Think Local: How to Migrate Intelligence from the Cloud to Embedded Devices at the Edge

Chris Shore
Director of Embedded Solutions

White Paper

For the rapidly expanding Internet of Things, “Think Local, Act Global” is a guiding principle. Traditionally, decision making has been concentrated in the cloud, at the centre, moving all the data and intelligence inwards. This places huge strain on the technical and commercial model of the network. In the future, we need to deliberately shift much of the “thinking” to the edge of the network, leaving the central systems to make longer-term strategic decisions based on aggregation of data trends and patterns.

In this paper, we discuss how Arm and its partners are enabling a step-change increase in edge compute capability, whether that’s tiny microcontrollers or multi-core gateways. Rather than a ‘one size fits all’ approach, we provide a unique combination of compute scalability, power efficiency, determinism, and interface options. This is making it possible to do the tactical action-taking quickly and autonomously, away from the centre, relieving pressure on bandwidth, increasing security, reducing latency and improving reliability.

Introduction

We have seen huge growth in computing over the last 30 or so years, moving through many stages in that time. 30 years ago, compute workstations were special. Your company might have had one, your school might have had one, if you were lucky you might have had one on your desk. If you were a really early adopter, you might have had one at home. A small number of powerful computers were shared amongst many users.

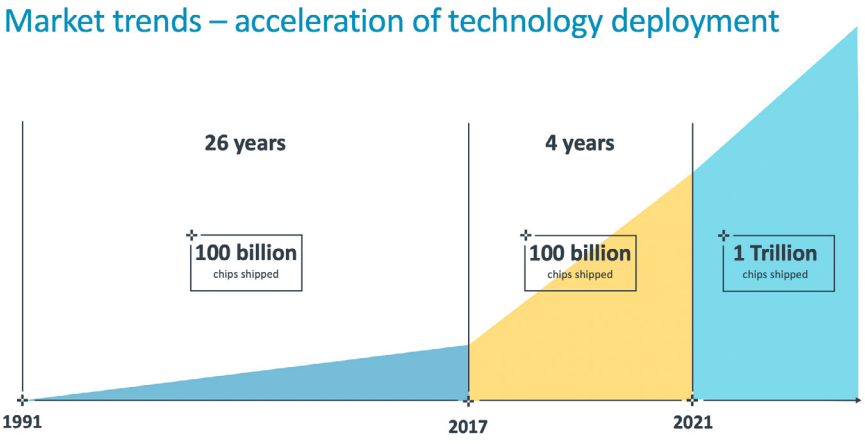
Next came the personal computing era, in which all of us had a computer on our desks, and many of us had one at home. After that it was the age of mobile computing, in which each one of us carries in our pocket – a compute device more powerful than anything of even ten years earlier.

Finally, we reach the era of connected computing, in which all these devices, edge nodes, gateways and servers are connected and share information across the world.

From only a relatively small number of computers 30 years ago, we now live in a world in which computing devices are everywhere. And, increasingly, they derive their value from the fact that they are connected into a huge network of intelligence. The commercial possibilities are endless. However, scaling is becoming a problem. A Gartner Special Report predicts that "a typical family home could contain more than 500 smart devices by 2022." That's a lot of devices to connect, secure and manage!

Arm's silicon partners have shipped over 100 billion chips since we started back in 1990. 50 billion of those have been in the last four years. We expect another 100 billion in the next four years.

Figure 1: Market trends and the acceleration of technology deployment



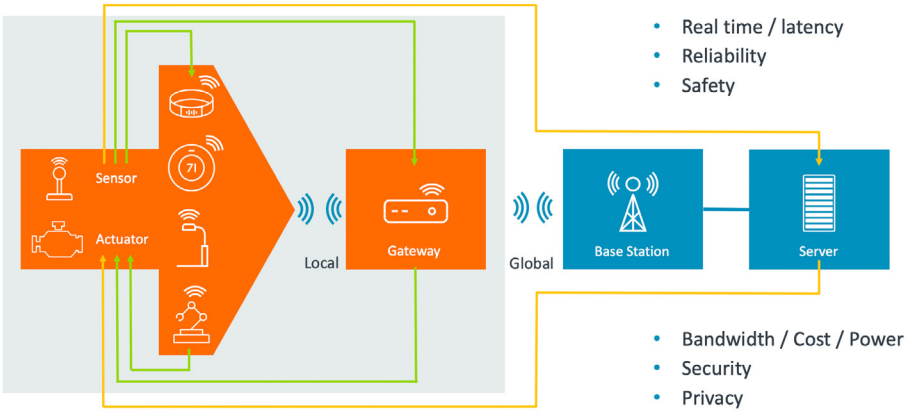
Much of this staggering growth will be in IoT.
These devices derive much of their value from being connected to something central, through which they provide a service to users and suppliers.

