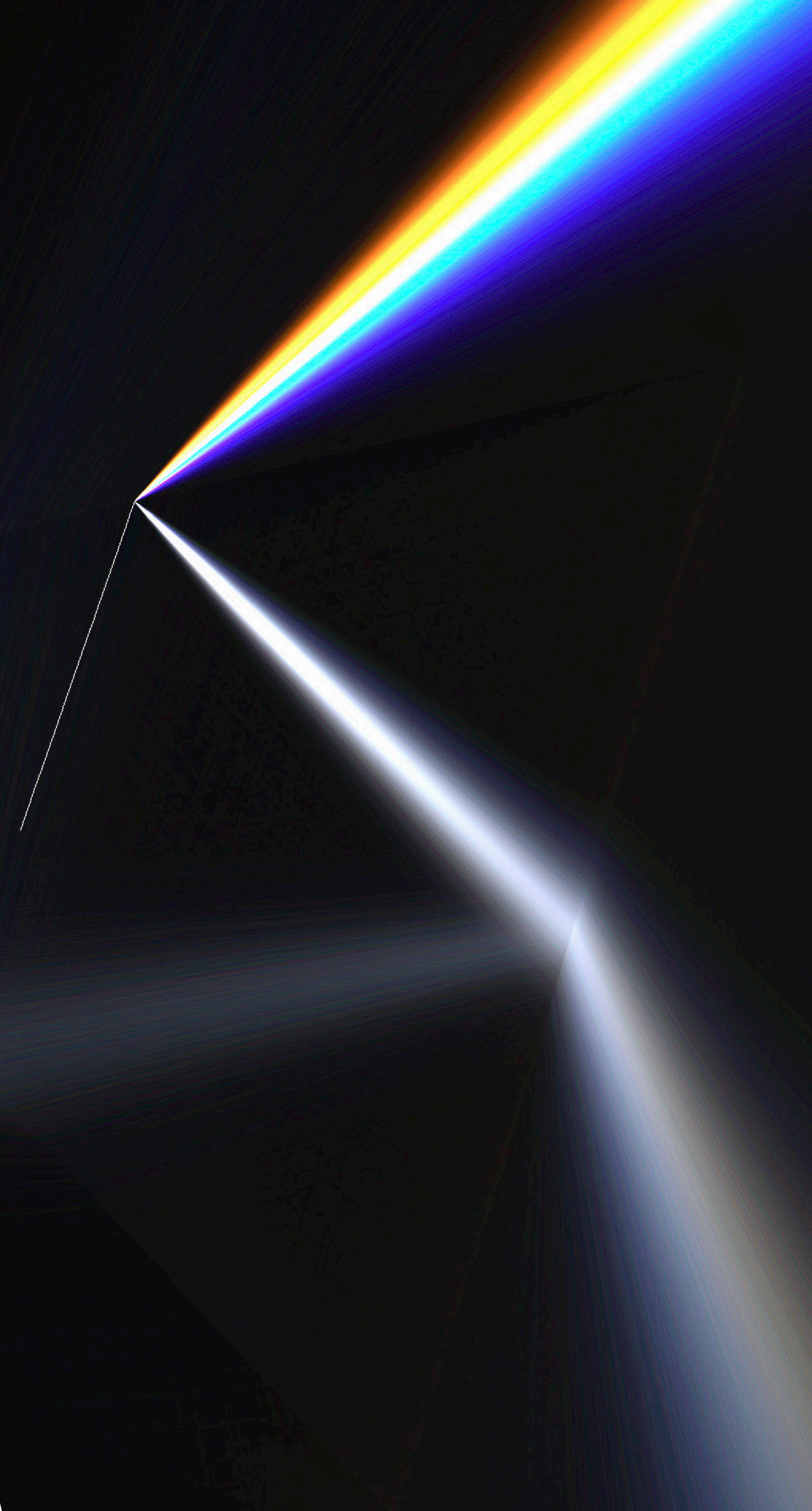
Volume 3, Issue 1

Methods

Mouser's technology & solutions eZine

DESIGN TRENDS AND THE TRANSFORMATION OF EVERYTHING

In this issue



Methods

Mouser's technology & solutions eZine

Foreword: Trends in the Transformation of Everything

Today is a phenomenally exciting time for both hardware and software engineers. It's a time when many different technologies are making it possible to connect, understand, and take informed action on what's happening in the physical world—all in real time. To understand the significance of these technology trends, I look at how they are affecting the business world.

Most of the big, emergent technologies are enabling companies to create digital feedback loops. A digital feedback loop is a means of connecting (in near real-time) products, the assets used to produce those products, employees, and customers. We call it a digital feedback loop because it's more than just a digital connection. It's a loop that collects insights that lead to further innovations that in turn lead to further insights leading to even further innovations. Thus, products and processes continuously improve. Digital feedback accelerates innovation in a way that would not be possible without the interplay of several emerging technologies. For example, in the case of edge computing and cloud computing working together, I can now deploy cloud functionality, like deep learning, or stream processing, such as serverless code-like functions, directly to IoT devices. Artificial intelligence (AI) only reaches its full potential when it's taking advantage of cloud and edge computing, and that depends on connectivity.

If you think about the way many businesses run today, especially those with long-established business processes, there are often gaps in feedback among these different aspects of the business. The result is that it takes a long time to compile sketchy information about how customers use a company's products, and the lack of details about facilities and manufacturing make optimizing business and manufacturing processes difficult.

Digital feedback loops provide real-time telemetry and feedback about the business processes and equipment used to produce products. This data shows how customers use those products. The feedback is actionable, which means that it can be used for both monitoring and control. By applying deep learning and AI, it becomes possible to optimize and automate control functions.

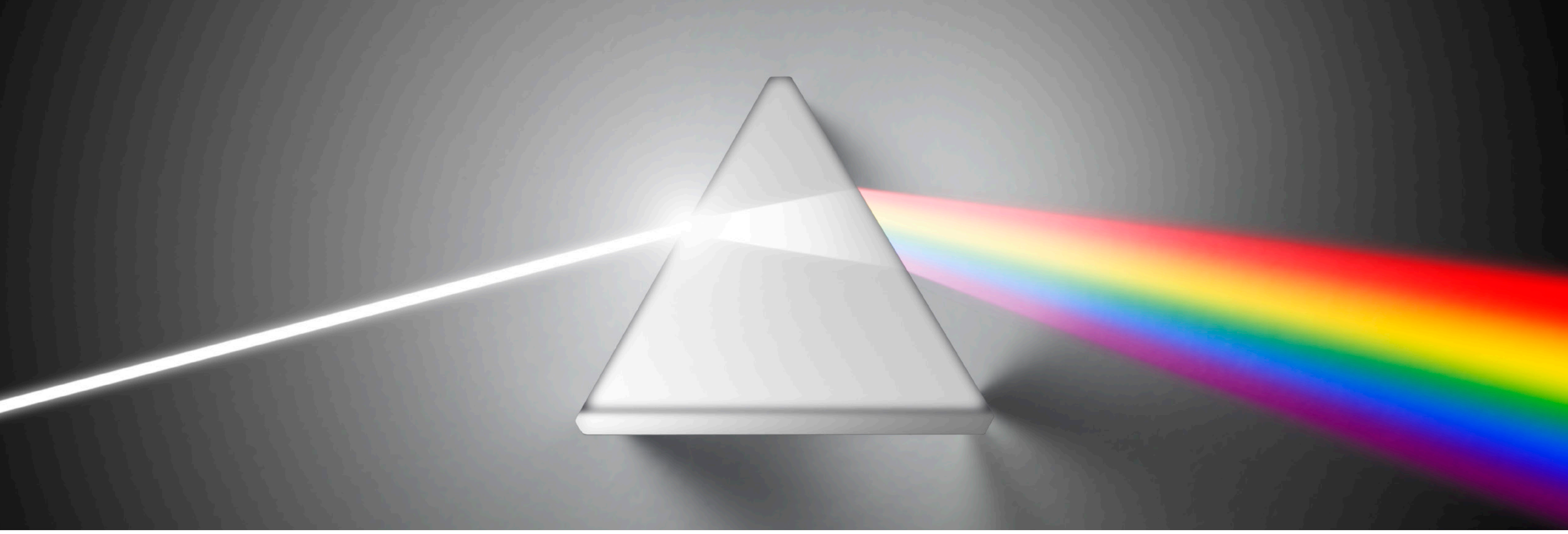
So the bigger frame of this transformational shift happening across businesses right now is

"I expect that 5G will unleash a massive amount of connectivity that will power digital feedback loops."

the connection of all the disparate, disconnected aspects of a business through the digital feedback loop. Viewed through this lens, many emerging technologies, such as the Internet of Things (IoT), augmented reality, 5G, autonomous vehicles, embedded systems, and others, make more sense.

For one thing, it's clear that several parallel technology developments are making digital feedback loops possible. For example, an IoT solution supported by connectivity and cloud-based processing has become the foundation for easily replicated process management systems. So you no longer need to build a new data center every time you want to implement an industrial control system, where before the cloud, devices communicated with each other in a process called machine-to-machine (M2M) communications. The cloud and connectivity have now made massive IoT implementations powerful and practical, increasing the power of IoT devices while simplifying the implementation of digital feedback loops.

Autonomous vehicles (AVs) are an excellent example of real-time digital feedback loops solving complex engineering challenges. They depend on many technologies including IoT connectivity, sophisticated edge computing, and cloud-based AI platforms. When an AV is driving, it's not consulting the cloud to make decisions about whether to make a turn or stop or accelerate. All those decisions happen locally. Yet the machine learning models and deep neural networks built to empower autonomous driving are in the cloud, and they're built over petabytes and petabytes of training data that help continuously improve the vehicle's performance. When an AV has issues—for example, if the user grabs the steering wheel and disengages it—those exceptions



are captured and sent to the cloud platform for retraining so that the vehicles become smarter and smarter. This feedback loop—this interaction between the edge and the cloud—is a critical success factor.

All these emerging technologies are moving forward together, and they are pushing the development of other dependent technologies. For example, the need for better connectivity to IoT devices is one factor driving the development of 5G. As 5G matures and 5G networks become available, it's likely to change how we connect IoT devices. Today, the typical approach for connecting a large array of on-premises IoT devices to a cloud-based processing solution involves connecting through an intermediate, server-class gateway device. As 5G connectivity becomes closer to reality, some companies are starting to think about 5G as a better Wi-Fi. With 5G, it may be possible to directly connect all those IoT devices without having to deal with a complex topology for connectivity.

It's worth noting that direct connectivity through high-performance 5G networks will not likely obviate the need for edge computing. Bandwidth comes with a cost, and there will continue to be natural advantages to moving computing out to where data are generated. I expect that 5G will unleash a massive amount of connectivity that will power digital feedback loops. This evolution is likely to involve having computing power at more areas and at different levels of the system topology.

"The bigger frame of this transformational shift happening across businesses right now is the connection of all the disparate, disconnected aspects of a business through the digital feedback loop."

Another trend resulting from the interconnectivity digital feedback loops is the growing importance and awareness of cybersecurity in devices that traditionally were not considered cybersecurity risks. For instance, in the past, industrial control systems typically were not connected to information technology (IT) systems. However, as intelligent controllers become connected devices, hardware and software teams that work on them are realizing that a connected but always up-to-date device is fundamentally more secure than a disconnected but never-updated device. Attacks like Stuxnet, which target industrial systems, have taught us that air-gapping techniques don't work.

IoT devices, platforms, and development tools, even at the microcontroller level, must support well-founded security controls and strategies. With Windows being the most attacked operating system on the planet, we have worked for a long time to develop sophisticated techniques for dealing with cyber-physical security. Design engineers must recognize that security is a huge part of the transformation required as we create incredibly robust and pervasive digital feedback loops. Continued improvements in AI and deep learning, edge computing capabilities, and IoT connectivity are creating amazing opportunities for both businesses and the hardware and software engineers who support them. In this connected world, many of these solutions must work together, which means growing collaboration between hardware and software teams working on these emerging technologies. M

Sam George
Director, Microsoft Azure IoT Engineering

The Road from Automation to IoT-Enabled

by Jason Shepherd – IoT and Edge Computing CTO, Dell Technologies

The Internet of Things is moving beyond the "hype" stage to practical application on a wide scale. The next phase involves overcoming psychological, business, security, and operational challenges to harness networked intelligence to meet the industry's evolving technological needs.

Industrial Internet of Things at a Glance

Industries ranging from retail to health care are experiencing the transformative impact of the Internet of Things (IoT), but the manufacturing sector is currently in the midst of the Fourth Industrial Revolution—and, with it, the rise of the Industrial Internet of Things (IIoT), also known as Factory 4.0. The IIoT enables disruption and innovation in manufacturing on a scale not previously imagined, unlocking opportunities spanning from operational efficiencies all the way to enhanced decision-making capabilities at the executive level.

The IIoT enables manufacturers to acquire and take advantage of far greater amounts of data much more quickly and efficiently than was possible just a short time ago. It can be tapped to enhance predictive maintenance, improve quality control, cut costs, and boost safety. Business leaders can engage the Industrial Internet of Things to connect all their people, data, and processes from the factory floor to the executive level, driving company-wide efficiencies and better strategic decision making.

Key Facts

- Global manufacturers are expected to invest $70 billion in the Industrial Internet of Things by 2020.
- Accenture anticipates that the Industrial Internet of Things could add $14.2 trillion to the global economy by 2030.
- In a recent article at *WIRED*, Robert Schmid explains how IIoT technologies help change the way products are made and delivered. The IIoT can make factories more efficient, ensure better safety for human operators, and drive considerable cost savings.

Mouser Manufacturers Leading the Way

- Semtech recently released next-generation long-range (LoRa) devices and wireless radio-frequency (RF) technology (LoRa technology) chipsets. These advanced chipsets are designed to improve the performance and capability of IoT sensor applications that require low power, have a small form factor, and use LoRa wireless connectivity.
- Carnegie Technologies and Semtech announced at the 2018's Mobile World Congress Americas conference that they are collaborating on LoRa solutions as part of Carnegie's launch of its Longview IoT solution.

See Also:

- *Methods* eZine: Data Security
- Article: "Trends in 21st Century Factory Automation"
- Article: "Smart Factories Need Smart Machines"
- Article: "Factory 4.0: The Biggest Problem May Be the Noise You Can't Hear"

"When designing solutions for the IIoT, the name of the game now is "flexible architecture" to enable an improved customer experience through rapid, software-defined innovation while meeting key stakeholder needs."

[CONT'D ON NEXT PAGE]

The Industrial Internet of Things (IIoT) is a concept, not a thing. In my role at Dell Technologies, I hear customers saying, "I want to buy some IoT." To which I joke, "Really? What color would you like?" An IoT solution represents a broad set of capabilities coming together to drive an outcome, and it's challenging stuff. The IIoT doesn't always involve the cloud or even the Internet. It involves any sort of networked intelligence that drives business gain.

Many organizations have been building and running IoT-like solutions since long before it was even called the Internet of Things. What's new in the industrial space is the notion of connecting long-running automation processes in the physical world to the digital world, beginning with real-time monitoring for better visibility and then moving to optimization through analytics.

Today's big IoT drivers include lower silicon costs, increased connectivity, and the rise of the cloud; equally important, however, is the building of major tech trends over the past 20 years, like the rise of the Internet, e-commerce, mobile communications, and social media. The result is an increasing demand for real-time information and a pressing need to stay competitive as the world rapidly evolves.

