and efficient workflow between designers and development teams with both people-focused activities like team building and technological solutions like designer-friendly tools.

*Quite simply, design can no longer be an afterthought.*

## 2 Include UI screens that are attractive and intuitive

Now that consumers have continual access to an amazing UI in their pockets, they're less tolerant of lukewarm experiences. They may not always be able to articulate their preferences but know how to vote with their wallets.

A good example is the growing popularity of the single-serve coffee machine. Price comparison shows a unit of K-cups (a single-serve of coffee grounds in a small filter) commands nearly five times more than a unit of traditional ground coffee.

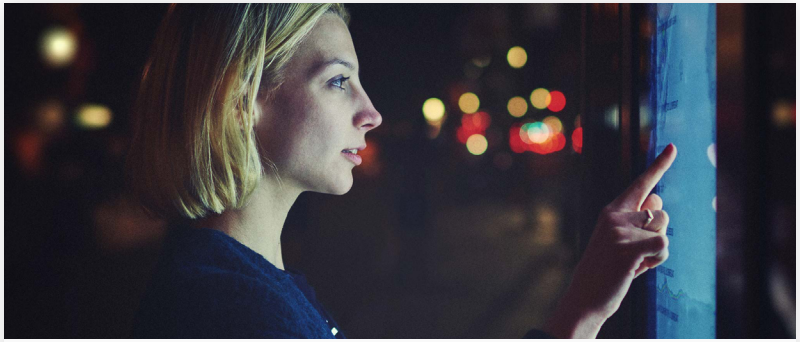
One of the reasons these coffee machines rank in the fastest-growing private-label food and beverage category despite the hefty price tag is their full-color LCD. Consumers see these displays as a valuable feature and are increasingly making purchasing decisions based on their availability.

Millennials in particular are very comfortable and familiar with screens (and are now the largest generational group in the US); they are in no small part responsible for creating a demand for screens that trickles down into all manner of embedded devices (Forbes, 2015).


Consumers see full-color displays as a valuable feature and are increasingly making purchasing decisions based on their availability.

Further proof of this screen envy is the growing hardware market for color LCD, OLED, and micro-LED displays, predicted to grow by $30 billion USD over the next five years (Marketwatch, 2019). Of these, the fastest growing category is the microdisplay—a tiny electronic display system usually less than two inches diagonal—that is the enabler for consumer goods discussed here.

Microdisplays leverage an iPhone glow—a product with a full color and attractive display feels like a smartphone, lending it a certain amount of sophistication and sense of capability. However, a screen that's difficult to use creates a negative effect. We recommend adding a screen but only if it's a good one.



Consumers love the tactile experience of interacting with sophisticated touch-screen displays.

### Reasons consumers are attracted to products with color LCDs

- **Modernity**: full-color dot-addressable screens are fresh, while custom-designed LCDs are reminiscent of old technology like clock radios
- **Capacity**: the flexibility of a product with an intuitive full-color screen makes the product appear to do more than its fixed-function LCD counterpart
- **Simplicity**: features can be revealed as needed (including usage guides or servicing instructions) while products that have many buttons or LEDs can confuse a user by presenting all options at once
- **Elegance**: end users have increasingly sophisticated design ascetics, making them prefer products with appealing, attractive, and artful screens

## 3 Plan for multi-modal input

Of course, there are other ways to interact with a product than through a screen. Digital assistants are increasingly integrated into devices beyond the smartphone, enabling our homes, cars, and appliances with artificially intelligent capabilities. Sometimes.

While digital assistants like Siri and Alexa are getting a huge amount of visibility and traction, the hard fact is that they aren't yet a fully baked solution. Voice-enabled helpers still face integration challenges, closed ecosystems, and limited competence that make them more technology hype than fundamental feature. That said, the digital assistant market isn't standing still. As voice agents increase in capability, quality, and usability, they'll no doubt become an indispensable and seamless part of our lives.

Another multi-modal input that cannot be ignored is the user's device of choice. Whether that means building your product to connect to a smartphone, laptop, connected home, or car, this type of flexibility enables greater integration into a consumer's lifestyle and creates a happier and more loyal customer.