Where is the intelligence?

A. The current assumptions

Currently, the assumption is that the intelligence lies in the cloud, at the centre of the network. This has the advantage of concentrating all the data in one place, but it also comes with some significant and growing problems. Fundamentally, it means that we have to spend time, energy and money transmitting data all the way from the edge to the centre for analysis. And then transmitting analysis and actions back out to the edge.

Figure 2: Requirements for intelligent edge computing



So what we have now are systems in which all data is shipped to the cloud. Transmitting all this data to the cloud for processing and then returning the actions all the way to the edge takes too long, costs too much, compromises privacy, uses power and doesn't scale. The founder of an IoT start-up recently said "We like to say that the cloud is a way to scale AI, but to me, it's a roadblock to AI. There is no cloud that can digest this much data."

Much better would be to build the intelligence into the gateway or into the edge device itself. This reduces latency, improves privacy and security, and reduces bandwidth. Then the data sent to the server becomes even more valuable to the service provider as only high-quality information derived from the data is sent to the centre where it can be aggregated and transformed into real business value.

B. The benefits of edge compute

Moving intelligence to the edge offers many benefits.
It improves local autonomy, increases security and privacy, and enhances safety. It also allows us to reduce energy consumption (a few millwatts per device, across a trillion devices, is a lot of energy), reduce latency and reduce bandwidth requirements. There are significant benefits to be had. And, as a solution, it scales to large numbers and large networks much better.

Think local

So, here is a new principle: “Think Local”. Do the thinking at the edge, rather than at the centre. However, we still have to work within the usual constraints that apply at the edge of the system – devices need to be energy-efficient, cost-effective, often battery powered and usually physically small.

Arm has traditionally operated in this area, and much of our technology directly addresses these constraints, helping to build powerful, efficient, intelligent edge devices.

Figure 3: Benefits of edge compute

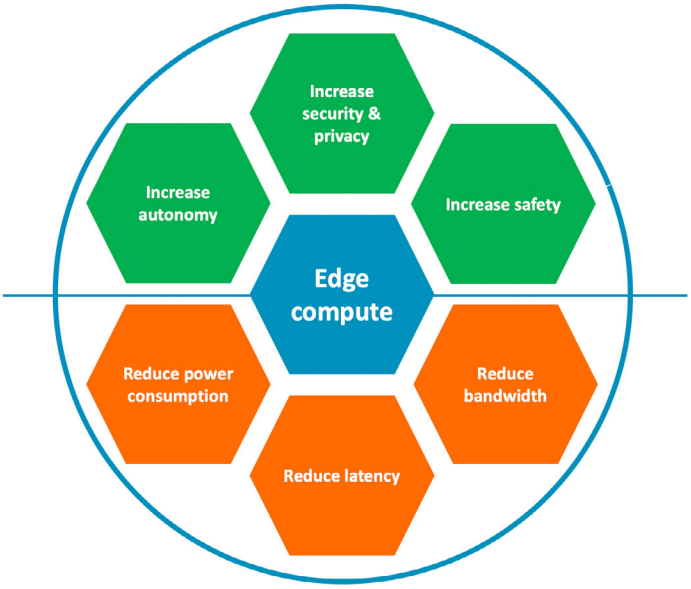
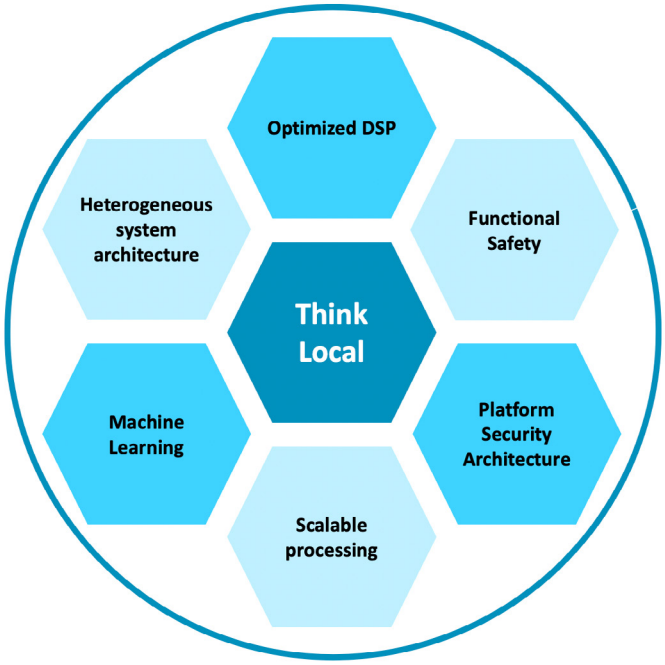