One trend that drives the IIoT more than anything else is the maker movement. With the tools available today, it has never been easier for design engineers to prototype new ideas rapidly. In turn, this ability is accelerating change in traditionally conservative markets, as innovations yield better outcomes at lower price points.

Key Challenges to Address

The top two challenges with the IIoT have nothing to do with technology. One is business—finding a reason to add risk and complexity into the life of a business for increased proficiency, productivity, and profit gains. Two good reasons for this challenge always exist:

- Saving money through improvements in efficiency and quality
- Making money with new business models and customer experiences

The second major challenge is people—more specifically, addressing the needs of people in three key stakeholder groups:

- Operations technology (OT) is historically about overseeing operations rooted in the physical world; for these folks, it's all about uptime, efficiency, and quality.
- Information technology (IT) is about scaling out computing, networking, and communication systems. The IT sect of an organization cares about information security, manageability, privacy, and governance.
- Lines of business (LOB) leaders are concerned about top-line revenue, which is often the real driver behind IIoT projects.

Each group is motivated by different things, and within these key stakeholder groups are many subgroups with conflicting goals. For example, in OT, the production supervisor's uptime and throughput goals are often at odds with the goals of quality and safety managers.

Security is an especially important consideration in the OT world because a successful attack often has an immediate impact on production or—worse—life and limb. That's why OT has historically practiced security by obscurity—for instance, by segmenting critical operations off from broader networks and the Internet. Much depends on the degree of risk, which relates to a specific use case. For example, connecting a control system to the cloud in a nuclear power plant is quite different from connecting a system that simply monitors the energy use of a building.

And that brings up a foundational catch-22 for the IIoT. To drive new outcomes, OT systems must be connected. To keep connected devices secure, they must be regularly updated; updates that aren't properly scheduled can cause unplanned downtime, which is a big "no-no" in OT. Therefore, a key challenge for design engineers is balancing the needs of various stakeholder groups by enabling security and manageability to scale without affecting production uptime.

Finally, IIoT solutions are inherently heterogeneous, and the closer you get to where physical processes run, the more fragmented and customized technology choices become. You will see this behavior in hardware requirements involving form factor, input/output (I/O), and ruggedness. Software also becomes more complex, especially when it comes to highly constrained microcontrollers and control systems running embedded applications on a real-time operating system to react in deterministic real time.

Getting Started: Thinking about Connectivity

Despite an increase in the number of IoT projects, I like to say we're in the "AOL stage of IoT"—in other words, just getting "things" online at scale. IIoT solutions generally start above the systems performing closed-loop process control. These control systems continue to be purpose-built, and it's important to maintain a separation between the process and any overarching networked intelligence.

New industrial control systems are typically natively Internet Protocol (IP) capable, which simplifies connecting them to systems that perform monitoring and analytics functions. However, most existing industrial equipment communicates via non-IP wired transports such as a 4–20mA current-loop I/O or traditional field busses, and you can't just rip this equipment out. Instead, these systems require retrofitting along with gateways to speak to IP networks.

IoT systems require connectivity both southbound to sensors and northbound to applications, so which technology is best? The answer (like many things in the IoT) is: "It depends."

Wired sensors are typically less expensive and generally the most reliable options, but it can be cost prohibitive to pull cables to necessary data monitoring locations. In addition, although wireless is definitely convenient, many OT professionals still believe that it is unreliable. This may not always be the case, but industrial environments can present clear challenges to range and performance caused by obstructions and radio interference.

When planning for wireless sensors, battery life is also a key consideration. The higher the throughput and range a wireless sensor has, the more power it consumes, and this consumption increases the expense of a battery replacement in the field. In some cases, OT can avoid this cost by using energy-harvesting technologies such as those from EnOcean, but such options aren't suitable for all applications.

When designing net new systems, including Ethernet is a given. To communicate with legacy equipment, however, you must also plan for legacy wired transports, such as serial or controller area-network buses, depending on your target use case.

When it comes to wireless, Wi-Fi and Bluetooth Low Energy (BLE) are often built in with the chipset. BLE can be attractive for general-purpose industrial use cases because it also allows the use of a smartphone as a gateway. LoRa is attractive because of its long range, but it is suitable only for nonmission-critical notifications because of its low transfer bandwidth and use of unlicensed spectrum. For more process-intensive use cases, consider industrial-grade wireless protocols like WirelessHART. For backhaul, cellular is an important option, as is satellite in the case of systems that will be deployed in extremely remote locations. When feasible, design products with multimodal transport capabilities, including taking advantage of software-defined radios to enable dynamic changes.

Beyond the many choices for transport-level connectivity is even more fragmentation in the application layer to support specific devices that communicate over any number of protocols. Over time, we'll continue to see strides in connectivity standardization, but it's unreasonable to expect that there will ever be one protocol to rule the world.

Key Solution Architecture Considerations for IoT Scale

The many IoT platforms out there today combine various degrees of security and management tools in with their application development capabilities. When building a system of systems brought together by different service providers, it's inevitable that many different application stacks will touch any given infrastructure—a scaling nightmare if every application stack has a unique way to secure and manage the infrastructure.

Security

With security, it's not a question of when there will be an attempt to hack your operations. Rather, it's all about how prepared you are to remediate an attack. The necessary "in-depth" tools to mitigate risks at various levels of defense—ranging from the root of trust to identify and access management, encryption, and threat analytics to data diodes—are all there. The biggest issue in the IoT is

[CONT'D ON NEXT PAGE]

what I call security usability, and, for this, design engineers need to strike a balance between creating easy-to-deploy solutions and locking things down.

Managing Data and Infrastructure

Users often start their IoT journey by hastily hard-coding a selection of devices to the cloud so that they can start building applications. They often quickly realize that their only option is to pay again to get their own data back. There's a need for solutions that not only make it easier for users to migrate compute workloads from the cloud to the edge but also that break hard-coupling to specific application stacks so these users can control their own data destiny. Further, many developers are often fine with brute-force command-line management in the prototyping stage, but they quickly realize that they need a true enterprise-grade solution for production.

Addressing Scaling Challenges

To address these scaling challenges, invest in platform- and application-independent security and manageability tools designed with usability in mind. In addition, consider taking advantage of cloud-native architectural principles in as many places as possible, from the edge to the cloud, while recognizing where embedded software is still required for constrained devices and real-time control. By cloud-native principles, I mean optimizing software for hardware and operating system independence and using loosely coupled microservices bound together through application programming interfaces (APIs). By using these principles, you can write individual functions in any language and then deploy and update them without bringing down complete systems, thereby addressing OT's top concern about uptime.

"What's new in the industrial space is the notion of connecting long-running automation processes in the physical world to the digital world, beginning with real-time monitoring for better visibility and then moving to optimization through analytics."

Efforts like the vendor-neutral EdgeX Foundry project are working to alleviate protocol fragmentation and cloud lock-in by building an open, cloud-native edge framework with de facto standard APIs to bring together an ecosystem of device interfaces and applications, regardless of the underlying protocol, operating system, or hardware in use. As an open source project, EdgeX also minimizes the amount of "undifferentiated heavy lifting" for design engineers, so that they can focus on customer-valued innovation.

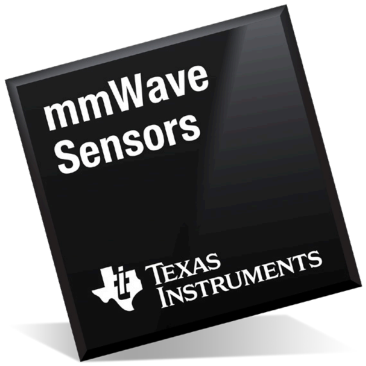
Looking Forward

When designing solutions for the IIoT, the name of the game now is "flexible architecture" to enable an improved customer experience through rapid, software-defined innovation while meeting key stakeholder needs. This goal includes investing in more consistent security and manageability tools while embracing cloud-native architectural principles from the edge to the cloud.

Open interoperability across heterogeneous parts is also critical for enabling customers to realize the true potential of the IoT, which leads to sharing and monetizing data across a system of systems that spans public and private entities. For this to happen at scale, you need pervasive trust, which isn't possible without open plumbing.

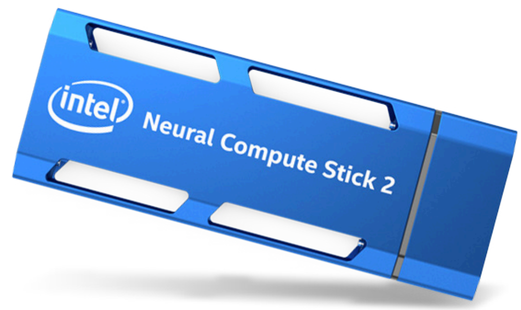
If you've ever seen the movie My Big Fat Greek Wedding, you'll know what I mean when I say: Blockchain, artificial intelligence, 5G, and augmented reality are current "Windexes" of technology. Don't get me wrong; these technologies are all set to transform society as we know it. But, they're also deep in the hype cycle and not panaceas. A good gauge for knowing whether a technology trend is still in the hype stage is when you hear more about the technology itself than the value of using it. This is where we've been with the IoT for the past three or four years, but we're starting to turn a corner.

We'll see a lot of progress with these and other technology trends this year. My recommendation to you as a design engineer is to be careful about getting caught up in the hype. Focus on creating architectures for foundational flexibility. In doing so, you will allow your customers to start seeing value today as they dynamically swap out solution components as needs evolve and emerging technologies mature. M



Texas Instruments mmWave Sensors

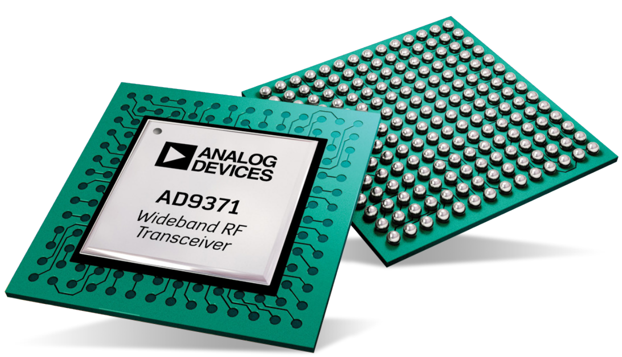
mouser.com/ti-mmwave-sensor



Intel® Neural Compute Stick 2

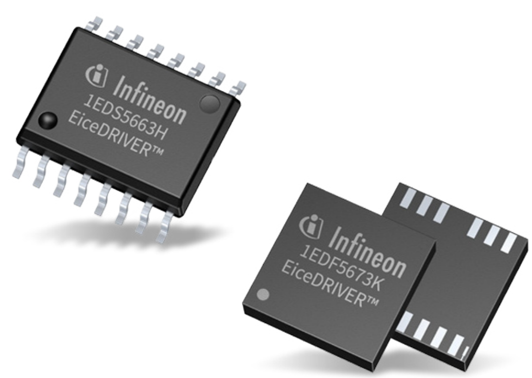
mouser.com/intel-neural-compute-stick-2

AHEAD OF WHAT'S POSSIBLE™



Analog Devices RadioVerse™

mouser.com/adi-radioverse



Infineon Technologies CoolGaN™ Gallium Nitride HEMTs

mouser.com/infineon-coolgan-hemts

A Glimpse Inside 5G: What to Expect

by Waleed Ejaz – Assistant Professor, Applied Science & Engineering, Thompson Rivers University

The primary goal of 1G to 4G was to increase data rates. However, 5G has a bigger agenda that includes low latency communications, connectivity for massive numbers of Internet of Things devices, efficient spectrum use, and more. This article looks at where 5G is and what that means for engineers.

5G at a Glance

5G is coming, but what does that really mean? Dreamers dream of 5G Internet of Things (IoT) networks and instantaneous broadband connectivity among people, devices, and infrastructure, but real things are beginning to happen. As of October 2018, Verizon is offering 5G home-Internet service in four cities: Houston, Indianapolis, Los Angeles, and Sacramento. AT&T is also beginning to offer 5G services in several cities. Most of the licensing for commercial 5G offerings is in the millimeter wavelengths, with frequencies between 30GHz and 300GHz. This is an important point because radio-frequency (RF) signals at these wavelengths do not penetrate objects, which means that part of the 5G challenge is designing short-range, line-of-site networks that support devices in motion (e.g., phones, cars). The best sign of progress may be rumors of the imminent release of 5G cell phones from several major manufacturers.

Key Facts

- In an October 2, 2018, PC Magazine article, Sascha Segan says, “4G will continue to improve with time. The upcoming Qualcomm X24 modem will support 4G speeds up to 2Gbps. The real advantages of 5G will come in massive capacity and low latency, beyond the levels 4G technologies can achieve.”

- In an August 2018 MIT Technology Review article, Elizabeth Woyke talks about how 5G can make cellular vehicle-to-everything (C-V2X) practical. C-V2X is a vehicle-to-vehicle (V2V) communication protocol being tested by many autonomous vehicle manufacturers.

- Qualcomm’s simulated 5G tests show that median 5G browsing speeds will be 7 to 23 times faster than median 4G browsing speeds, with median download speeds 10 to 18 times faster with 5G.

Mouser Manufacturers Leading the Way

- Although Qualcomm has not yet made any official announcements, there is wide speculation in the industry that the company is on the verge of releasing a chip for 5G cell phones. They did, however, recently announce their 5G Snapdragon modem.

- Intel accelerates timing for its XMM 8160 5G multimode modem to support broad global 5G rollouts.

- Broadcom introduces the industry’s first 5G radio switch

See Also:

- RF Wireless Technology

"Ultra-reliable, low-latency communications benefit applications such as autonomous vehicles, mission-critical applications, and industrial operations."

[CONT'D ON NEXT PAGE]


5G

The fifth generation (5G) of cellular communications is still in development and only just beginning to appear in limited commercial applications. 5G is designed to address a range of requirements that earlier generations could not. All prior advancements from 1G to 4G were driven by the primary goal of increasing data rates. Further increases in data rates are important; however, 5G is designed to address so much more:

- **Enhanced mobile broadband:** This feature enables faster data rates, which will support functions like 3D video applications, online gaming, and augmented reality.

- **Support for the Internet of Things (IoT) communications:** IoT devices themselves may not require high data rates, but Industrial IoT (IIoT) solutions can include massive arrays of IoT devices, many with different data formats. 5G must support large volumes of heterogeneous data to be valuable in industrial applications, smart homes, and smart cities.

- **Reliability and low latency:** More and more artificial intelligence (AI) applications are remotely handling industrial data for real-time process control. To support this capability, 5G needs high reliability and low latency. Ultra-reliability and low latency communications benefit applications such as autonomous vehicles, mission-critical applications, and industrial operations.

- **Support communications for fast-moving devices:** As high-speed trains become more common, 5G must be able to support communications with people and things moving up to 500kmph.

- **Greater spectrum efficiency:** Spectrum is a limited resource, and with rapidly increasing numbers of diverse connected devices, 5G will need to support more efficient use of this limited spectrum.

- **Greater energy efficiency:** As sustainable engineering is becoming an increasingly important consideration, the growth in numbers of connected devices means that 5G must operate with greater energy efficiency.