**Use a tool that can bridge to C/C++ to maximize your ability to expand in the future.**

We recommend designing with multi-modal input in mind—even if your product doesn't immediately require voice or device connectivity, it should be able to accommodate it. This means designing your product for expansion, exposing APIs for remote functionality, and planning to provide over-the-air updates.

## 4 Plan for scalability

If the best products provide an excellent user experience, the best selling products also provide excellent value for the money. With so many forces attempting to get a share of the consumer wallet, you need to build products that are "worth it", and that usually means a bill of materials that's as low as you can manage.

While it may be accepted practice to create an initial line of products with high prices (and high margins) to test out market acceptability and build up cash reserves, this is really a short term strategy. Tesla is a perfect example of this approach: the company started with a pricey Model S before releasing the Model 3 for middle-class buyers. Companies that don't make the transition to commodity pricing are often stuck creating a niche product for a limited market instead of breaking into the mass market.

**Use a tool set that works with microprocessors and microcontrollers to target products at all price points.**

Build your niche-market product with scalability in mind so you can move from a high-end microprocessor to a lower-priced microcontroller with ease.

The solution is to plan for scalability. From the electronics point-of-view, you should choose a microprocessor line for your initial product that's part of a product family with lower price-point cousins. Use a tool set that isn't inherently tied to beefier processors and that can scale into smaller—yet still capable—microcontrollers. Finally, plan your RAM and Flash consumption carefully so that you're not trapped into an architecture that is resource heavy and can't be scaled down. These considerations will help ensure you're building a product that could run on a lightweight microcontroller even if your initial design hasn't yet taken the necessary steps.

PART 2

# Two ways to meet the relentless pressure to deliver faster

Everyone knows that the innovation treadmill is leading to ever-shortening development cycles and increased competition. The big question is: how can manufacturers compete under that relentless pressure to deliver faster and faster? Just as critical is maintaining quality at this break-neck speed.

Since the software development timeline is the single biggest factor that inhibits speed to market, let's take a closer look at ways to reduce it.

## 1 Streamline design and development

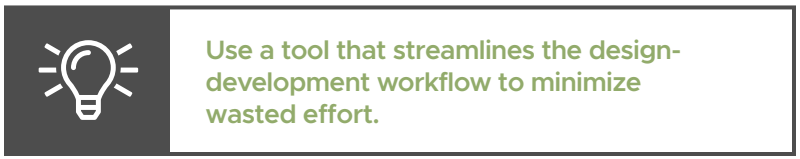
As discussed, design is a key part of today's product success. But the normal design-developer cycle is often inefficient.