Figure 4: “Think local” concept that combines these key elements



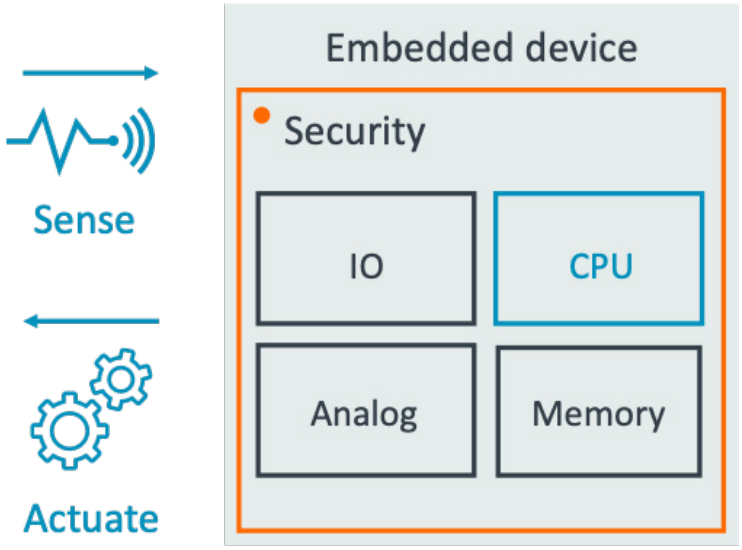
Three key technologies that will impact moving intelligence to the edge are security, digital signal processing (DSP) and Machine Learning. I will now go into detail about these technologies.

A. Anatomy of an embedded device

A typical embedded device senses some inputs from its environment and actuates some control output. Within the system-on-chip (SoC), key IP blocks may include I/O and analog interfaces specific to the application, memory for code and data, and of course, a CPU to perform the necessary compute. Key design criteria are:

- We have enough of the right kind of compute capability
- The device is power-efficient
- The right support is available in terms of software IP, RTOS, middleware and development tools
- It implements an appropriate level of security

Figure 5:
How an embedded device senses inputs from its environment and actuates some control output



Note that, even without connectivity to an external system, security is often still a critical component to protect the integrity and confidentiality of data and firmware, and to control lifecycle management. Security may also offer protection to the manufacturer and supply chain to validate the authenticity of the device, and ensure it's not a counterfeit or cloned part.

Connecting a device to external, central systems adds a whole new level of complexity and affects the design criteria in many ways.

B. Extending with connectivity

When the device is connected securely to “the cloud”, to enable an IoT “service”, the cloud has to be able to “trust” the device, and this makes the security requirements significantly more important. The service will be enabled by software running both on the device and in the cloud, for example, processing captured data locally on the device and sending information about “interesting” events or situations to the cloud, where it can be turned into long-term actionable insights.

An IoT device also needs to be manageable - this includes provisioning the device when if first connects to the cloud, authenticating it when it communicates, controlling updates and security patches, and securely terminating its operation at the end-of-life.
The physical connectivity needs suitable transport-layer security to ensure that private data is kept private, and to avoid all sorts of communication attacks, such as man-in-the-middle attacks.

As you can see, security requirements touch many parts of this design, including identity, connectivity, and device lifecycle management.