Meeting these requirements has resulted in several differences in the way 5G works compared with earlier generations. First, commercial 5G applications will operate in multilayer frequencies including lower frequencies (below 2GHz), medium frequencies (2 to 6GHz), and higher frequencies (above 6GHz). Further, millimeter waves consist of spectrum between 30 to 300GHz. That's much higher than 4G networks, which operate at lower frequencies. Particularly, millimeter waves offer several advantages including enhanced mobile broadband. Providing higher data rates to a growing number and variety of devices requires more spectrum and more efficient use of that spectrum. Currently, more spectrum is available in the millimeter wavelengths, although moving to these frequencies introduces new challenges, such as a shorter broadcast range and the inability to penetrate objects. Still, millimeter waves are critical to the success of 5G.

Another difference between 5G and earlier generations is the use of the 5G duplexing mode, which offers flexible and dynamic use of spectrum. In full duplex mode, the downlink spectrum resource can be useful for uplink transmission and vice versa. Previous generations always treat downlink and uplink resources separately. 5G duplexing mode offers more flexibility to dynamically use paired or unpaired frequencies.

Two questions on everyone's mind:

- How solid is the new standard?

- What is the status of 5G engineering?

The Third-Generation Partnership Project (3GPP) is in the process of developing the 5G standard. This collaboration of groups was originally founded to establish a global 3G specification, but its role has expanded to maintaining follow-on cellular generations, including 5G. The first 5G specifications were published in Release 15 of the 3GPP standards, which came out in 2018. 3GPP will enhance the specifications (also called phase 2 of the 5G specifications) in Release 16, which will release in 2019. Release 16 is expected to include the definition of related band-duplexing mode use cases.

Making 5G successful is a collaborative effort, with many industry players participating. Companies like Qualcomm, Huawei, MediaTek, and Intel are working on 5G components, and they likely all have 5G test beds. Qualcomm has released a 5G modem—the Snapdragon X50—that many companies are using in trial versions of their products. Some companies have begun deploying limited 5G services. South Korea tested a 5G network at the Winter Olympics in February 2018, and several service providers, including Verizon and AT&T, have announced that they will begin offering 5G home Internet access in some cities in 2019. Although a lot of testing is going on and mobile phone manufacturers, such as LG, are planning to launch their first 5G mobile in 2019, we aren't likely to see widespread use of mobile 5G for a while.

Nevertheless, there are many opportunities for engineers to begin designing 5G products. For those interested in taking that step, keep the following 5G engineering challenges in mind:

- **Greater spectrum efficiency:** In addition to the 5G duplex mentioned earlier, there are other approaches to using spectrum more efficiently. One is non-orthogonal multiple access (NOMA). Current generation Long-Term Evolution (LTE) and Long-Term Evolution Advanced (LTE-A) in cellular communications use orthogonal frequency division multiple access (OFDMA) to enable several users to share a carrier wave, an approach that has limitations—for instance, within a frame, one resource block can have only one user. However, some 5G engineers are thinking of ways to use NOMA, where one resource block is shareable by multiple users. Users will be distinguishable based on their power levels or other features. Engineers can now move to the implementation phase of NOMA.

- **Reconfiguration capacities:** Communications should be able to operate on multiple frequency bands with the same hardware.

- **Antenna design:** Engineers will need to explore and design antennas that provide effective communication in millimeter wave frequencies.

- **Cognitive radio technology:** NOMA and 5G duplex use spectrum more efficiently, but another approach to greater spectrum efficiency is based on the spectrum analysis by the Federal Communications Commission (FCC). According to FCC analysis, there is not a lack of spectrum but rather mismanagement of the spectrum we have. The thinking goes that plenty of spectrum is available, but it is scattered. Cognitive radio technology works to find and operate on available spectrum, a concept that requires reconfiguration at the hardware and software levels.

- **5G network densification:** The limited range and penetration of millimeter waves mean that 5G is moving toward small cells—much smaller than cells in earlier generations of cellular communications. Engineering challenges exist in how to design small cells, how to use them to create better coverage, and how these small cell sizes will communicate with each other and with a base station.

- **Massive connectivity of IoT devices:** How to provide massive connectivity for billions of IoT devices expected in the coming years is a significant challenge. How do you allocate spectrum, and how will these devices connect to the network continually? How will you satisfy their diverse requirements? Some IoT applications may require higher data rates; others may require lower data rates but also low latency. Engineers will need to satisfy a range of requirements for large numbers of IoT devices.

- **Energy efficiency:** Engineers are working on strategies to enable devices to harvest energy from radio-frequency (RF) power and other sources, such as renewables, while the devices are on the go.

The 5G specification is a work in progress. 3GPP is expected to complete Release 16 of the 5G specifications in 2019. The best way for engineers to stay abreast of this rapidly moving technology is to follow 3GPP's releases and keep track of what major players like Qualcomm and Intel are doing in this area. These organizations are great sources of information about new technologies and the results of 5G testing. M


5G

Artificial Intelligence: An Engineer's Perspective

by M. Tim Jones – SSD Frimware Architect/Managing Principal Engineer

As artificial intelligence advances ever more rapidly and becomes more widely used, understanding its potential and limitations is extremely important. For design engineers in particular, it's essential to consider hardware capabilities as well as training and implementation processes.

Artificial Intelligence at a Glance

Artificial intelligence (AI) is one of the hottest technology topics around right now because of how rapidly it is becoming an essential part of applications and devices we interact with every day. It's also an area of great interest for engineers, not only because of advances in AI and machine learning (ML) capabilities but also because of a growing trend in designing devices optimized for AI capabilities and function. Designing an AI solution, and especially a device that's optimized to operate with AI capabilities, requires rethinking physical elements such as human interfaces and the hardware platform. It also requires engineering the data science and algorithmic aspects of the solution that ultimately make it "intelligent." With all the interest in and wild claims about AI, this article takes a sober look at where the technology is, what's involved in building an AI solution, and resources that can help you learn more.

Key Facts

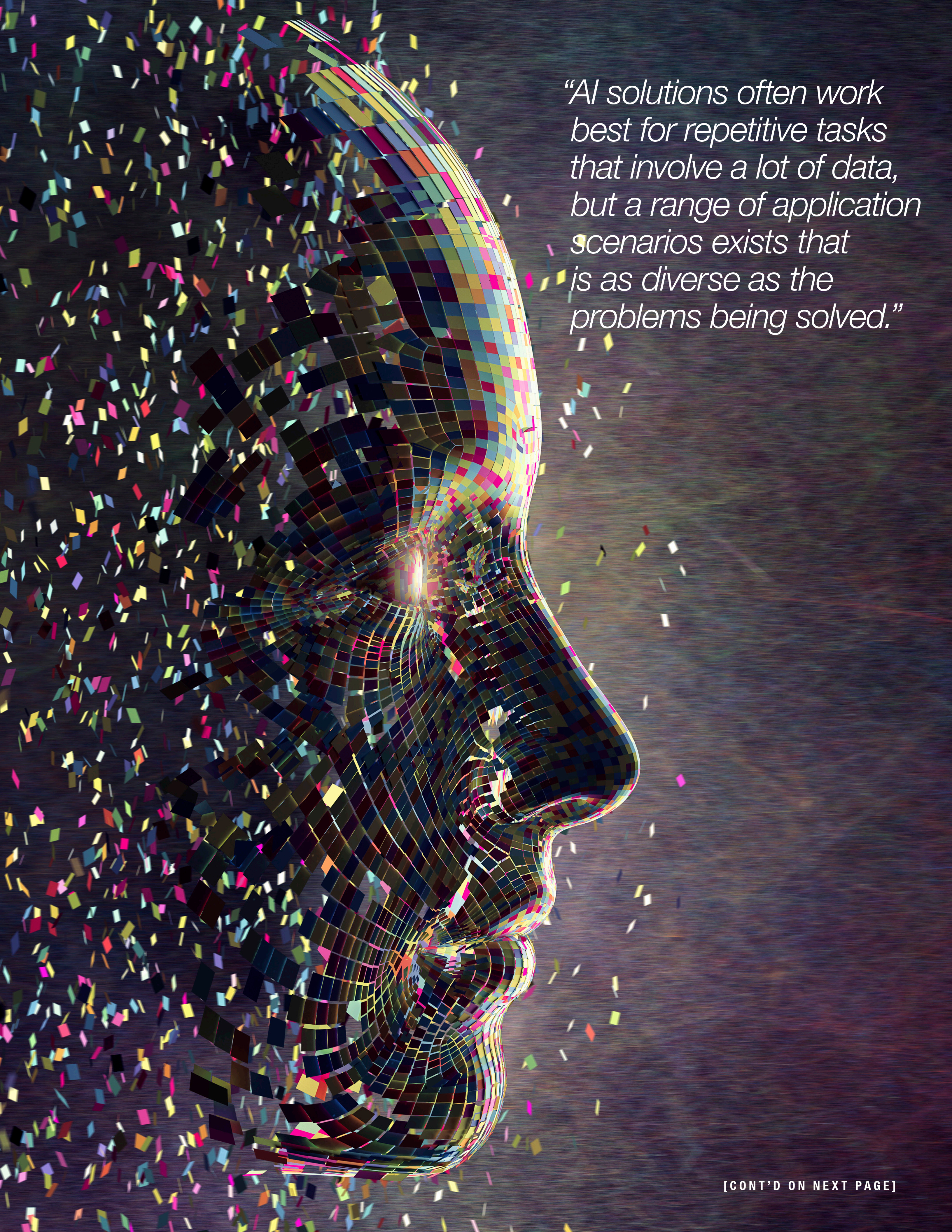
- In 1958, the first implementation of a neural network was developed in software for the IBM 704. It was next implemented as a pure hardware design, with the neural network weights implemented as potentiometers.
- IDC predicts that by 2019, 100 percent of all functional Internet of Things solutions will be supported by AI capabilities.
- According to Forrester, 9 percent of all new jobs created by 2025 will be the result of technologies such as AI, ML, and automation.
- Analysts expect significant advances in AI in 2019, particularly in terms of natural language processing (NLP), AI-optimized devices, and AI-optimized silicon chips.

Mouser Manufacturers Leading the Way

- On November 14, Intel introduced the Intel Neural Compute Stick 2 (Intel NCS 2), which is designed to build smarter AI algorithms and computer vision prototypes at the network edge. Based on the Intel Movidius Myriad X vision processing unit and supported by the Intel Distribution of the OpenVINO toolkit, the Intel NCS 2 affordably speeds the development of deep neural network (DNN) inference applications.
- NXP Semiconductors has released EdgeScale, a suite of cloud-based tools and services for secure manufacturing and enrollment of IoT and edge-computing devices. The solution provides a secure mechanism for developers to use popular cloud-computing frameworks in their applications and to remotely deploy and manage an unlimited number of edge devices.
- On September 12, 2018, Panasonic announced that it has partnered with Caspar to develop an AI smart home system.

See Also:

- Empowering Innovation Together: Generation Robot
- Blog: "A Look Forward at the Trends Shaping AI"
- Article: "Augmenting Intelligence: Enabling Neural Networks"



"AI solutions often work best for repetitive tasks that involve a lot of data, but a range of application scenarios exists that is as diverse as the problems being solved."

[CONT'D ON NEXT PAGE]

Artificial intelligence (AI) is an old idea rooted in the goal of building intelligent, conscious machines. Today's AI systems contain many elements, but one of its key definitions is an embedded set of capabilities that enables a device that would otherwise require human-level intelligence to function independently.

AI is inspired by human cognitive processes, but it's difficult to compare the two. One of the first AI models was the neural network, initially called a perceptron—a simple network with a single layer of neurons that implemented a binary classifier to solve simple classification problems. Multilayer networks soon followed that could solve more complex problems. Fast-forward to today, and we have deep learning systems built from many layers of diverse neural network types, solving problems that seemed insurmountable a decade ago.

Nature also provides inspiration for AI systems. Researchers have discovered optimization techniques by analyzing heating and controlled cooling in metallurgy (simulated annealing), Darwin's theory of evolution (evolutionary algorithms), and the way ants build paths through their environment (swarm intelligence). While we may not be constructing algorithms that mirror human cognitive processes, we've built models for distributed processing that are both capable and evolve over time.

Early AI focused on symbolic approaches to intelligence, but modern AI emphasizes machine learning (ML) and deep learning. ML uses statistical methods to give machines the ability to learn and adapt. Deep learning is a subset of ML that uses neural network architectures, which evolve far beyond multilayer networks into large numbers of specialized layers and new structures. Deep learning solves more complex problems, such as image processing and recognition.

Most industries today are benefiting from advances in AI. From the financial sector to insurance, medicine, security, and beyond, AI helps predict trends, classify risks, and protects us in our physical and virtual lives. Deep learning has emerged as the new platform for AI applications used in self-driving cars, detection of cancer from images, language translation, face detection and recognition, voice-based Q&A, and many forecasting and prediction solutions.

Developing AI Solutions

One key question for solution designers is when to apply AI. AI solutions often work best for repetitive tasks that involve a lot of data, but a range of application scenarios exists that is as diverse as the problems being solved. In most cases, the solution is made up of software and firmware, with hardware assists and accelerators such as graphics processing units (GPUs). It's possible to implement simple prediction algorithms by using neural network or clustering algorithms in microcontrollers. At the high end, a real-time facial-recognition system can reside in a cluster of enterprise-class compute and storage nodes.

A key job of the design engineer is to ensure that the hardware platform supports today's requirements, with the flexibility to scale up to tomorrow's applications. This platform work includes computing capabilities (involving processor speed and the right instruction sets), with memory and storage bandwidths to ensure that the processors are never starved of data. Engineers also need to recognize that AI systems are not magic solutions for every problem: AI systems have characteristics that, in some cases, may be problematic. For example:

- Although deep learning can outperform dermatologists in identifying skin cancer from images, the algorithm cannot explain why it came to its conclusion. Radiologists can offer evidence for why they believe a spot is cancerous or benign. Deep learning simply offers an answer with a degree of probability.
- ML algorithms typically focus on singular tasks and are unable to generalize their knowledge into other areas. Humans learn, and then apply that knowledge more broadly; ML is locked into its problem domain.
- In many cases, industries lack knowledge of where they can apply AI and ML. Design engineers with knowledge of AI algorithms, methods, and domains in their particular field will unlock new applications for deep learning and other ML approaches.

Implementing AI

Another design consideration is that AI systems must be trained. Training an AI model is a function of the algorithm and data set involved. A useful model relies on a diverse pool of data in the given problem domain: It involves collecting data (often aggregated from multiple sources) and then cleansing, normalizing, and optimizing it. Data science and data set management are key functions for training AI models.

A simple clustering algorithm or neural network can take minutes to train on a desktop computer, but it might take a deep learning network hours or weeks in a complex data set on a large cluster of servers. Part of the training includes a validation of the AI model. Training is first done with a training data set, and then validation tests use a portion of data, which the algorithm hasn't seen, to determine the model's accuracy. Testing a deployed model is crucial to ensuring that the model will remain accurate over its lifetime.

Once in production, inputs (involving audio, video, gestures, text, and other data) are presented to the model to generate an output (representing an identification, classification, prediction, control, etc.). Commonly, training and deployments are independent of one another, as a model is first trained then deployed to a population of devices. Once deployed, the model is static and doesn't change on the device unless updates occur.