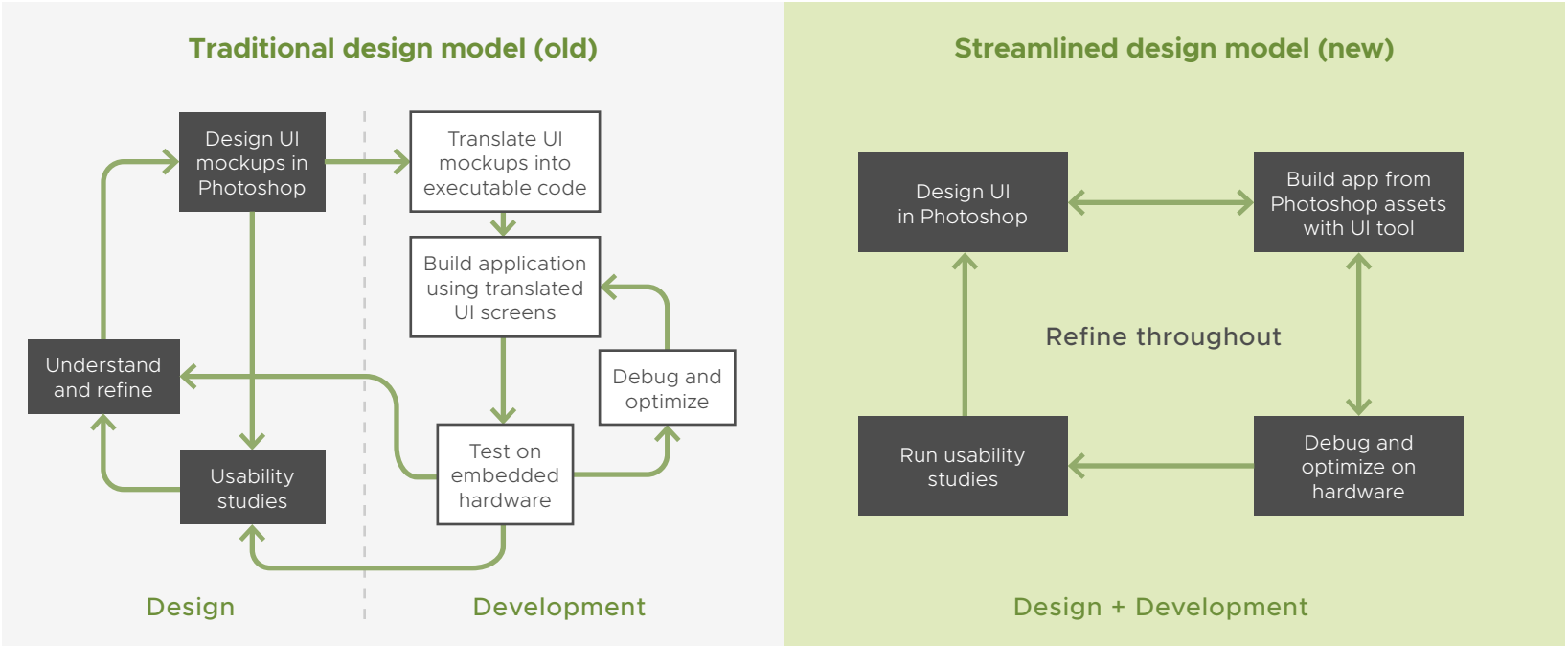
Designers often use one set of tools (such as Adobe Photoshop) to craft the look and feel of a UI and another set for prototyping (like Adobe Experience Design). Developers then use a third toolset (often C/C++) to implement the final product. All of these changes in tools, process, and workflow suck up time and introduce errors. Thankfully, there are tools on the market that can help create a streamlined design-development process.

The best development tools are ones that are simple enough for the designer to create a prototype UI but deep enough for the developer to produce complex logic, include pre-existing libraries, and modularize components.

As always, your mileage may vary—depending on the skills and breadth of your team—but keep in mind that your team's current skill level in various tools often changes when principle members move on or you need to staff up.

If your team doesn't already use a tool to streamline the design-development process, you'll want to seriously consider one.






## 2 Architect code for reuse

If you're building a one-off product, code reuse doesn't matter. That being said, you almost never use good code only once. Initial releases invariably get updated with new features and bug fixes, one product can lead to a family of products, and reliable code from older projects is often implemented into new incarnations. So if you're writing code with a plan to throw it away, you're doing it wrong.

A rough guideline is that writing reusable code takes at least three times longer than writing single-purpose code. Not everything however needs to be built for reuse or you may end up over-engineering your code and adding unneeded complexity. If code is written too generically but only used once, it's a waste of development time and code overhead.

**Ensure your UI tool can cleanly separate UI logic from business logic in order to leverage visual elements and themes across projects.**

But somewhat paradoxically, code reuse is very important to rapidly producing reliable products, even with the extra time required to make code reusable. You can always get one product to market quickly. With reusable code you can get subsequent products to market even quicker.

Experienced developers constantly make subconscious trade-offs about the architecture of the software they're writing—which parts are likely reusable and deserve extra attention. It pays to discuss, and incorporate reuse guidelines into your team workflow.

Finally, use a methodology that separates the UI logic from the business logic. Doing this will allow common visual elements and themes to be leveraged across products and platforms, making it easier for you to build derivative products. Ensure your UI tool of choice makes this simple. Your development staff shouldn't constantly fight against the tool to implement a clean MVC (model view controller) paradigm.

Allowing designers and developers to use the same tool eliminates disconnects and rework.