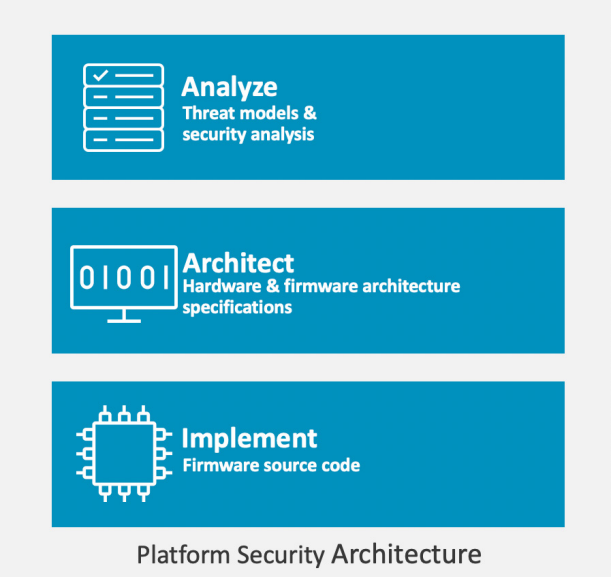
So, we’ll look at security first.

Increase security

First, let’s take a look at how Arm technology provides the foundation on which you can build a secure solution.

A. Security architecture and IP

Figure 6: Platform Security Architecture



The Platform Security Architecture, or PSA, was introduced by Arm in 2017 It is an open methodology for analyzing threats, architecting secure designs, and then implementing secure solutions.

The analysis stage is very important. Not all security solutions are applicable to all situations. For instance, a device which is not physically accessible may not require mitigation for physical security threats. In all cases, the security solutions you adopt must be appropriate and proportionate to the threats that you identify.

PSA talks about four main types of threat:

- Physical: Invasive and non-invasive
- Software: Buffer overflow, interrupts and malware
- Communication: Man-in-the-middle, weak RNG and code vulnerability
- Lifecycle: Code downgrade, ownership change, over-production and debug attack