Federated learning is a new method for large populations of devices. In this approach, an AI model is delivered and refined at the target device (learning locally). The model's refinements are sent to a centralized server where they average with refinements from other devices to update the model. The newly refined model then returns to the remote devices.

Looking Ahead

Given the growing popularity of approaches like deep learning, many changes are occurring in the engineering ecosystem. New processors with tailored instructions are accelerating the math behind neural networks and deep learning. Software frameworks are helping to train and deploy AI applications in the field. Existing products like GPUs are also evolving to help optimize deep learning flows. Deep learning algorithms themselves are also undergoing change. ML researchers are finding ways to shrink deep learning implementations to execute them on systems with fewer resources or lower latencies. SqueezeNet is one such implementation: Requiring five percent of the resources of competing approaches for certain problems.

Numerous skills are critical in designing and implementing AI solutions, including:

- Data science, which involves understanding not only algorithms but also end-to-end data flow, from gathering, cleansing, and training to validating an AI model
- Tools that are useful for prototyping, such as the R Language, and frameworks for AI model development and deployment, such as TensorFlow
- Python, with its massive library of AI modules, and C (the lingua franca of embedded developments), which are key building blocks
- Design engineers who understand the software/firmware and interactions with hardware that are a key part of the equation

These tools and environments provide useful resources and, combined with research from larger companies that are pushing the boundaries of ML (e.g., Google, Facebook, Microsoft, Apple), will help you get up to speed on AI and ML.

"Training an AI model is a function of the algorithm and data set involved."

Augmented Reality: Beneath the Radar

by Darryl Wright – Research Engineer, Georgia Tech Research Institute

Augmented reality is playing an increasingly important role in many industries. Improvements are being driven by increasing hardware densities, more powerful local processing platforms, improvements in wireless connectivity, and continuous improvements in development kits.

AR at a Glance

Many people still think of augmented reality (AR) and virtual reality (VR) as gaming technologies, but increasingly, AR/VR applications are appearing in other places. Health care is seeing increased use of AR tools to support remote consultations, remote surgery, and education. Automotive, aerospace, and other engineering-intensive industries are adopting AR solutions for design visualization and collaboration. A growing number of AR applications are being used in retail as visualization aids for consumers. Many see AR as eventually playing a key role in how we interact with the world around us—to the point that we take personalized AR enhancements for granted. Challenges exist in reaching that goal, including the data and compute-intensive nature of many AR applications, the design challenge of optimizing AR experiences for their intended purpose, and the convergence of technologies required to make AR's utility broadly scalable.

Key Facts

- In a recent article by In'saneLab, Antoni Zolciak cites data showing that the number of VR/AR users has increased from less than a million to more than 150 million in the past four years.
- A May 3, 2018, Information Age article projects that the top applications for AR over the next few years will be in the gaming, health care, and engineering spheres.
- A recent study by Capgemini Research Institute, focusing on the use of AR and VR in the manufacturing, automotive, and utilities industries, found that 31 percent of companies use VR or AR to view digital technical content, 30 percent use them to remotely connect users with experts, and 30 percent use them to view components digitally.

Mouser Manufacturers Leading the Way

- Intel's Augmented Reality brings Red Bull Rampage to your living room.
- TDK acquires sonar-on-a-chip inventor Chirp Microsystems to expand its portfolio of VR/AR sensors.
- Bosch wins Automechanika innovation award for its augmented reality application at technical training.

See Also:

- eBook: *Augmented Reality*
- Article: "Augmented Reality: Beyond Gaming to Real-World Solutions"
- Article: "Augmented Reality Hardware: Key to HMI"
- Article: "IMUs: Inertial Measurement Units"
- Applications: Medical

“The ultimate goal is to make using phone-based AR applications possible without holding the phone.”

[CONT'D ON NEXT PAGE]

Virtual and augmented reality are not new, but they are more capable and more pervasive in our everyday lives than ever before. Although we're not on the verge of routinely walking about in a real-time, digitally enhanced world, piece-by-piece the convergence of technologies needed to make this kind of enhancement a normal part of our lives is happening.

Flavors of Digital Enhancement

To understand where the technology is now and where it's going, it's useful to review the three different flavors of digital enhancement:

Virtual Reality

This is a totally immersive reality. It replaces everything that's around you, and it immerses you in a new reality, whether you're moving and interacting with objects or just looking around. Typically, it offers a 360-degree view of a virtual world. The most common applications of virtual reality (VR) today are in gaming, although there are VR apps beginning to appear such as the Google Earth VR that could have potential in education.

Augmented Reality

Augmented reality (AR) provides an informational overlay that adds digital elements to your live view, enhancing the real-world view around you. AR applications typically augment real-world views with data, but the data is not interactive with the real world. Examples would be a heads-up display (HUD) in a car to provide speed, navigation, and other driving information so you don't have to take your eyes off the road. Snapchat photo filters, phone-based stargazer applications, and furniture apps that scan a room and drop in properly scaled images of furniture from the IKEA catalog, are other familiar examples. AR applications are beginning to be widely used in engineering, education, and health care. For example, AR applications allow a surgeon to see blood pressure and other vital patient information in real time without having to look up from a surgical procedure. It is also being used by design engineers to see objects as they would appear in the real world.

Mixed Reality

Mixed Reality (MR) is a combination of VR and AR, where virtual objects are anchored in the real world. Examples include interior design apps or automobile apps which allow user to configure their dream car in the showroom floor. It is also used in medical applications where manipulating a real-world object also manipulates data associated with that object, such as in a 3D-image overlay. Other applications include engineering and equipment maintenance. In some applications, the distinction between AR and MR is blurred, and many consider MR a variation of AR.

AR: A Convergence of Technologies

AR is currently the most widely used non-gaming technology in this family of enhanced realities. VR, AR, and MR all share certain requirements, including a computer platform capable of combining all the image data, motion sensing, real-time display data, and audio data if that is part of the application along with a means for viewing the virtual or augmented reality images. Today's AR solutions run on a variety of visualization hardware types, including computer screens, smartphones, and wireless smart glasses, such as HoloLense and Magic Leap, that are used in AR and MR applications. HUD systems project data overlays on a transparent medium, such as the windshield of a car, or, in the case of civilian aircraft, on a small viewing screen in front of a pilot's eyes.

> ***"Piece-by-piece the convergence of technologies needed to make digital enhancement a normal part of our lives is happening."***

It's important to recognize that AR is not one technology. It is a convergence of technologies, and the more sophisticated the application, the more technical integration must take place. For example, an AR retail application that involves price comparisons might incorporate Global Positioning System (GPS) data, stored database data, machine learning, and analytics in addition to the computing platform and visualization technology. Likewise, the total package needs to be mobile in order to be practically useful, which means it must work on a smartphone. Engineering constraints of any AR, VR, or MR application are generally dictated by an application's functionality and system hardware capabilities. For example, a game running on a computer typically runs at 45 or 60 frames per second. However, in order to avoid motion sickness in a VR version of that game, the frame rate must increase. A VR headset has a screen for each eye, and each screen must have a minimum of 45 frames per second to avoid motion sickness, which means the application now requires a total frame rate of at least 90 frames per second. To get equivalent image quality in VR, you need to nearly double the frame rate.

The convergence of technology and performance requirements places a big load on computing platforms. Yet relying on cloud or network-based components can create latency issues that might deliver a poor real-time AR or MR experience. That's one reason why designers want as much processing done locally as possible. It's also why there's a lot of interest in building smartphone-based AR applications. Smartphones are powerful computing platforms with built-in motion and location functions, cameras, and image processors. The major operating system (OS) providers all have their own development kits for building AR and VR applications. Apple provides ARKit, Google's is ARCore for use on Android systems, and Microsoft recommends Unity 2D and 3D for AR and MR applications that use HoloLense.

The technological convergence demanded of AR solutions means more than just software and hardware integrations. Developing these apps requires a breadth of skills. AR and MR solution developers follow the practices of gaming developers who work as a team, that typically includes software developers, hardware specialists, database specialists, designers, user experience specialists, possibly sound specialists, and production managers. The size and skills required depend on the nature of the application and what it must do. It's worth noting that the user experience specialist is very important for the success of AR and MR applications. For example, this is the person who makes a HUD in your car a valuable driving aid, without being a dangerous distraction. AR solutions must be carefully optimized for their intended purposes.

Conclusion

Advances in AR and MR will continue as hardware densities increase, local processing platforms become even more powerful, wireless connectivity improves, and as development kits continue to improve. In the coming year, keep an eye on progress in these areas:

- Continual updates to ARKit, ARCore, and Unity
- Smaller, more compact smart glasses for AR and MR applications
- More research into headset controls, such as eye tracking and voice control
- More smartphone-based AR applications
- Efforts to create a more hands-free smartphone-based AR experience by linking more streamlined smart glasses to phones, with the ultimate goal to make using phone-based AR applications possible without holding the phone.

Although 2019 is not likely to see dramatic breakthroughs in AR and MR technologies, these systems will steadily improve and be more widely applied in many fields.

The Road to Level-5 Autonomous Vehicles

by Sudha Jamthe – CEO, IoT Disruptions

Most autonomous vehicle players today say that they're testing vehicles that are technically able to operate without humans at the controls, but humans are there just in case. What does this mean, and what's involved in getting to the next level?

Vehicle Automation at a Glance

The Society of Automotive Engineers' system for rating vehicle automation is based on the following scale:

- **Level 0:** No automation—humans fully control vehicles
- **Level 1:** Driver assistance (such as cruise control)—humans perform driving tasks
- **Level 2:** Partial automation—humans select multiple functions for execution, such as steering and speed
- **Level 3:** Conditional automation—systems monitor a driving environment (e.g., Tesla running on autopilot exhibits level 2 or 3)
- **Level 4:** High automation—systems drive a vehicle in all conditions, but a human driver can intervene if necessary
- **Level 5:** Full automation—systems drive a vehicle with no human driver and no human driver controls

Autonomous driving technology is advancing rapidly, with all the major players testing level-4 vehicles. Most expect to test level-5 vehicles on public roads within the next few years, and this activity has spawned its own ecosystem of specialized sensors, power systems, data processing, and software.

Key Facts

- In March 2018, the California Department of Motor Vehicles (DMV) gave approval to 52 companies to test autonomous vehicles on California roads and highways.
- In November 2018, Waymo received the first permit ever from the California DMV to road-test a level-5 autonomous vehicle (AV) with no human at the controls.
- It's estimated that in 2025, 8 million level-3 or higher AVs will be sold.
- Waymo AVs have logged more than 5 million road miles in 25 cities.

Mouser Manufacturers Leading the Way

- On October 29, 2018, the Volkswagen Group, the Intel Mobileye autonomous vehicle platform, and Champion Motors announced that they will deploy Israel's first self-driving ride-hailing service in 2019.
- In September 2018, Micron announced plans to invest in its facility in Manassas, Virginia, which produces high-performance memory chips for AVs.

See Also:

- eBook: *Internet of Moving Things*
- Block Diagram: Advanced Driver Assistance Systems Block Diagram
- Automotive Application: Central Body Control

"All the players in this space are striving for fully autonomous level-5 vehicles, and they're getting closer to that goal."

[CONT'D ON NEXT PAGE]

Most autonomous vehicle (AV) players today say that they're testing vehicles that are at level 4 (based on the Society of Automotive Engineers zero-to-five scale). That means these vehicles are technically able to operate without human input, but humans can take over if the AV gets into trouble. True, these vehicles typically run in limited geographies restricted by geofencing, but this feat is still an impressive accomplishment and the cars keep getting better. All the players in this space are striving for fully autonomous level-5 vehicles. Waymo, Google's AV project, is already preparing to test level-4 vehicles without a human backup driver on public roads in California.

Essential Capabilities

To understand what has been accomplished and what still needs to be done, consider these four essential capabilities a self-driving car must have to function without human input:

Localization

An AV must always know where it is relative to its known world. It must know its starting point, whether to turn, how to navigate around an obstacle, and where to detour when a road is closed. If the vehicle shuts down for a time, it must be able to put its current location on the "map." Notice that I say it must localize itself in its known world: This is because AVs are not likely to know the entire world. Level-4 AV testing typically begins with a restricted area. As the vehicle gets better at negotiating this area, testers expand its routes. When AVs become commercially viable, many will function in limited ranges and routes. Regardless of the size and shape of an AV's world, the vehicle must always be able to localize itself at all times.

Perception

An AV must be able to perceive the physical world around it, including recognizing roads, lanes, traffic signs and lights, other vehicles, pedestrians, obstacles, and all the things humans see and process as we drive. The most common AV sensor technologies include video cameras, stereo video, radar, and light detection and ranging (LIDAR)—a pulsed laser technology used for object detection and avoidance. Currently, no standard approach exists for the perception layer. Many players are testing various combinations of sensor arrays, and cost is obviously a factor. Whatever sensor array an AV has, it must be able to stitch together all the data from all those sensors to build a detailed and complete local map of its surroundings, which updates continuously in real time as the vehicle moves.

Path Planning

An AV must be able to make every navigational and operational decision in every moment as it's traveling. This decision-making goes beyond getting from point A to point B in its known world. The vehicle must also make decisions about what to do next if it stops, when it should speed up or slow down or change lanes, how to execute turns, what to do if it encounters an obstacle, and what to do if it encounters something it has never experienced before. Of course, it must be able to decide when to leave point A and what to do when it reaches point B.

Control

An AV must be able to reliably and accurately operate the proper sequence of driver controls to empower it to perform essential operations. If the AV decides it needs to go around an obstacle, and doing so requires a lane change, it must use its sensor array to ensure the path is clear to perform the lane change without violating traffic controls for that stretch of the roadway; it must then execute the proper speed adjustments, signals, and steering to perform the action.

All four of these capabilities are essential for an AV to function at all levels. Commercially viable level-5 AVs must exhibit these capabilities with considerable accuracy and reliability and be able to react to all road situations independently without human intervention. Achieving this level of reliability depends on another critical piece in an AV system: The artificial intelligence (AI) platform, with its deep machine learning (ML) and inference modeling, as well as its training to drive.

What's Happening Today

Much of the level-4 testing happening today is really training the system to react properly in every possible driving scenario. Typically, this work starts small, with the test vehicle traveling a narrowly prescribed route over and over. Through this process, the AV is being taught the basics through simulation. Every time it travels a test route, all its operations are logged. If the AV gives control to a human driver, it's called a disengagement, and that human intervention becomes part of the log. Every AV disengagement is reported to the DMV by regulation. When the car returns to the shop, engineers tweak the vehicle's AI brain, using what it learned on that test run. They then create a new inference model for the AV to use when the vehicle goes out again. This process is complex deep learning that involves processing large volumes of unstructured data and ultimately being able to make decisions in real time. Human interventions at this stage are a critical part of the training process.

As the AV masters a route, the testers expand the vehicle's range to take in new road situations and geographies. That new geography may cover a community and then expand to an adjacent community. Waymo has been training level-4 AVs in Phoenix, Arizona, and several other cities. It's worth noting that many of these testing programs cover routes and regions that would potentially support an AV-based business model, such as Mobility as a Service. Major players are being strategic about where and how they test their vehicles.

The driving knowledge the AI system gains is cumulative and transferable to other vehicles. At the level-4 testing stage, driving knowledge enhancements may look like beta code updates. When advanced AVs become commercially viable, they will receive regular updates of accumulated knowledge. The key is that the more miles AVs log, the better and more reliable they become because they're continuously learning systems.