### Top things to consider when choosing a design-friendly UI tool

- Scripting for simple understanding by designers and speed of creation by developers
- Ability to invoke C/C++ code and libraries from within the UI environment
- Flat tool hierarchy with easy learning curve
- Easy integration with existing design tools
- Ability to manage round-tripping design assets

PART 3

# Selecting silicon and software to differentiate your product

If your component evaluation process is like the majority, it can be overwhelming with all the comparative options, and rarely is there a single right choice. Rather than building a comparative chart, you should be laser focused on the factors that will allow you to capitalize on consumer preferences and overcome competitive pressures. Let's start with a look at hardware.

## Hardware

While full-featured microprocessors have traditionally been the solution for complex designs, microcontrollers have been steadily gaining power and capability and have spawned a new category of crossover microcontrollers (see sidebar).

These crossovers are excellent choices for many consumer electronics products, because they provide a high level of sophistication at a much lower price point.

In fact, they sport many of the same features. Yet they have few of the frills that are standard fare on application processors like memory management units, multiple cores, on-board GPUs, floating point units, and so on. But make no mistake, they can still pack a lot of power into a small chip—enough to handle all but the most demanding designs.

### Crossover Microcontrollers

Like a full microprocessor, crossover microcontrollers use a standardized instruction set, include a host of onboard interfaces, and provide enough power for running complex applications written in high-level languages. Like a traditional microcontroller, they are designed for smaller form factor designs, normally run applications on bare metal without an OS, and include onboard RAM and Flash for a single chip solution. The best of both worlds.

The really good news is the price of a crossover micro-controller is usually significantly less than a low-end application processor. With RAM and Flash on board the microcontroller, crossovers are often able to remove the cost of additional chips from a design. Depending on which RAM and Flash chips are eliminated, a crossover micro can reduce cost from $4 to $40 (sometimes more), which is significant even on the lower end.

This substantial price advantage is why we're looking at crossover microcontrollers in this e-book.

## How do I pick the right hardware?

We've taken the consumer and competitive factors, translated them into the most important hardware features, and created the chart at right to help you decide which hardware is best for your project. This list isn't exhaustive but should get you thinking. For instance: if you want your product to appeal to millennials and intuitively communicate the functionality it houses, your hardware better have fantastic graphics support.

If you want to create a product with mass-market appeal, keep on the lookout for chips that minimize your BOM cost—such as ways to eliminate external SDRAM.

We suggest using this chart to review the features of each microcontroller. We've started you off by doing this for a few of today's popular ones in the table on the following page.

**Prioritizing hardware selection criteria based on product requirements**

| To support these features... | Look at these criteria... |
|---|---|
| Functionality and design | Graphics support |
| Attractive screens and UI | Graphics support |
| | CPU horsepower |
| Multi-modal inputs | CPU horsepower |
| | Wide array of interfaces |
| Excellent price point | BOM cost |
| | Size of on-board memory |
| Streamlined design and development | Standardized instruction set |
| Reusable software | Standardized instruction set |
| | Scalable product family |

## Microcontroller comparison

| | Processor attribute[1] | NXP i.MX RT1064 | ST Microelectronics STM32H753 | Microchip PIC32MZ |
|---|---|---|---|---|
| **Graphics support** | Acceleration | 2D acceleration engine, pixel pipeline | 2D bitblt engine | n/a[3] |
| | Screen | 1366 x 768 WXGA | 1024x768 XGA, HDMI | Available through external bus interface (EBI) |
| **CPU horsepower** | CPU frequency | 600MHz | 400MHz | 200MHz |
| | CoreMark | 3036 | 2020 | 710 |
| **Wide array of interfaces** | Peripherals | WiFi, BT, BTLE, ZigBee, Touch screen, Ethernet, USB, UART, CAN, I²C | USB, Ethernet, I²C, UART, SPI, CAN | USB, Ethernet, I²C, UART, SPI, PMP, SQI, EBI |
| | Audio | 3 I²S, 2 A/D | 3 I²S, Serial audio | I²S, A/D |
| **Size of onboard memory** | RAM | 1MB SRAM | 1MB SRAM | 128KB |
| | Flash | 4MB | 2MB | 512KB |
| **Standardized instruction set** | Core architecture | ARM Cortex M7 | ARM Cortex M7 | 32-bit MIPS microAptiv |
| **Scalable product family** | Substitutable chips | Scales up to i.MX6/7/8, scales down to RT1050 | Scales up to STM32MP1, scales down to STM32H750 | Scales up to SAM A5, Scales down to PIC32MM GPL |