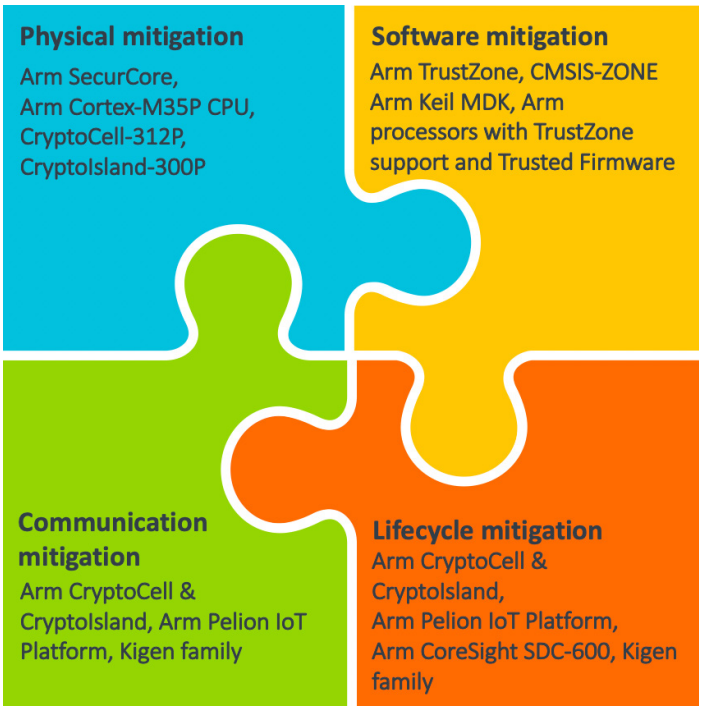


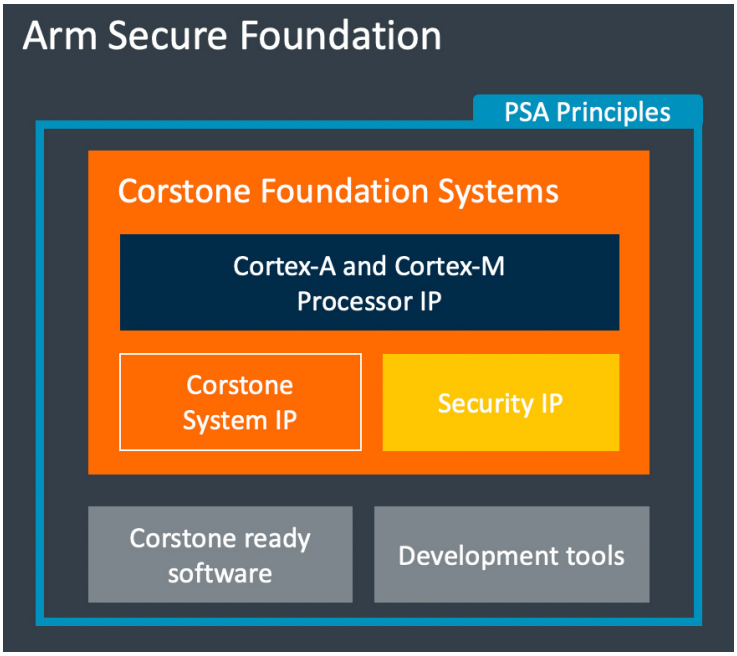
Figure 7: Four main types of threat

Arm has hardware and software IP to address these four main threat types: technologies like Arm TrustZone help to protect against software threats; hardware IP like Arm CryptoCell helps protect against communication threats; Arm Cortex-M35P, SecureCore processors and Arm CryptoIsland products help protect against physical attacks. Arm Kigen family of products helps protect against communication and mitigation attacks. You can find more details about these threats and the relevant IP on our website here.

B. Arm secure foundations help design secure devices faster

Arm secure foundations and Corstone foundation IP offer SoC designers a great solution to build secure designs faster. A range of secure foundations allows for constrained, mainstream and rich embedded designs. Constrained designs are based around Cortex-M microcontrollers, mainstream designs incorporate multiple processors and rich designs use Cortex-A processors for increased performance.

Figure 8: Arm Secure Foundation



The accompanying software packages include security, RTOS and firmware support. Using these designs as a starting point can significantly reduce your design time and get your product to market more quickly.

Optimized DSP functions

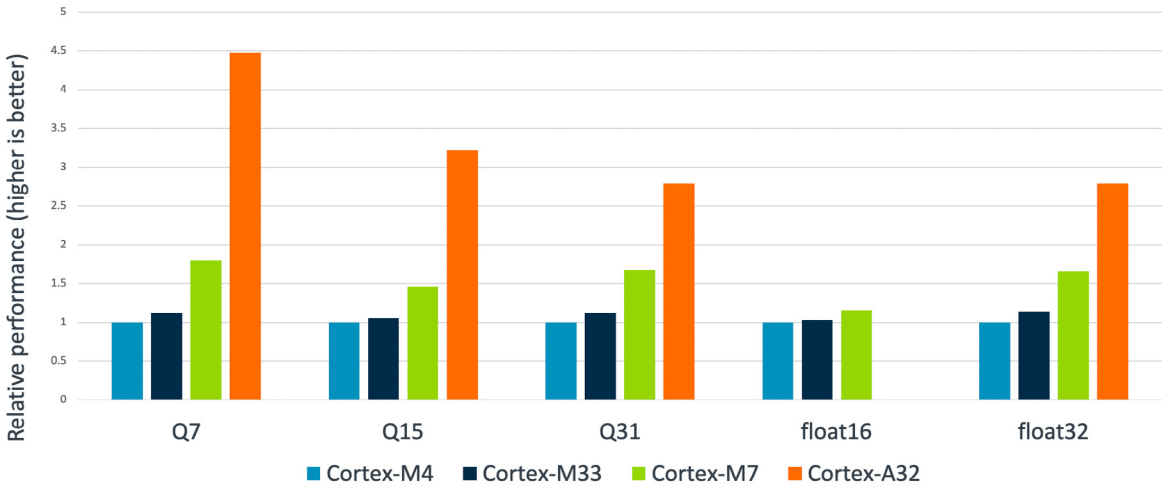
Much of the intelligent behavior that we need to implement relies on having access to optimized software libraries, particularly for functions like DSP and Neural Network acceleration. The two are very closely related.

A. Scalable DSP performance for edge devices

In one sense, all Arm processors are perfectly capable of executing any DSP algorithms. However, as you would expect, some are better at it than others!

You can see an indication of the range of performance points that are available in Figure 9 below.

Figure 9: Relative performance points for Cortex-M processors with DSP extensions



The Cortex-M4 processor, for instance, has many DSP instructions to improve performance. The Cortex-M33 processor offers roughly a 10% improvement on that. The Arm Cortex-M7 processor, however, currently the highest performance microcontroller core, yields a 50 to 75% improvement for many algorithms.

And if you need significantly higher performance, it is worth considering using a Cortex-A processor instead. The Cortex-A32, one of the smallest Cortex-A devices, provides between 200 and 400% improvement over Cortex-M4.