"Major players are being strategic about where and how they test their vehicles."

What's Ahead

In the coming year, we will see more about level-5 AV road tests, and AV developers will continue to address the engineering challenges that remain. For example, current AVs do not fit into the social fabric of the road. If you honk at an AV, it will not know what to do. It cannot communicate with pedestrians or other drivers. Recent industry discussions have focused on establishing communications standards to avoid the chaotic situation of every AV manufacturer having its own communication protocols.

Level-5 AVs will ultimately have no need for driver controls, and there will be no reason for everyone to face forward. AVs will become high-bandwidth devices capable of streaming media for passenger consumption, and they will have to do so without disrupting the control systems and connectivity associated with a vehicle's operation. These design parameters call for a complete redesign of car interiors.

It's a tremendously exciting time for engineers in the automotive space. AVs are poised to transform the auto industry through sensory and control systems and vehicle ergonomics up to a greater diversity of purpose-built vehicles and an expanding role for AVs in the continued emergence of all-electric vehicles. It may all be happening faster than we think. M

Maker Movement: Innovation to Productization

by Bob Martin – "Wizard of Make" Microchip Corporation

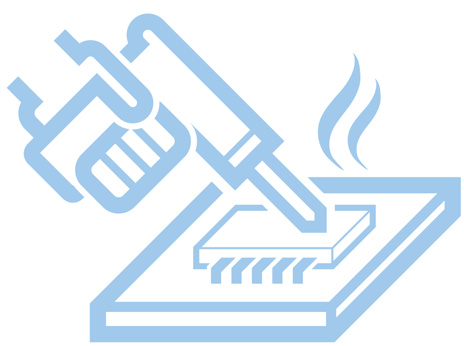
Three important trends are changing the way innovations become products: The speed to a prototype, access to markets, and the design impact of sensors. Design engineers need to know how developing trends in productization will affect them in the months ahead.

Maker Movement at a Glance

The number of technology "makers"—that is, people using basic electromechanical components to build something new—is increasing, and this increase is influencing professional electrical design engineers. These professional innovators are now incorporating products an amateur maker might use, such as a single-board computer or a module, into the engineering process. At the same time, the quality of the tools available to amateur makers is increasing, making it even easier and more cost-effective for professional electrical design engineers to incorporate these tools into their own work. The costs of iteration are lower, and iteration can happen more quickly. It's also easier than ever before to add sensors and take advantage of code examples to build relatively complex systems without investing a lot of time. All that adds up to a faster track from innovation to productization, and it increases opportunities for design engineers looking to bring their ideas to market.

Key Facts

- A Pivot International blog article says that for every seven product ideas only one will yield a successful product.
- Research by Lab42 suggests that more than a third of consumers like owning products that make them look innovative to others.
- A *WIRED* article entitled "The Sensor-Based Economy" cites research estimating that by 2020, 20.8 billion objects will be sensor equipped and Internet connected.



Mouser Manufacturers Leading the Way

- Microchip Technology, Inc. is making it easier and faster to develop remote Internet of Things (IoT) nodes with long-range, low-power technology.
- Texas Instruments now offers microcontrollers that can withstand high temperatures and reduce development costs for sensing applications.
- Silicon Labs gives developers the ability to prototype IoT applications in a single day with drop-and-connect simplicity: No software development necessary.

See Also:

- Technology Pillars: All Things IoT
- Dev Tools: Mouser's Development Tools Center
- Article: "A Combo for Innovation: Open Source and Crowdfunding"

"...give the product to other people and dare them to find problems with it."

[CONT'D ON NEXT PAGE]

Going from an idea to a successful product has always presented multiple business and technical challenges for innovators. Today, three developing trends have the potential to ease some those challenges even while they create new ones. To clear the path from innovation to productization, electrical design engineers need to know how those trends will affect them in 2019.

Trend 1: Speed to a Prototype

Speed is one key trend changing the way innovations become products. Thanks to 3D printing, rapid prototyping, and hardware accelerators, electrical design engineers can now turn their ideas into prototypes incredibly quickly and get feedback fast. That acceleration makes it easy to iterate, which benefits designers, their companies, and even their customers. It can also create the illusion that the entire innovation-to-productization process can and should speed up.

Take a pair of electrical design engineers with a brilliant idea for a product. They've built a version that works—at least under the right conditions—and they decide to take it to a Maker Faire event. After attracting positive attention, they go to a hardware accelerator with the goal of radically shrinking their prototype from, say, the size of a lunchbox to the size of a deck of playing cards. What happens?

Unless the system is very simple, there are so many changes happening all at once that the new, much smaller version is unlikely to work the first time. To complicate matters, with a complex system crammed onto a small board, it's difficult to test and debug the design.

Using an incremental approach in this case is slower but more effective. Reducing the product size in stages gives you the chance to debug interim boards and optimize the layout. Our pair of engineers may resist this step because the interim board layouts take time and incur costs, but at the end of the day, they'll save both time and money. The incremental approach pays for itself because problems can be isolated and solved throughout the process.

Trend 2: Access to Customers

Easy access to customers is another trend that affects the innovation-to-productization cycle. In fact, it can lead to the valley of death. To understand what that means, let's revisit our pair of engineers. They've made it to the stage where they have a viable product and have even sold a few hundred units online. Then, a major retailer notices the product and places an order for a few thousand units.

Fulfilling that order requires a big upfront investment in everything from material to equipment. Our engineers assume that they'll be able to fund that investment with the money from retail sales, but retailers often pay slowly, potentially just once a quarter. Now the engineers are at risk of falling into the valley of death, which is the time gap between when money goes out to pay for upfront costs and when revenues come in.

The valley of death can destroy a young company, no matter how great a new product is. To avoid it, innovators must secure sufficient capital before they try to fulfill a big order. Here's an important note about what sufficient truly means: Unexpected problems are guaranteed to pop up during the manufacturing and distribution processes, so the innovators should secure more capital than they think they'll need.

Design engineers face another (similar) risk when turning innovations into products: Death by crowdfunding. In this case, our pair of engineers decide to ask for $25,000 to build 500 units but end up receiving more than $1 million in pledges. Now they must build 20,000 units. Unfortunately, they simply can't do it because they'll need years to scale up to that level of production, and few customers will wait that long.

Trend 3: The Impact of Sensors

A third trend that design engineers need to consider when going from innovation to productization is the impact of using sensors. So many devices are now equipped with sensors that interference can become a problem. For example, radio-frequency interference (RFI), such as from antennas, can cause inaccurate sensor readings. Engineers should test sensor results to ensure that they're accurate and repeatable.

Another sensor-related challenge is durability. Just because a sensor-equipped product worked well for a few days in an indoor, climate-controlled environment doesn't mean the product is ready to ship. Again, engineers should test the sensors under various conditions and for varying periods to ensure that they'll last in real-world usage situations. In fact, even if the product isn't equipped with sensors, it's an excellent idea to give the product to other people and dare them to find problems with it.

Getting Help: Hardware Accelerators

Fortunately, design engineers don't have to cope alone with the potential challenges that these trends present. Hardware accelerators and product development companies offer many ways to connect with helpful resources. They host events that are usually low cost or free as well as offer the chance to network. Meeting people in such spaces is invaluable because they can offer feedback that could save you a lot of time. People at these events also know other people who can offer manufacturing resources, venture capital, or simply useful connections.

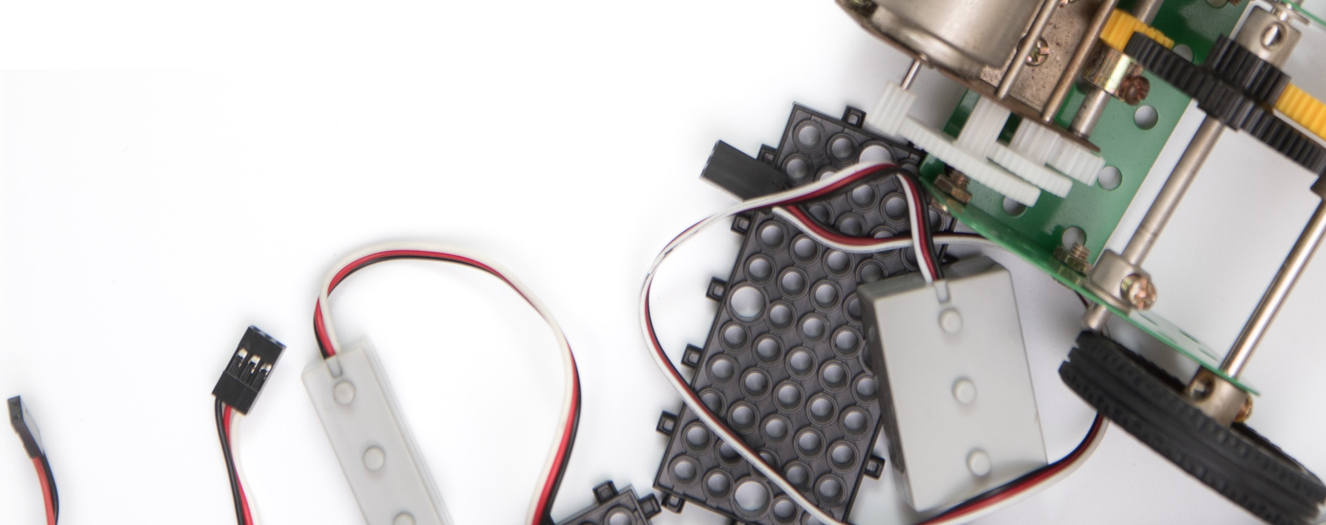
Some design engineers worry that telling other people about their innovations creates the risk of idea theft, yet you can always ask for someone to sign a nondisclosure agreement. And, remember: Professionals running hardware accelerators must preserve their reputations to remain successful. They don't want to risk becoming known as untrustworthy.

"Hardware accelerators and product development companies offer many ways to connect with helpful resources."

Hardware accelerators can also offer insights into the specific needs of individual markets. After all, consumers want products with different characteristics than business users. Consumers want intuitive user interfaces with simple, easy-to-understand menus and clearly labeled buttons. Business users want products that work reliably and won't disrupt their processes. An electronic cash register, for example, needs a quick error-recovery process. Otherwise, a cashier could spend valuable time trying, and failing, to restart it while customers grow frustrated with the long line and possibly decide to go elsewhere to make their purchases.

Industrial users want products with robust designs that will work as advertised. However, when industrial products are distributed at scale, they're often treated roughly and subjected to vibrations or shock that could damage key components. The connections you make through a hardware accelerator can help you anticipate and prevent these kinds of problems.

In addition to making the production of an innovation easier, connecting with others can spark new ideas. Technology is waiting for the next truly disruptive innovation. Perhaps it will come from you, but remember: An innovation can succeed only if it's successfully productized. M



Embedded Hardware Development with MicroPython

Interview with Limor "Ladyada" Fried – Founder, Adafruit

Connected devices are getting smarter, thanks in part to advances in programming languages. These languages shorten development cycles for embedded hardware solutions, giving microcontroller programmers the power to not just innovate more quickly but also check for errors more thoroughly.

MicroPython at a Glance

These days, people are paying a lot of attention to the Internet of Things (IoT) and how connected devices are changing just about everything. The engineering behind this phenomenon centers on embedded hardware development, which is creating smarter IoT devices and pushing more computing power to the network edge. Several factors are making this advance possible. One important advance is the growing popularity of MicroPython, a version of the Python programming language specifically designed as a simple language for quickly developing programs that run on microcontrollers that have limited processing speed and memory. MicroPython makes it possible to shorten the development cycle for embedded hardware solutions, which is an important consideration when designing IoT products.

Key Facts

- Gartner predicts that the market for "connected things" will continue to grow by more than 30 percent per year through 2020.
- Based on a Stack Overflow Trends report, Python is currently the fastest-growing computer language in the world.
- In a recent article at *Electronic Design*, William Wong explains how MicroPython provides advantages for embedded systems that require shorter development cycles and platform portability.

Mouser Manufacturers Leading the Way

- Adafruit's recent partnerships with Microsoft MakeCode and Cartoon Network will help inspire the next generation of engineers. And Ladyada's recent awards from Inc. and Forbes further solidify Adafruit's place in the future of electronics.

See Also:

- Tools: Embedded Development Tools That Support MicroPython
- Accessories: MicroPython-Related Accessories
- Blog: Master Chocolate, Army Knives, and Multi-Dev Tools

"MicroPython and CircuitPython have no pointers or memory alignment issues to worry about, and string and file parsing are built into Python natively."

[CONT'D ON NEXT PAGE]

MicroPython and a variant, CircuitPython, are enjoying increasing popularity among microcontroller programmers. To learn more about why this is so and how to apply MicroPython, we reached out to Limor Fried—a leader and innovator in this space. Here's what she had to say.

Q. What are MicroPython and CircuitPython? How do they stack up to the Python installation you may find on a client or server machine?

A. Damien George created MicroPython in 2013, as an implementation of Python designed to run on microcontrollers—processors with limited speed, RAM, and flash memory relative to more sophisticated microprocessors. Damien implemented MicroPython from scratch, making design decisions that favored efficient use of the limited resources available on these chips.

The core language of MicroPython is close to Python's version 3.4, with some rarely used features omitted or slightly changed. Many of the "batteries included" libraries that ship with the standard implementation of Python are not included, largely for space reasons. In addition, MicroPython typically has either no underlying operating system or a simple, real-time operating system. Many operating system features, such as file system access, MicroPython or CircuitPython must provide directly or through specialized libraries. Both MicroPython and CircuitPython include natively implemented Python modules that enable "close-to-the-metal" access to the on-chip peripherals that microcontrollers provide.

Scott Shawcroft, working for Adafruit, started CircuitPython in 2016. The original idea was to port MicroPython to the SAMD21 series of microcontrollers, which Adafruit uses extensively in its products. Eventually, because the goals of MicroPython and Adafruit were different, Adafruit forked MicroPython and named the fork CircuitPython. Adafruit considers CircuitPython a "friendly" fork: CircuitPython incorporates updates from MicroPython regularly, and Adafruit keeps in touch with Damien about what both organizations are working on.

Why would someone use MicroPython/CircuitPython for an embedded project instead of traditional C/C++ tools?

As with Python, MicroPython and CircuitPython allow for a quick development cycle. You can write the code quickly and make changes and retest code quickly. The higher-level language features of Python, such as automatic storage management, easy string manipulation, and variable-length sequences, reduce program size and remove a lot of tedious code. Mapping and filtering a list, for example, require 20 to 30 lines of C/C++ code but just a single line of readable Python code.

MicroPython and CircuitPython have no pointers or memory alignment issues to worry about, and string and file parsing are built into Python natively, which make it a great language for the Internet of Things and user interface products. Using Python for embedded applications is never going to be as fast or as memory efficient as using C/C++ or similar compiled languages, but an embedded application often doesn't need to be that fast or small. When the underlying technology is more powerful and complicated, it's important to be able to get something working quickly and to change it on the fly. For example, consider the user of embedded computers in a laboratory environment or a flexible manufacturing environment. Changes can be made and tested much more quickly with a language like Python.

What's the typical edit/debug cycle in a MicroPython/CircuitPython environment? Do you typically develop directly on the target hardware, or do you cross-develop using a separate workstation?