[1] Processors being compared are not intended to be an exhaustive list of all compatible options
[2] Pricing estimates are from the manufacturer websites but must be confirmed with suppliers
[3] Graphics rendering is through software only

Compare hardware

## Software

What about UI  tools, the other critical ingredient for building a beautifully designed product? Many companies still build their UI code in-house. With all the excellent options out there that are extremely capable, already written, and pre-debugged, this is a complete waste of energy. You're not going to differentiate your product based on custom UI tools—you're going to waste a lot of time reinventing the wheel. Save that energy for building things that will make your product stand out and leverage someone else's hard work.

That's not to say that UI choice is any easy one; rather it's a fundamental one. Your UI tool can make a development job easy or hard, shape the look and feel of your product's screen, determine how easily designers can be involved, dictate minimal hardware requirements, set the development language and toolchain, and either constrain or enable the inclusion of additional features.

## How do I pick the right software?

If multi-modal inputs are critical, you're probably going to need to be calling code in C libraries. What if getting to market in record speed is at the top of your list? Make sure your tool breaks down the barriers between designers and developers.

Use the chart at right to guide you when evaluating software tools. We've started the process for a selection of UI tooling products as shown in the table on the following page.

**Prioritizing software tool criteria based on product requirements**

| To support these features... | Look at these criteria... |
|---|---|
| Functionality and design | Designer-friendly |
| Attractive screens and UI | Designer-friendly |
| | CPU horsepower |
| Multi-modal inputs | External code bridge |
| | HTML support |
| Excellent price point | Microcontroller support |
| | Runtime size |
| Streamlined design and development | Designer-friendly |
| | Easy learning curve |
| | Powerful scripting |
| Reusable software | Easy learning curve |
| | Powerful scripting |

## UI tool comparison

| | UI tool attribute | Crank Storyboard | The Qt Company Qt | TARA Systems Embedded Wizard | Altia Deep Screen |
|---|---|---|---|---|---|
| **Designer friendly** | Designer-based workflow | Yes | No | No | No |
| | Custom animations | Yes | Yes[3] | No | Yes |
| | 3D support | Yes | No | No | No |
| | Ability to round-trip design assets | Yes | No | No | No |
| | Uses Photoshop assets | Yes (directly) | Yes | No | Yes (import with PhotoProto) |
| **External code bridge** | Able to call C/C++ | Yes | Yes | No | Yes |
| **HTML support** | Embedded browser | Yes[2] | Yes[2] | No | No |
| **Microcontroller support** | Compatible with microcontrollers and microprocessors | Yes | No[1] | Yes | No |
| **Runtime size** | RAM required by framework | 120 KB | 256 MB | 80 KB | 25 MB |
| **Easy learning curve** | Able to be effective without training | Yes | No | Yes | Yes |
| **Powerful scripting** | Scripting environment | Yes (LUA) | Yes (QML—proprietary) | Yes (Chora—proprietary) | No |
| **Overall project applicability** | Vertical markets | All | All | All | Primarily automotive |

[1] MCU is not normally supported. May be possible with expert customization or through services contract
[2] Runtime memory estimates do not include HTML option
[3] Custom animations are not supported in the IDE but can be created by calling C++/QML APIs

Free Storyboard trial

PART

# Two sure-fire ways to break into the product sweet spot

If you're starting a product from scratch, you have the freedom of creating something that accommodates this advice from the start. But what if you've already got products that don't have attention-getting interfaces with mass-market prices? Thankfully, we've thought of that. We've got two ways for you to consider adapting your existing designs.

## 1 Downward migration

It often makes sense to introduce new features—especially screens—into high-end products first. With larger margins, premium products are better able to absorb the cost of extra hardware and early adopters are usually willing to spend a little more coin to get the latest tech. However, as these features catch on with the general public, it makes sense to migrate them into mass-market products.