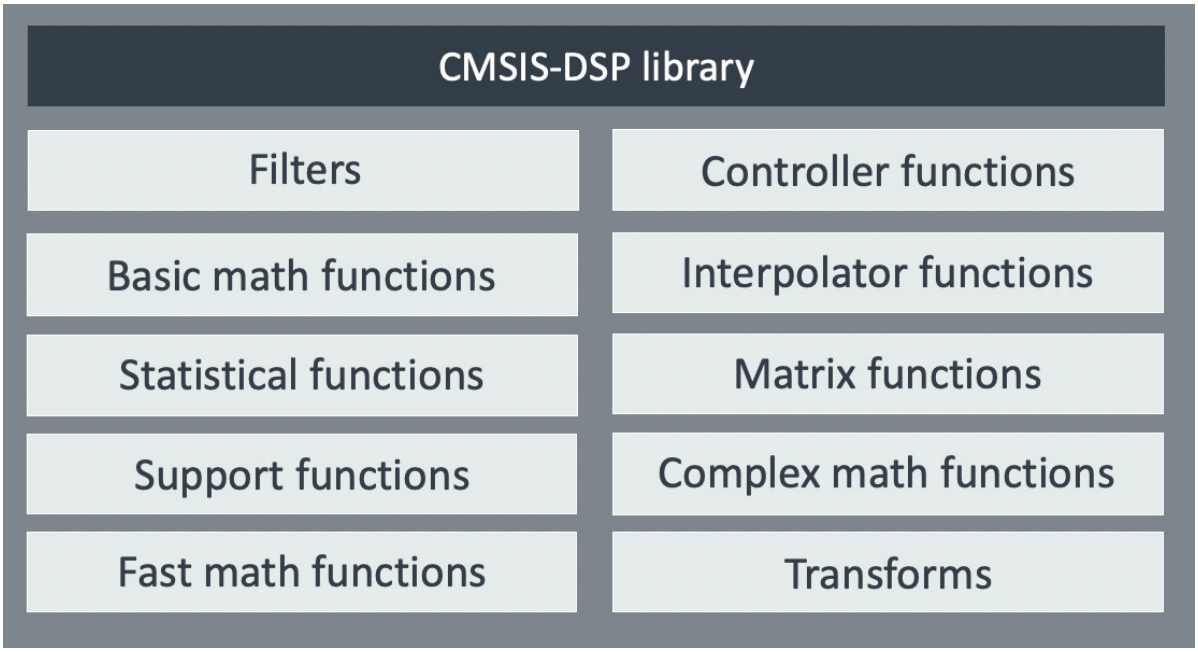
However, extracting maximum performance for complex algorithms like this is often very difficult, relying on specialist knowledge and very careful optimization. The good news is that there is a huge and growing ecosystem of support for just this problem. For more information about DSP for Cortex-M, click here.

B. A versatile DSP ecosystem for Cortex-M

For Cortex-M processors, such as those mentioned above, Arm provides the CMSIS-DSP library. It is hand-optimized by our experts, it's open source, and it's free. You can download it from our website, complete with source code, and use it in your products. Since you get the source code, you can tailor it exactly to your requirements, if you need to.

CMSIS-DSP is quite a low-level library, providing implementations of many common building blocks for DSP functions. If CMSIS-DSP doesn't provide what you need or isn't high-level enough, there is a large number of ecosystem partners with specialist software support for a wide range of common requirements. It is easy to find and get access to functions like image processing, sensor fusion, motor control and so on. And the ecosystem is always growing and expanding into new areas.

Figure 10:
DSP functions on Cortex-M that are available for free



C. A versatile DSP ecosystem for Cortex-A with Neon

Cortex-A processors offer significantly higher computational performance. Much of this extra performance is enabled by the specialist instruction set Arm Neon. This is a powerful and versatile vector processing instruction set with a dedicated register bank.
Neon is well supported by our Arm Development Studio compiler, and by compilers from other sources. Additionally, there is a huge range of available libraries and solutions already implemented that leverage Neon to great effect. Much of it is open source and very readily available.

Machine learning capability

Finally, we touch on the increasingly important area of Machine Learning. This is an area of huge interest right now.

And nowhere is the interest more intense than in the drive to implement meaningful intelligence on constrained devices - the kind of devices which we often find at the edge of the network.

A. Flexible, scalable ML solutions

You might ask what will be the killer app for ML? It's easier to turn the question around and ask what application won't be affected by ML?