A big advantage of MicroPython is that it has a faster development cycle than C/C++. The MicroPython or CircuitPython board provides an onboard file system that appears as a USB flash drive. You can edit code on the board directly from the host computer. The main program—main.py (MicroPython) or code.py (CircuitPython)—executes when the board starts and contains a loop that runs forever, analogous to the `loop()` function in an Arduino program. You can store additional code files as Python source (.py) files or as precompiled bytecodes (.mpy) and data files in whatever format you like.

In addition to the disk drive, a serial USB connects to the Python interactive `read-eval-print loop` (REPL), so you can type Python code at the REPL prompt for testing. The onboard program can also print and read to and from that serial connection.

Being an interpreted language, Python runs code instantaneously: There's no compile, link, convert, bootload, upload, like in C. Instead, you edit code directly on the device, and it executes the moment you save changes. When you're working on projects that require a lot of calibration and tweaking, Python can save you hours.

Adafruit recommends the Mu editor (https://codewith.mu) for beginners. It contains a simple Python-aware editor and a serial connection to the MicroPython or CircuitPython board. It automatically detects the device when plugged in, has a REPL window, and even comes with a built-in plotter.

Does MicroPython/CircuitPython support direct hardware I/O access? What kind of TCP/IP networking support does it typically support? What about Bluetooth?

Both MicroPython and CircuitPython include natively implemented Python modules that enable direct access to on-chip peripherals, such as general-purpose I/O pins and analog conversion. In MicroPython, these modules can vary between ports. In CircuitPython, Adafruit has tried to make these libraries as uniform as possible across microcontrollers and microcomputers. For example, you can use our more than 100 CircuitPython drivers and libraries with Raspberry Pi or BeagleBone Linux running classic Python 3.

MicroPython/CircuitPython provide network stack access to boards that have Wi-Fi and Ethernet. There are MicroPython ports to both the ESP8266 and ESP32 models. CircuitPython has been ported to ESP8266. Simple wired Ethernet support was recently added to CircuitPython as well. MicroPython can use Bluetooth Low Energy or other radio-based protocols on boards that support them. CircuitPython is currently being ported to the nRF528xx series of microcontrollers; an alpha version of Bluetooth Low Energy is currently available.

What range of microcontrollers and embedded processors can run MicroPython/CircuitPython? What are typical RAM and flash minimum sizing requirements?

CircuitPython has ports to the Microchip/Atmel SAMD21 (M0) and SAMD51 (M4) microcontrollers and to the ESP8266 module. SAMD21 chips run at 48 MHz, with 32KB of RAM and 256KB of on-chip flash memory—the minimum for getting started with basic projects. SAMD51 chips are much faster, with a 120-MHz clock, up to 256KB of RAM, and 1MB of flash storage. A port to the nRF52832 and nRF52840 microcontrollers is underway and available in alpha versions. Because Python is interpreted and has a garbage collector, plenty of RAM is handy (we recommend 64KB or more).

Adafruit's "Express" boards, which are designed for CircuitPython, also provide an onboard external SPI flash chip, typically 2 to 8MB, used as the file system for storing Python code and other files. MicroPython is not ported to SAMDx1 chips but includes many ports for STM32, ESP32, nRF51, nRF52, Kinetis, and several other processors.

Does MicroPython/CircuitPython have any special features for supporting embedded development, such as multithreading, memory management, and the ability to differentiate read-only memory and read/write memory?

Many issues that come up in embedded programming are already solved in Python. Take memory management. Humans and pointers don't mix very well, but in Python, you never have to worry about accidentally overwriting memory because all objects are checked during all instance accesses.

Also, many embedded engineers struggle to perform full error checking at all steps. In C, returns checking is particularly clunky and easy to mess up. In Python, exceptions are built in, so you can halt at any failure. If you make a "divide by zero" or a memory access mistake, your code doesn't completely halt. Instead, it will bump out an exception that you can display to the user, communicate up the control queue, or handle immediately. Adafruit implements locking for peripherals, so you don't have to worry about multiple objects talking to the SPI port, for example.

MicroPython has optional threading and interrupt callback support as well. In CircuitPython these features are not currently supported, but Adafruit is examining ways to provide concurrency and asynchronous execution that is simple to use and understand. Currently, we implement those operations that require concurrency and asynchronous handling in the native CircuitPython libraries.

We have built some nice features into CircuitPython that handle concurrency for you. Obviously, the USB parts are completely handled for you, separate from the runtime code. Timers are dynamically allocated to the first one available. Audio playback is built in: You can play .wav files right off the file system, completely asynchronously.

[CONT'D ON NEXT PAGE]

We also added direct memory access for all underlying peripheral transactions—with no effort required.

Does MicroPython/CircuitPython follow Python 2 or Python 3 syntax? What are the implications for embedded development?

MicroPython and CircuitPython are up to date with Python 3.4. Support for Python 2 is not a good idea because that version has been deprecated, and all new development is in Python 3 only.

Can I use any of the standard Python packages found in the Python Package Index (PyPI)? If not, what packages are typically compatible with MicroPython/CircuitPython? How do I know which packages will work with MicroPython/CircuitPython?

Packages that depend on the built-in libraries and are not optimized may not fit in memory for the smaller microcontrollers. Many PyPI libraries are too large or not written with limited storage in mind. Some smaller libraries and a lot of example code will work just fine (just, not NumPy, for example).

For chips, actuators, and sensors, Adafruit has written well-supported and easy-to-use libraries that support various hardware devices available on PyPI, such as I^2C sensors and LED color management. Adafruit also wrote a wrapper library (`adafruit-blinka`) that enables those libraries to run using native, regular Python on the Raspberry Pi and other boards that support access to chip peripherals. More than 110 CircuitPython drivers and libraries are available, and we are adding more all the time. Just search for “adafruit-circuitpython” on pypi.org.

Are there any runtime royalties or licensing considerations for MicroPython/CircuitPython, particularly with regard to using it in product design?

Both MicroPython and CircuitPython use the permissive MIT open source license, which allows free use for any purpose as long as copyright notices and original license text are included in derived products. There are no license fees for production use.

For a product application, can my Python source code be read back out of the embedded hardware? If yes, are there ways to protect my code so that no one else can access it?

Yes. The Python source is stored in the onboard file system and can be accessed again. You can store the code as a compiled .mpy file (similar to a .pyc file), which stores bytecodes. This format obfuscates the code but doesn’t hide it.

It’s possible to further restrict the source to the onboard file system, but we have not implemented such a feature. At this time, MicroPython/CircuitPython would not protect code against advanced hardware snooping. In contrast, most microcontroller chips can have firmware read out of them, and many devices are moving toward embedded Linux with Python, where a whole file system is available. If you have keys that you want to keep secret, Adafruit recommends a security chip to handle that encryption and privacy for you.

Amphenol® RF
Global RF Solutions.

Amphenol RF HD-EFI Series Micro-Miniature Interface Connectors

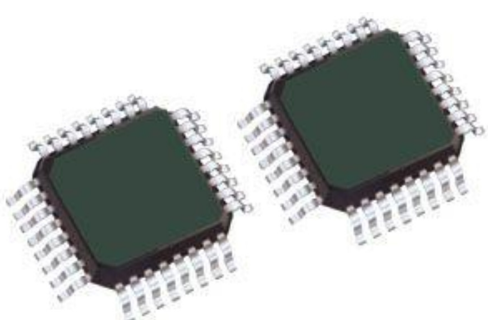
mouser.com/amphenol-rf-hd-efi-connectors

SILICON LABS

Silicon Labs LTE-M Expansion Kit

mouser.com/silicon-labs-xbee3-lte-m-expansion-kit

life.augmented



STMicroelectronics STM32G071x8/xB 32-Bit Microcontrollers (MCUs)

mouser.com/stm-stm32go-microcontrollers

NORDIC®
SEMICONDUCTOR

Nordic Semiconductor nRF9160 Development Kit

mouser.com/nordic-semiconductor-nrf9160-kit

IoT Development Kits

by Greg Toth – Founder and CEO, IoT Development Labs

The rapid growth of the Internet of Things applications has brought with it a growing number of development kits that provide new tools for design engineers who are working on Internet of Things products and systems. Selecting the right development kit can be a challenge.

IoT Dev Kits at a Glance

Design engineers can rapidly develop new Internet of Things (IoT) products and systems through the new tools found in IoT development kits (dev kits), which are sets of hardware, software, and firmware components that enable engineers to prototype and test their IoT applications rapidly. These kits resemble typical microcontroller development and evaluation boards, but they focus more on the sensors, actuators, and communications interfaces that we frequently find in IoT use cases.

Developers use IoT dev kits to create applications that access cloud computing services where sensor data processing and storing occurs. They can also use IoT dev kits to create stand-alone IoT applications. Some kits feature expansion connectors that allow developers to add new modules in a mix-and-match fashion. Although IoT dev kits typically support software and firmware developments in C/C++ programming languages, some of them work with other languages, such as Python or JavaScript.

Key Facts

- Gartner predicts that spending on IoT services will reach nearly $3 trillion by 2020.
- An IoT dev kit typically falls into one of three categories: Cloud-connection, local network, or gateway (a system that connects legacy applications to the Internet).
- IoT dev kits often include components such as breadboards, jumper wires, expansion boards, power supplies, batteries, sensors, and actuators.

Mouser Manufacturers Leading the Way

- ON Semiconductor recently released a modular platform, the Bluetooth IoT dev kit, for developing and prototyping Bluetooth Low Energy (BLE) IoT applications. At the 2018 electronica conference, ON presented IoT demonstrations of battery-less and wireless edge nodes implemented with BLE.
- Seeed Studio created the Eagleye 530s Dev Kit for the Samsung ARTIK IoT platform to enable developers to prototype and build with off-the-shelf sensors, voice recognition, relays, a Global Positioning System (GPS), cellular connectivity, and more.
- Cypress Semiconductor recently announced two new dev kits for the PSoC 6 family of devices that provide secure, low-power processing to a broad range of IoT applications at the edge.

See Also:

- Article: “Creating Quick Connections with IoT Development Kits”
- Article: “World Wide What? An Honest Look at the State of the Internet of Things”

“Many IoT dev kits offer cloud connectivity, either to a Microsoft, Google, Amazon, or IBM general-purpose cloud platform or to an IoT-specific cloud platform such as Medium One or Particle.”

[CONT’D ON NEXT PAGE]

The rapid growth of the Internet of Things (IoT) applications has introduced a growing number of IoT dev kits that provide new tools for design engineers working on IoT products and systems.

What Is an IoT Dev Kit?

An IoT dev kit is a set of hardware and software or firmware components that help design engineers create, test, and evaluate new IoT designs using a combination of microcontrollers, sensors, actuators, and comm-unications interfaces. These kits are an evolution of the development/evaluation board concept, which has long been used to test and esvaluate microcontrollers and peripherals.

The main difference that IoT dev kits make is creating an increased focus on sensors (e.g., temperature and motion), actuators (e.g., light-emitting diodes [LEDs] and displays), communications (e.g., Wi-Fi, Bluetooth, and Long-Term Evolution [LTE]), specialty processors (e.g., secure elements and artificial intelligence [AI] engines), and firmware for secure connections to cloud computing environments.

Some IoT dev kits have a fixed set of components. Others have modular designs that enable you to add components using Arduino shields, vendor-specific expansion buses, or headers, which provide access to inter-integrated circuit (I2C), serial peripheral interface (SPI), universal asynchronous receiver and transmitter (UART), analog-to-digital converter (ADC), digital-to-analog converter (DAC), general-purpose input/output (GPIO), and other signals.

In addition to hardware components, many IoT dev kits include software/firmware libraries to control peripherals and provide different communication protocols, including Transmission Control Protocol/Internet Protocol (TCP/IP) networking and wireless communication protocols. IoT dev kits are typically used during the prototyping and proof-of-concept phase of the innovation-to-productization life cycle and serve as educational tools to learn about the IoT. When used during product development, these kits are typically stepping-stones to customize and optimize an electronic design for mass production. They are generally available as off-the-shelf items that vary in cost depending on the functionality and components included in each kit.

Technical Domains

IoT applications typically involve multiple domains, including hardware, sensing, signal processing, communication, security, and data analytics. As a result, multiple engineering skills are necessary. IoT dev kits strive to accelerate usage by providing end-to-end sample applications along with documentation and tutorials. Firmware programming is typically done in C/C++ integrated development environments (IDEs), but

some kits support other languages, such as Python and JavaScript. Some suppliers offer free IDEs for their kits; others use licensed development tools that you must purchase separately.

Selecting an IoT Dev Kit

IoT dev kits vary by supplier, and the primary differences involve the microcontrollers; sensors, actuators, and specialty processors; communications interfaces; available software/firmware libraries; and the supported programming languages. IoT dev kit choices are guided by an evaluation of kit features and functionalities that will most effectively support the IoT application you want to develop. For example, if you're developing a smart home device with a Wi-Fi and smartphone (via Bluetooth) connection, an IoT dev kit that supports Wi-Fi and BLE communications is a priority. Other selection criteria may be the number and variety of software/firmware libraries available to support your application's development: For example, real-time operating systems (RTOS) libraries, secure communications libraries (e.g., for Hypertext Transfer Protocol Secure [HTTPS] or Message Queuing Telemetry Transport [MQTT]), device control libraries (e.g., for sensor input/output [I/O]), wireless communication libraries (e.g., for Wi-Fi, BLE, and Zigbee), and libraries for connecting to different cloud computing systems. These libraries may be provided in source code or binary form and may be open source, closed source, or licensed by the supplier.

An emerging area in IoT design is the inclusion of AI processing near the sensors, either directly at the sensor or in a nearby IoT gateway. This inclusion enables faster and more complex local data processing, instead of sending large volumes of raw sensor data over the Internet to a cloud computing back end. Some IoT dev kits directly support AI applications by including specialized AI processing chips.

Cloud Connectivity

Many IoT dev kits offer cloud connectivity, either to a Microsoft, Google, Amazon, or IBM general-purpose cloud platform or to an IoT-specific cloud platform such as Medium One or Particle. Many of these cloud platforms offer low- or no-cost ways to connect a small number of devices.

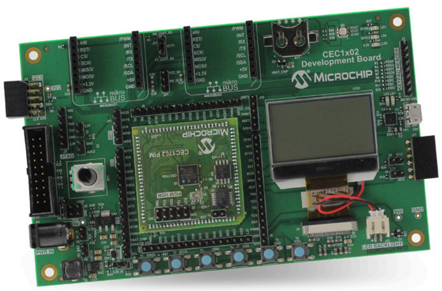
Example IoT Dev Kits

Many IoT dev kits are available. On the next two pages you'll find six that represent what's currently available and the functionality they include.

"IoT dev kits are typically used during the prototyping and proof-of-concept phase of the innovation-to-productization life cycle and serve as educational tools you can use to learn about the IoT."