Automotive instrument clusters are a prime example of this downward feature migration. Replacing physical gauges with digital screens started in premium level cars but has now trickled down to mid- and entry-level vehicles. Moving screens from high-end products into lower-end ones may be a successful strategy for you as well. If you've already built a great UI on your top-end device, bringing it into a mass-market product is a sure-fire way to differentiate it from the screen-less competition.

To succeed at this downward migration however, requires dramatically reducing costs and simplifying features. Thankfully the transition to a microcontroller provides enough cost savings to allow you to move premium features into mid-level products, even though that journey often requires simplification to meet the tighter constraints of the smaller


A premium code base can support a smaller screen with fewer pixels by using lower resolution images, excluding textures, and removing low-priority elements.

hardware. The process of extending your UI downwards is dramatically simplified if you plan in advance to build your UI for reuse and testability, and make UI reuse and testing part of your completion criteria.

The UI feature simplification required to reach lower-level screens has an added benefit. Besides smaller bill of materials costs, it also better segments your product family so that the newly enabled mid-market devices won't accidentally cannibalize your upper-end products.

## 2 Upward migration

The other direction for product migration is upwards, by adding screens to products that are already mass-market to help them attract more customers and increase their market penetration. The nearly ubiquitous smart device is a case in point. While not every consumer product has a smart variant, anybody who's been to CES or recently visited an appliance store can tell you which way the wind is blowing—screens are quickly becoming ubiquitous.

Adding a screen and smarts to an existing product lets you get more dollars and longevity out of an existing product line.

While screen-less products clearly get the job done, products with intuitive interfaces offer busy consumers additional convenience and efficiency. Take the white goods market as an example. Consumers can now set up laundry cycles so that they run in off-peak hours. They can also receive alerts from their fridge instead of wondering which condiments are running low. You need a screen to make features like this shine; it's nearly impossible to imagine these advancements implemented through old-fashioned dials and buzzers.

The public's openness to smart screens in all manner of devices is thanks (again) to smartphones and apps, the continual fast pace of technology breakthroughs, and the increasingly larger segment of society that embraces technological advances. An additional consideration in the white goods market is that governments and regulatory bodies introducing energy reduction incentives often coincidentally drive sales of the more energy-efficient smart appliances.

While these reasons for scaling your mass-market products upwards might be sufficient on their own, another major factor is the bottom line. By shifting focus from the lower-margin segment of the consumer electronics market towards the higher-margin and higher-growth segment, you stand to improve revenue and shareholder value.

Full color screens allow you to target the higher-growth and higher-margin segments of your market.