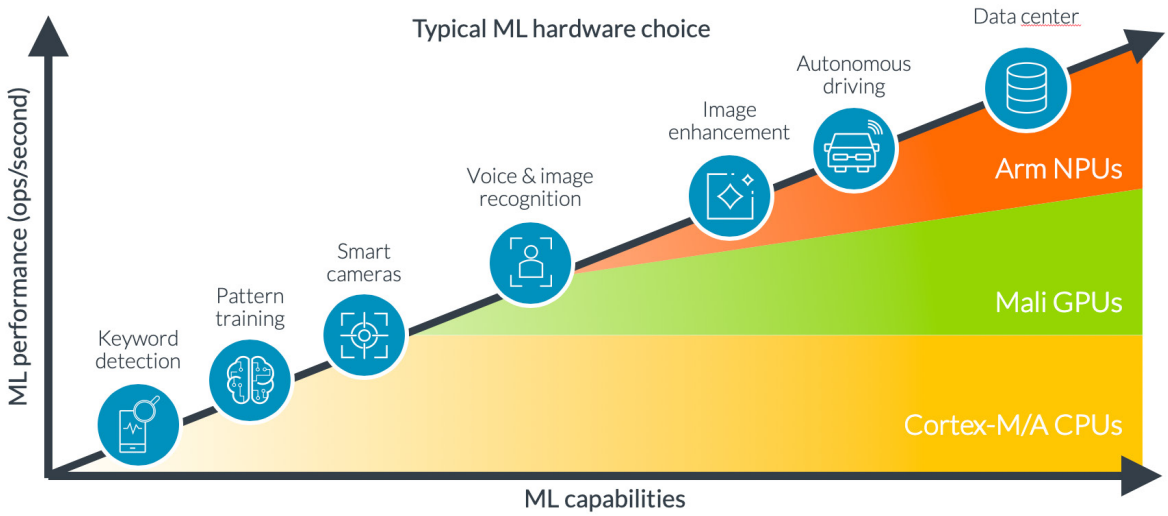
ML is taking place today in everything and Arm is involved at every level. Today, over 95% of AI-enabled devices are in mobile, smart home and IoT market segments, with the remaining 5% in infrastructure.

From that data, we can conclude that over 90% of the AI-enabled devices being shipped today are based on Arm architecture.

Only Arm, through Project Trillium, has the breadth of hardware and software IP offerings and the partner ecosystem to address all these market segments.

And there isn't one way to do machine learning. Many different combinations of Arm IP can achieve the same results with different tradeoffs. From always-on, always-aware devices through to data centers, the Arm ML architecture scales to meet these changing use case needs. Arm ML capability scales from 2 GOPS for always-on devices through to 150 TOPS for server and infrastructure applications.

Figure 11: Flexible, scalable Arm ML solutions



The first dedicated Arm ML processor is targeted at the ML use cases for mobile and consumer devices.

Arm IP supports different blends of compute capability for ML applications. This ranges from standard Cortex processors, through Neural network Processing Unit (NPU), to specialized image signal processors.

This extensive set of options is crucial to enable designers to balance the intelligent capability requirements against the constraints of cost, power consumption and more. All of these architectures can be combined in many different possible heterogeneous systems. For instance, always-on functionality might be provided by small and highly-efficient processors, coupled with much more powerful processors and accelerators in the same physical device.

All of this benefits from extensive software support, specialist Neural Network software packages (such as the open-source Arm CMSIS-NN library), as well as the more general-purpose Compute Libraries. All of them seamlessly support the same machine learning API across a variety of underlying compute hardware.