Example IoT Dev Kits

Microchip Technology CEC1x02 Development Board

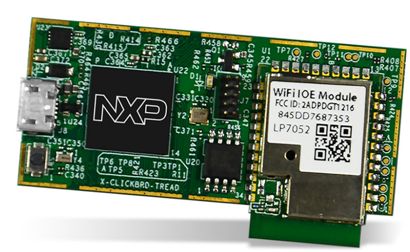


The CEC1x02 Development Board is a development, demonstration, and testing platform using the CEC1702 cryptographic embedded controller, a 32-bit Arm® Cortex®-M4-based microcontroller with integrated crypto-accelerators. The board features a variety of hardware options (including a graphic LCD display, headers for ADC, I²C, GPIO, and two mikroBUS™ interfaces) that enable rapid prototyping and development of embedded, secure Internet-of-Things applications. The platform is also programmable in C/C++ and has SDKs for Microsoft Azure and Amazon Web Services

mouser.com/microchip-techn-dm990013-dev-board

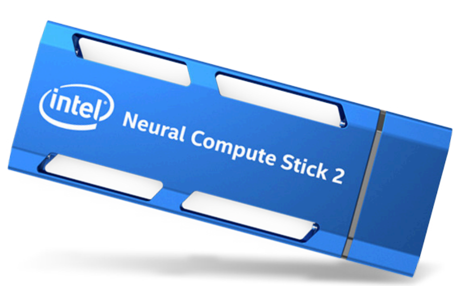
NXP Semiconductor LPC54018 IoT Module

The NXP LPC54018 IoT Module contains an Arm® Cortex®-M4 microcontroller unit (MCU), a Wi-Fi communications module, a high-speed USB port, and expansion connectors that enable it to connect to an NXP OM40006 baseboard. The baseboard contains additional peripherals including an LCD with touchscreen capabilities, LEDs and push buttons, an audio codec, microphone, accelerometer, Ethernet, microSD card slot, Arduino UNO expansion connectors, and additional random access memory (RAM). The baseboard is also programmable in C/C++ and supports the Amazon FreeRTOS operating system.



mouser.com/nxp-lpc54018-iot-module

Intel® Neural Compute Stick 2



The Intel Neural Compute Stick 2 kit is a Universal Serial Bus (USB) stick that contains a central processing unit (CPU) cluster coupled with a neural compute engine—a dedicated hardware accelerator for deep neural network applications commonly used in computer vision use cases. You can use any platform with a USB port to create prototypes and operate AI applications without dependence on cloud computing. It is programmable in C/C++ or Python and comes with a software development kit (SDK) and AI frameworks for developing applications.

mouser.com/intel-neural-compute-stick-2

Cypress Semiconductor PSoC® 6 WiFi-BT Pioneer Kit

The Cypress PSoC WiFi-BT Pioneer Kit contains an Arm® Cortex®-M4 MCU, an Arm Cortex-M0+ MCU, flash and RAM memory, USB interfaces, a graphic liquid crystal display (LCD), six-axis motion sensor, light sensor, digital microphone, audio codec, CapSense® buttons and a slider, LEDs and push buttons, a Wi-Fi and BLE wireless module, and Arduino-compatible expansion connectors. It is programmable in C/C++ using the Cypress WICED® (short for Wireless Internet Connectivity for Embedded Devices) IDE environment, which supports Wi-Fi and BLE communications and connectivity to cloud platforms including Microsoft Azure and Amazon Web Services.

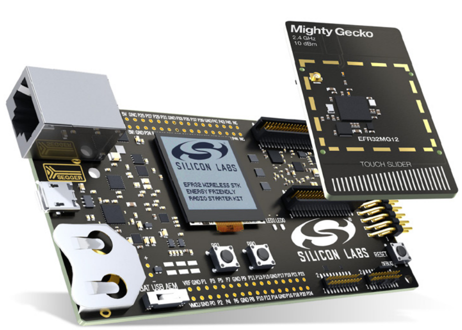
mouser.com/cypress-psoc6-wifi-bt-pioneer-kit

Renesas Electronics Synergy™ AE-CLOUD2 LTE IoT Connectivity Kit

The Renesas Electronics Synergy™ AE-CLOUD2 LTE IoT Connectivity Kit contains an Arm® Cortex®-M4 MCU; flash and RAM memory; a USB interface; peripheral module (PMOD™) and Grove connectors; LEDs and push buttons; an Ethernet interface; a Wi-Fi module; an LTE Cat-M1, Cat-NB1, and enhanced general packet radio service (EGPRS) modem module with GPS; a six-axis motion sensor; a geomagnetic sensor; a gas, pressure, temperature, and humidity sensor; a light sensor; a microphone; and Arduino-compatible expansion connectors. It's programmable in C/C++ and has software libraries to connect to multiple cloud environments including Microsoft Azure, Google Cloud Platform, Amazon Web Services, and the Medium One IoT Prototyping Sandbox.

mouser.com/renesas-synergy-ae-cloud2

Silicon Labs Mighty Gecko Mesh Development Kit



The Silicon Labs Mighty Gecko Mesh Development Kit contains an Arm® Cortex®-M4 MCU, flash and RAM memory, a USB interface, an Ethernet interface, LEDs and push buttons, temperature and humidity sensors, an LCD graphics display, a capacitive touch slider, a coin cell battery holder, 2.4GHz wireless modules that can run multiple mesh network protocols, and an expansion header to connect additional peripheral devices. Multiple boards come with the device to develop wireless mesh network applications. The Mighty Gecko is programmable in C/C++ and has software libraries for multiple radio protocols including Zigbee, Thread, and *Bluetooth*.

mouser.com/silabs-mighty-gecko-mesh-devkit

The IoT and Sustainable Engineering

by Knud Lueth – Founder and CEO, IoT Analytics

Engineers will play a key role in achieving sustainability goals established by the United Nations. Internet of Things solutions can contribute enormously to sustainability because they make new, more sustainable business models possible. How should engineers practice "sustainable engineering?"

Sustainable Engineering at a Glance

Many Internet of Things (IoT) devices designed for industrial automation and control help to streamline processes so they run faster, more reliably, and with lower operational overhead. As it turns out, these same IoT systems are making a significant contribution to environmental sustainability. For example, they enable on-demand power consumption decisions that increase the efficiency of heating, ventilation, and air conditioning; lighting; and power-intensive industrial processes. In that way, they significantly reduce the carbon footprint of those systems. Countries are implementing "smart cities" programs that use controllers and IoT devices to better manage energy and water consumption. As IoT systems become smarter and more widely adapted to optimizing physical infrastructure, including utilities, buildings, and transportation, many see these devices and systems as keys to future sustainability.

Key Facts

- Research by the World Economic Forum finds that many executives do not see a direct link between their IoT projects and sustainable developments, even though research shows that 84 percent of IoT deployments—70 percent of them driven by the private sector—are addressing sustainable development goals.
- Connected Cities USA says that cities consume about 70 percent of the global energy. By 2030, about 60 percent of the world's population will live in cities.
- Connected Cities USA also says that 70 percent of energy consumption in major cities comes from buildings, and buildings are the source of 30 percent of all greenhouse gas emissions globally.

Mouser Manufacturers Leading the Way

- In November 2018, Texas Instruments released new mmWave technology for the global industrial market. The IWR1642 mmWave sensors enable industrial automation through on-chip processing capabilities, providing real-time decision-making and signal processing in industrial settings.
- Maxim Integrated is showcasing new high-performance analog-integrated circuits for building automation.
- In November 2018, Analog Devices announced a broad range of solutions aimed at helping manufacturers of original industrial equipment accelerate their path to Industry 4.0.

See Also:

- Applications: Solar Power
- Article: "New PV Cells Benefit Energy Harvesting"
- Article: "Solar Energy Harvesting Project to Power a Remote MSP430 with 2.4GHz Notification"

"Sustainable components are important, but engineers also need to consider the larger context of their designs."

[CONT'D ON NEXT PAGE]

Many blame the technological innovations of the past 150 years for the sustainability challenges we face today, but it is just as true that technology has a key role in building a more sustainable world. To understand why this is, you need to understand exactly what is meant by sustainability.

Sustainability is the balance among four elements: Resource exploitation, investment direction, technological development, and institutional change. All four must be in harmony while they enhance both current and future human needs. In 2015, the United Nations (UN) released 17 Sustainable Development Goals (SDGs), which provide an excellent starting point for sustainability. They include high-level goals such as "no poverty," "zero hunger," and "affordable and clean energy," which are then further broken down into more detailed objectives such as "creating new agricultural practices" and "improving the global rate of energy efficiency."

Sustainable Engineering

Engineers play an important role in achieving these goals through "sustainable engineering," which includes methods such as life-cycle analysis, pollution prevention, Design for the Environment (DfE), design for disassembly, and design for recycling. The question for designers and engineers is not how and where they fit in a sustainable design process but whether the complete process has been considered with sustainability in mind.

One area where engineers are having a big impact now is through connected devices and the Internet of Things (IoT). Research shows that IoT technology is contributing tremendously toward many sustainability goals, mostly through new business models and use cases enabled by the IoT. We analyzed more than 600 IoT deployments, ranging from smart factories to smart agriculture and smart grids. The result: Eighty-four percent of existing IoT deployments can address the 17 SDGs that the UN has laid out.

Why is the IoT having such a big impact? At its core, the IoT is about measuring and remotely controlling previously unconnected "things." It reaches people and objects that older technology could not. However, when you look at some of the key use cases, you can see how important the IoT is for our sustainable future. Some of the most promising cases focus on IoT in smart cities and agriculture.

Many cities are currently installing IoT-enabled sensors that continuously collect and transmit environmental data such as air quality, carbon dioxide (CO_2) levels, noise, and other information—a crucial step for taking the right actions to make cities cleaner and quieter. Another use case is smart parking. New IoT-enabled smart parking solutions help people find appropriate parking spaces more quickly, reducing road congestion while at the same time reducing CO_2 emissions.

In agriculture, smart irrigation systems save a lot of water. IoT sensors and controllers not only facilitate exact monitoring of water usage but also enable remote shutoffs and more accurate monitoring of where and how much water is actually needed.

Smart water is now a focal point, not only for irrigation but also for water consumption monitoring and usage by people around the world. I spent part of last summer in Cape Town, South Africa, where I witnessed firsthand frustration, as the city almost ran out of water while having limited insight into who the biggest water users were. IoT solutions such as automated water metering will have a positive effect on such issues, creating sustainability.

Quantifying Sustainability

One of the challenges facing engineers who build sustainability into their design and engineering process is the potential negative sustainability impacts of IoT technology. Every sensor and controller we deploy consumes resources, including raw materials and energy. With billions of new devices expected to hit the market, the quantity will be significant. As such, part of the design process will involve determining the net negative or positive impact of an engineered solution, which raises another engineering challenge: How do you quantify sustainability?

Measuring sustainability is not easy. First, you must determine whether the solution at hand improves any of the SDGs the UN has laid out and how the improvements are expected to unfold. This analysis should also include the scale of the solution's deployment. Several standard bodies provide frameworks for measuring sustainability, including the Global Reporting Initiative, the International Organization for Standardization, and the CDP (originally named the Carbon Disclosure Project, this non-profit now works with industries to study climate impact). These frameworks mainly target global corporate activity and general business models and are of limited value to engineers working on specific engineering problems and who want clear, actionable guidelines they can apply to their product designs.

Still, engineers can dig into these frameworks for more specific guidance. For example, the DfE framework includes a concept called design for energy efficiency, which is particularly relevant for today's engineers. It looks at ways to design a product to minimize overall energy consumption throughout the product's life. With the IoT, we now have communication methods such as low-power, wide-area wireless technologies, which allow for much lower power consumption and battery usage than traditional cellular networks. There are also strategies such as energy harvesting through small solar cells or similar technologies that can completely change the energy efficiency equation of a device or solution.

"The question for designers and engineers is not how and where they fit in a sustainable design process but whether the complete process has been considered with sustainability in mind."

Engineers should also consider the components they use in their IoT-device designs. Some components are clearly more sustainable than others. For example, components that contribute to sustainability include:

- Anything that uses energy harvesting
- Components that use low-powered communication technologies
- Components that use edge computing to filter and process data locally rather than consuming power to transmit data to a power-hungry cloud computing resource
- Components that avoid hazardous substances by incorporating sustainable materials (e.g., using conductive adhesives or other lead substitutes rather than soldering a component assembly)

Using sustainable components is not always possible, though. Sometimes, it is not even the solution's biggest contribution to sustainability. Sustainable components are important, but engineers also need to consider the larger context of their designs. IoT solutions, for example, can contribute enormously to sustainability because they make new, more sustainable business models possible.

Conclusion

For engineers interested in learning more about using IoT devices in sustainable design and how to become involved in sustainable design initiatives, many resources are available. At the corporate level, the World Economic Forum has several initiatives related to the IoT and sustainability. We work together with them to help firms around the world lead the discussion on how they can get started.

Hands-on engineers may be more interested in the working groups that engineering organizations offer. For instance, the Institute of Electrical and Electronics Engineers has an initiative called Sustainable ICT. The German engineering association VDI has a guideline called Nachhaltiges Wirtschaften, and the American Society of Mechanical Engineers has a website dedicated to sustainability (https://www.asme.org/engineering-topics/sustainability). In addition, many universities, including Purdue and Georgia Tech, have research teams devoted to the topic of sustainable engineering.

Now is a great time to further sustainability. The emergence of smart IoT devices is creating much-needed opportunities for building a more sustainable world.

Component Design: What Comes After Moore's Law?

by Mouser Editorial Staff, based on an interview with Andrew "bunnie" Huang, conducted on November 19, 2018

Moore's law, which has predicted the rate of technological advancement for the past 50 years, is coming to an end. Engineers now have the luxury of taking the time to optimize, which should result in more efficient systems at a given process node. What does this mean for component design?

Moore's Law at a Glance

Since the 1970s, Moore's law has governed the pace of technological innovation. Until now, every two years chipmakers could cram double the number of transistors into a single chip as the transistors shrank in size. As they did, engineers and end users were able to take advantage of rapid performance improvements on a predictable schedule. Moore's law has begun to decelerate, however, making it more time-consuming and costly for chip manufacturers to achieve the same performance gains within this historically expected time frame. As a result, engineers at small businesses now have intriguing opportunities to innovate their products in a way that they could not before. Industry players may soon welcome a new era of open hardware initiatives, including the creation of stable, performance-competitive open platforms. This article looks at some of the exciting possibilities that may arise as Moore's law slows its pace.

Key Facts

- For 50 years, Moore's law predicted that the number of transistors that could be crammed onto a chip would double every two years.
- Design and development costs are constraining factors: As geometries shrink below 22nm, we start to see costs rise faster than features shrink.
- Moore's law really tapped the brakes as semiconductor processes passed the 14nm node.
- System integrators are now encouraged to improve their existing product designs, rather than simply waiting for the next generation of chips to arrive.

Mouser Manufacturers Leading the Way

- Intel is exploring a new design concept called chiplets, which are modular pieces of silicon that can be assembled like LEGO bricks.
- In January 2018, Intel announced a processor for mobile personal computers (PCs) that combines an Intel central processing unit (CPU) with a custom graphics module from AMD (formally called Advanced Micro Devices), marking the first time Intel has included a core from another company in its main processor line.

See Also:

- Articles and More: Memory and Storage

"For 50 years, Moore's law predicted that the number of transistors that could be crammed onto a chip would double every two years."

[CONT'D ON NEXT PAGE]

For the past 50 years, Moore's law has successfully predicted that the number of transistors per unit area on a chip would double every two years. This growth was possible because the cost per transistor went down with every new generation of silicon. However, once we reached transistor sizes of 28 nanometers (nm)—and particularly after we crossed the 20nm threshold into fin field-effect transistor (FinFET) territory—this once-standard rate of technological progress began decelerating as lithographic light sources failed to keep pace. This trend forced manufacturers into a Faustian bargain between adding more process complexity using traditional light sources or decreasing throughput with new light sources. Either way, even though manufacturers can still technically produce chips at even finer line widths, it's less economically viable to do so because the cost per transistor is starting to rise.