PART 5

# How to determine if your product is a good candidate

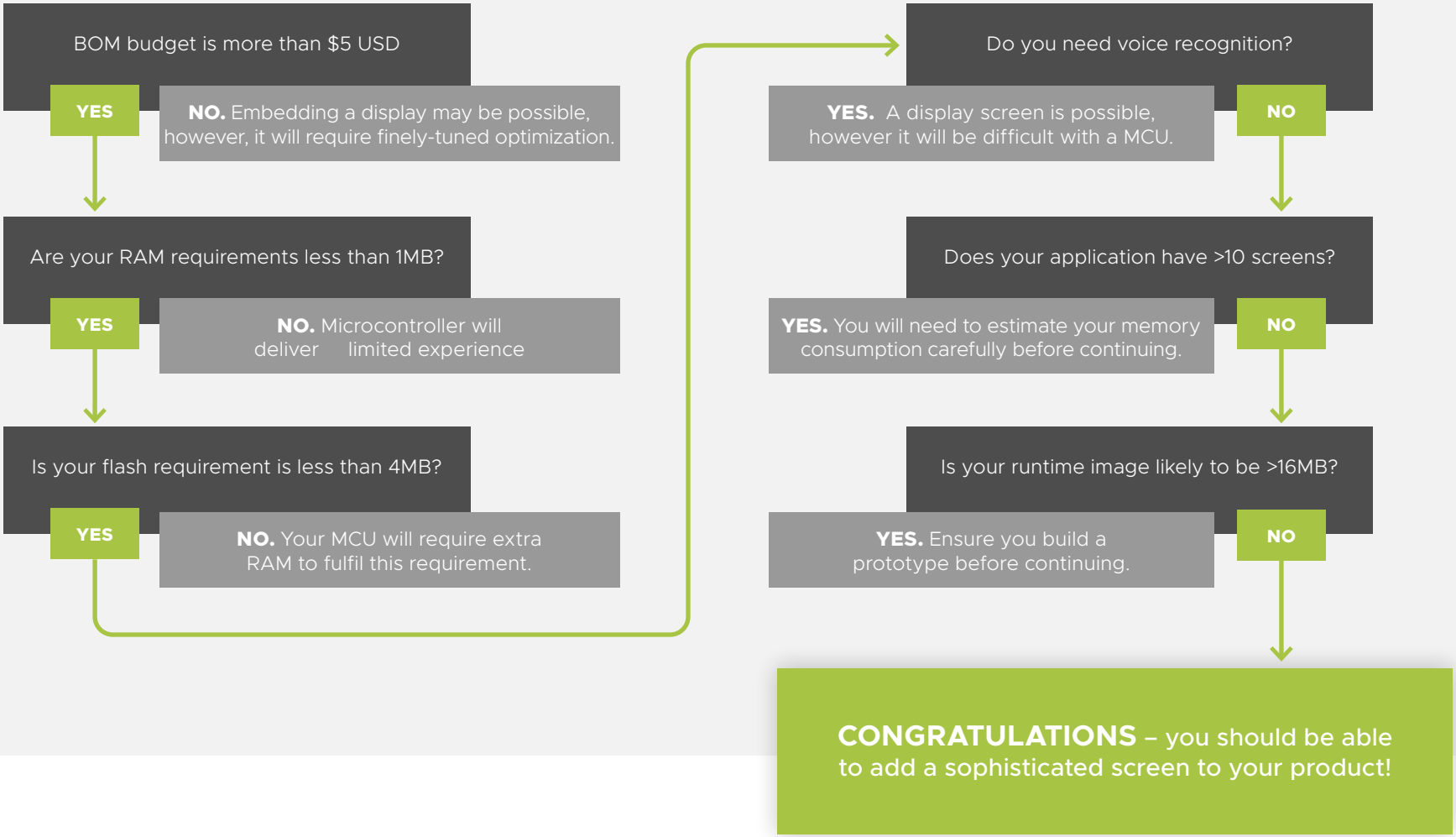
You're now convinced that adding a small screen and amazing UI to an existing product is a great idea. The next step is deciding if this approach makes sense for your product regardless of whether it's already on the shelves or just on the drawing board.

The simple checklist on the following page will help you quickly eliminate poor candidates for adding microcontroller-driven small screens to a product, potentially saving you some time for an evaluation that likely wouldn't pan out.

Even if you pass the checklist, you need to create a prototype to test out the concept. A recommended approach is to consult with your chosen UI framework company. They'll be able to give you their experience-based opinion if your product is feasible and if their platform is appropriate. Many also offer consulting services that can help you prototype, migrate, train, and develop your product.

Don't forget all parts of your software stack when calculating your RAM and Flash budget.

## Should you incorporate a display screen on your device?



PART 6

# Key takeaway

Digitally savvy consumers are fundamentally reshaping the B2C market. Manufacturers need to adapt to this digital disruption—and quickly—by responding to and exceeding consumer expectations, and by adapting to ever-shortening development cycles. Thankfully there is a way forward. By rapidly incorporating polished GUIs into products on inexpensive hardware, you can hit the product sweet spots while maintaining a good margin.

What's the bottom line? Embrace a future where screens are in everything. Be sure your products aren't just functional, they're also elegantly designed. Pick hardware and software suppliers that most easily enable you to create consumer electronics masterpieces. And lean on those providers for their expertise in helping you deliver these systems.


A supplier with experience in creating rich UIs on a wide range of silicon can help you diversify your portfolio.

Try Storyboard today

**"Storyboard made something possible that none of our competitors can come even close to reproducing."**

— Hank Bezuidenhout, Embedded IQ

**Crank Software**
1000 Innovation Drive, Kanata, ON, Canada K2K 3E7
+1 (613) 595-1999 | info@cranksoftware.com | cranksoftware.com



CRANK®
AMETEK®