B. Example: bringing intelligent vision to rich embedded

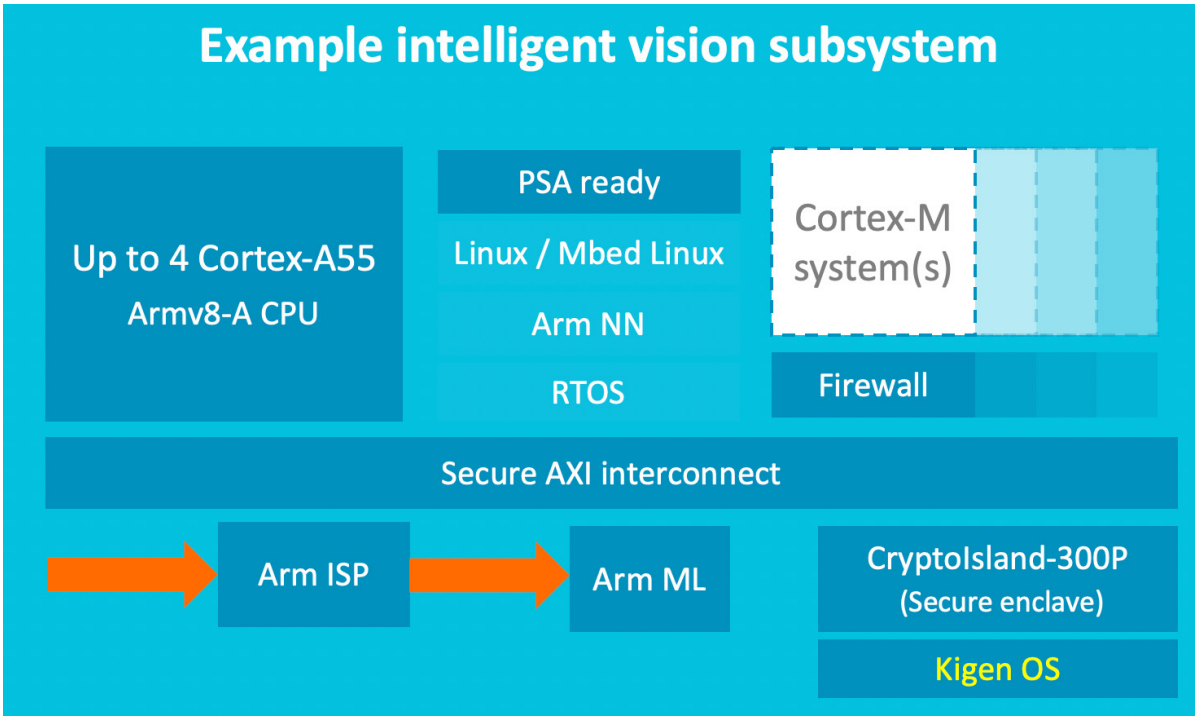


Figure 12: Example intelligent vision subsystem

Here is an example of a system which brings vision capability into a rich embedded device.

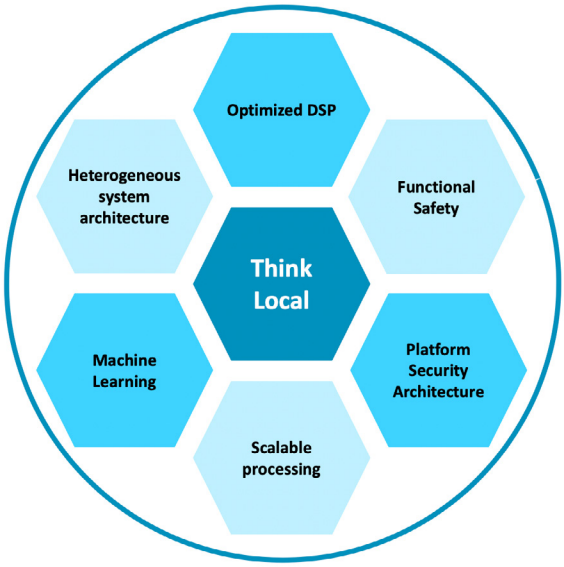
The design includes Cortex-M processors, providing always-on capability, coupled with Cortex-A processors for heavyweight processing when required. All supported by dedicated image processing, object detection and machine learning accelerators. The elements of the software stack are also largely available from Arm too, no matter which combination of IP you are using. In the bottom right-hand corner, you can also see dedicated Arm security IP providing mitigation from a wide range of security threats, and the Arm Kigen embedded SIM subsystem, providing secure identity management.

Think local

In this paper, we have looked at three key technologies which Arm provides into this space to enable the migration of intelligence into constrained edge devices: security, digital signal processing and machine learning.

If space were available, we could also have talked about Arm's significant investment in Functional Safety, increasingly important for a wide range of medical and automotive applications; or the way in which Arm's technology range scales easily from the smallest requirements to the most demanding systems; or how Arm IP makes it easy to build heterogeneous systems that combine extreme efficiency, cost-effectiveness and almost unlimited processing power - all in the same device.

So, it may indeed be right that "no cloud can digest this much data" and to guard against that, we need to "Think Local" and make as much use as possible of the compute capability available to us in edge devices.



I have provided a list of useful resources for more information:

- Developer resources for machine learning on Arm
- Platform Security Architecture resources
- CMSIS-NN software library
- CMSIS-DSP software library

arm