"Moore's law really tapped the brakes as semiconductor processes passed the 14nm node."

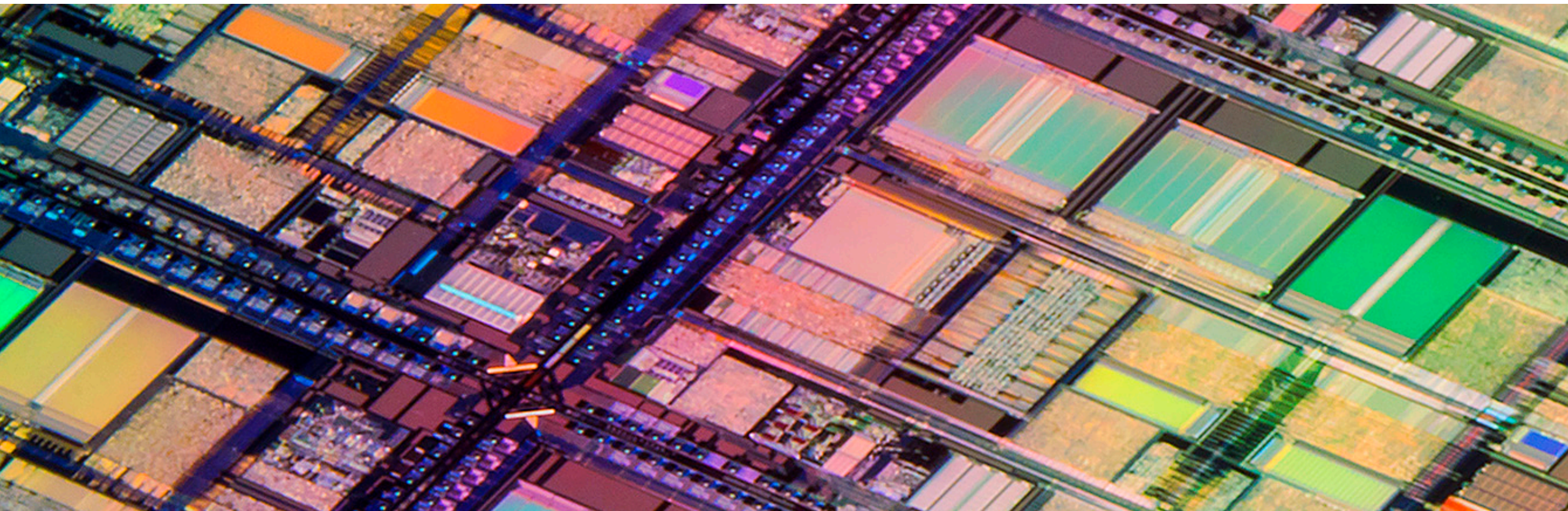
As Moore's law begins to slow down, the electronics industry must consider new technologies and novel approaches to systems design.

Systems integrators now have the luxury of time to optimize and refine architectures. For the first time in quite a while, they are incentivized to move to the latest generation process only if they require enhanced performance or greater transistor density (which comes at a higher price instead of getting both for less). As a corollary, value-conscious market segments will continue to use older processes while looking for new ways to accommodate a lower performance in order to hit the price targets that the market will bear.

It used to be that if it took more than two years to optimize a system by customizing an interconnect fabric, parallelizing the code, and ultimately building a supercomputer around a given process node, that the project would be considered dead on arrival. The reason is that by the time it was finished, a new computer would already be on the market that would run twice as fast and cost about the same. But now that the succession of process nodes on a two-year timescale is no longer a given, system-level designers and system integrators looking to deliver more value must pay closer attention to factors like bandwidth limits, data locality, core utilization, and power efficiency. As a result, optimization will become a more common refrain among engineers, which may ultimately result in more efficient systems for a given process node.

For example, intriguing application-specific architectures (ASICs), such as the Tensor Processing Unit (TPU) that Google and its artificial intelligence (AI) partners have created to accelerate machine learning workloads, are starting to appear. To service machine learning computational loads, engineers have spent time and effort turning out architectures with massive arrays of 8-bit multipliers and adders that can calculate with less precision than a graphics processing unit (GPU) but that are perfectly suited for machine learning applications. Until recently, by the time you finished such a project, regular computers running at twice the speed would have already outstripped all the advantage of a custom-built ASIC. Now, in the post-Moore's law world, it has become practical—perhaps even imperative—to build ASICs that achieve far greater efficiency and performance.

Greater design optimization is also evident in the mobile space, which is extremely power sensitive. You'll find that algorithms implemented in software in the CPU are inevitably going to cost much more in terms of power than algorithms reduced directly into silicon gates. In a Moore's law regime, it would not be worth your while to cast an algorithm into silicon,



because first, algorithms would constantly evolve to take advantage of faster and greater numbers of transistors, then the standards would change, and then a faster general-purpose CPU would come along and blow your custom silicon away. All your investment would have been wasted. However, in a post-Moore world, as speed and density increases level off, it's less risky to cast malleable software algorithms into silicon and save some power.

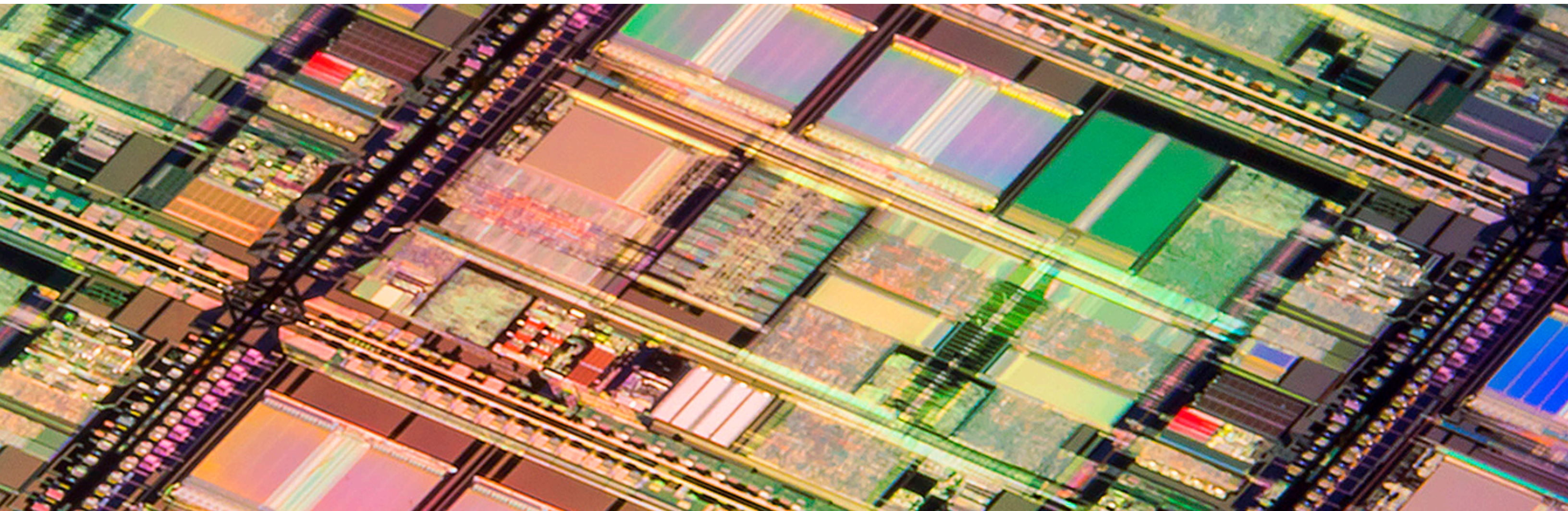
Software developers are finding greater incentives to optimize as well. In the late 1990s and early 2000s it was okay—nay, expected—to pack so many new features into a software release that it could barely run on average machines because within a couple years Moore's law would take care of any performance deficiencies. Today, this sort of practice would not be bearable. Instead, now engineers can spend months to refine performance without worrying too much about being obsoleted by a legacy codebase running on a new CPU, with significantly faster single-threaded performance.

Open hardware and boutique hardware engineering will enjoy increased prominence, too. Open source projects typically take several years to take off and reach maturity. Unfortunately, in a Moore's law world, only hobby projects can afford to take three to four years to take off. Now it's conceivable that after spending three or four years implementing something in a field-programmable gate array (FPGA), you may find that you might actually have the world's best-performing product for that specific problem domain. We're not quite at that stage for general problems, because Moore's law is still moving a little bit upward, but this could become a reasonable general expectation if Moore's law continues to slow.

The design tool and chip-building ecosystems will also have reasons to become more open in this new world. Historically, there has been a real economic benefit from timely access to the latest technologies, so foundries typically have not made their latest processes available to everybody, just to those who could pay the most for it. However, as the shine wears off of a new process node and replacement nodes seem farther and farther away, the greatest economic benefit to a foundry will shift from charging a premium just to get into the door to opening the doors to get more customers into the shop. Thus, once the initial investment of building a fab depreciates, the market logic can shift toward a more open model, where the non-recurring engineering (NRE) costs are more reflective of the mere time and materials required to set up a fab run, rather than the amortization of a multibillion-dollar investment.

In the future, we may just have a big tent under which everyone can come together, including more designers who are making more application-specific circuits. No longer will you have to go to a big supplier for your analog components. Instead, you might be able to simply pay someone a few thousand dollars for a design, then pay a reasonable price for a spin of a few wafers, to have exactly the chip you require for your application. I am looking forward to seeing what novel sensors might crop up as access to ultra-low power and dense transistors become increasingly affordable.

The end of Moore's law points to the creation of a new regime, with "brand-new" market incentives influencing the rate of technological advancement. This new regime is altogether different from how the electronics industry has been operating for the past several decades. It is a disruptive development (to be sure) but one that offers intriguing possibilities for the future. M



Authors

Waleed Ejaz

Waleed Ejaz is an assistant professor at Thompson Rivers University in Kamloops, British Columbia, Canada. He received his PhD in engineering from Sejong University in the Republic of Korea in 2014. His current research interests include the Internet of Things, energy harvesting, 5G cellular networks, and mobile cloud computing. His is an associate editor of *IEEE Communications Magazine*, the *Canadian Journal of Electrical and Computer Engineering*, and *IEEE Access*. He is a registered professional engineer in British Columbia.

Limor “Ladyada” Fried

Massachusetts Institute of Technology-educated hacker and engineer Limor “Ladyada” Fried founded Adafruit in 2005 to be the best place online for learning electronics and designing products for makers of all ages and skill levels. Adafruit has grown to more than 100 employees in a 50,000-square-foot factory in New York City, New York. It has expanded to include tools, equipment, and electronics that Limor personally selects, tests, and approves before they go into the Adafruit store. Adafruit is a 100 percent woman-owned company.

Sam George

Sam George is the director of the Microsoft Azure IoT engineering team in program management, delivering a broad portfolio of features and capabilities that help customers and partners realize the full potential of the Internet of Things. He is responsible for such solutions as Microsoft IoT Central, Azure IoT Suite, Azure IoT Edge, Azure IoT Hub, Azure IoT Device Provisioning Service, our IoT Device SDK, the Microsoft Connected Vehicle Platform, and more. An accomplished industry leader, Sam’s contributions during his 18-year tenure at Microsoft involve impacting various technologies including video, smartphone, PC, and cloud services.

Andrew “bunnie” Huang

Andrew “bunnie” Huang is best known for his work in hacking Xbox and for his efforts designing and manufacturing open source hardware including chumby (an app-playing alarm clock), Chibitronics (peel-and-stick electronics for crafts), and Novena (a DIY laptop). He received his PhD in electrical engineering from the Massachusetts Institute of Technology (MIT) in 2002. Currently, bunnie lives in Singapore, where he runs a private product design studio called Kosagi and actively mentors several startups and students of the MIT Media Lab.

Sudha Jamthe

Sudha Jamthe is CEO IoTDisruptions.com and DriverlessWorldSchool.com, as well as a recognized technology futurist. She has more than 20 years of experience with companies like eBay, PayPal, and GTE. Sudha is also the author of 2030: The Driverless World and three books on the Internet of Things. She teaches courses about the business of IoT and autonomous vehicles at Stanford Continuing Studies and DriverlessWorldSchool.com, is chair of the strategic advisory board for Barcelona Technology School, and is an ambassador for FundingBox Impact Connected Cars.

M. Tim Jones

M. Tim Jones is a veteran embedded firmware architect with over 30 years of architecture and development experience. Mr. Jones is also the author of several books and many articles across the spectrum of software and firmware development. His engineering background ranges from the development of kernels for geosynchronous spacecraft to embedded systems architecture and protocol development.

Knud Lueth

Knud Lasse Lueth is the founder and CEO of IoT Analytics, a leading market research firm focused on the Internet of Things (IoT) and Industry 4.0. For the last 10+ years, he has been working with numerous IoT software and hardware companies as well as technology end-users. His primary focus is on helping firms identify the next big trends, and then turning these insights into actionable strategies. Among other things, Knud is a co-author of the "Internet of Things Guidelines for Sustainability," published by the World Economic Forum in January 2018, and prior to founding IoT Analytics, Knud worked at the Boston Consulting Group, BMW, and Lufthansa Technik.

Bob Martin

A maker before it was a term, Bob Martin has been tearing apart things to see how they work all his life. He has done everything from installing specialized instruments at arctic weather stations to supporting atmospheric research campaigns. Now he serves as the "Wizard of Make" for Microchip where he keeps the maker spirit alive within Microchip; educates aspiring makers; and continues to create, experiment and explore. He lives in Sunnyvale, California, with his wife, twin daughters, a cat with no teeth, and a garage full of surplus wire and "vintage electronics" that really should be recycled.

Jason Shepherd

Jason Shepherd is the Dell Technologies IoT and Edge Computing CTO, leading a team responsible for market and technology strategy, solution planning, and strategic ecosystem development within the Dell Technologies IoT and Edge Computing Solutions Division. As a thought leader in the IoT market, Jason has successfully built-up the award-winning Dell IoT Solutions Partner Program and established the vendor-neutral, open source EdgeX Foundry project to facilitate greater interoperability at the IoT edge. He was recognized as one of the Top 100 Industrial IoT influencers of 2018. He also holds 14 granted and 11 pending US patents.

Greg Toth

Greg Toth is an architect, engineer, and consultant with more than 30 years of experience in sensors; embedded systems; the Internet of Things; telecommunications; enterprise systems; cloud computing; data analytics; and hardware, software, and firmware development. He has a BS degree in electrical engineering from the University of Notre Dame and an MS in computer engineering from the University of Southern California.

Darryl Wright

Darryl Wright is a research engineer at Georgia Tech Research Institute with over 20 years of industry experience. Mr. Wright created and leads the "Immersive Computing" group within GTRI, which focuses on augmented and virtual reality (AR/VR) development, military wargaming simulations, and user experiences and design. He has been the lead designer on several user-experience focused projects, including projects for the DoD, DARPA, and other governmental and non-governmental organizations, and developed mobile applications for military use. During the summer months, he mentors high school students in AR and VR design and application development.