# [Rhombus-Tech](#) / [whitepapers](#) / ecocomputing 07sep2015

**Fund us on [Liberapay](#) . Bitcoin 19trK4AVnCC1YHdJfgWWfpZ9c2uLociNjQ. Bitcoin bech32:**
bc1gx4gy7gguhn79nls58z0y4irz0cxw8a0ew7kwkd

## Table of Contents

## Introduction

The prime strategic benefit - for all its faults - that advances in computing technology have brought us is, by far and above, the global connectivity. Yes there are extreme examples, especially in China, where there are now special laned walkways for people to use smartphones, and there are even "Internet Addiction Bootcamps", but the fact that we can keep in touch instantly with people on the other side of the world, is undeniably beneficial. Especially when the environmental impact of our way of life, incuding the *manufacture* of the devices we now depend on, is beginning to be felt. Ironically, we can at least, using that very connectivity, find out about that very same environmental impact and much more.

The questions this paper asks are, therefore, why are there no truly environmentally-conscious yet desirable mass-volume products available, that both save people money,

give them freedom of choice, allow them to upgrade the hardware on a rolling basis without throwing the entire product into landfill... oh, and as an incidental consequence reduce environmental impact? What is stopping the traditional manufacturers, both Western and Asian, from producing such products (or if there are any, why are they not highly successful, right now?)

Often, by simply asking a question, the answer itself naturally follows. So, therefore, this paper defines a solution, through examples and comparison with existing technologies (both old and in development), to show that it is both possible and practical to provide desirable eco-friendly computing appliances, right now.

# Background

First it's important to go over a few issues - some of them environmental, some psychological and some historical. With this background in mind both the importance of developing eco-conscious computing, as well as how to make it attractive to buy, becomes clear.

## A Brief potted history of computing, including Koomey's Law

Ada Lovelace was the first person to recognise the true analytical value of Babbage's Engine. Bear in mind that this was in an era where "Computer" was, according to Asimov's book "The End of Eternity", used as a title like "Professor" or "Dr", to refer to "one who performs computations". Another sci-fi book, in which the crew lose their computers, has the crew working together *in parallel* over many years, to "compute" a course back to within hailing distance of Earth.

Fascinatingly, many of the parallel computational algorithm techniques developed and well-known before the advent of "silicon computing" fell into disuse and were entirely lost as a science, thanks to the fact that even the slowest computers of the 1970s could, in their sequential fashion, outperform an entire team of human mathematical "Computers". Ironically as the limits of sequential processing have been stretched to breaking point, those parallelisation techniques are now needing to be rediscovered.

The first valve and relay-driven computers filled rooms with literally tons of metal, copper and glass. The "computer" in the Apollo Lunar lander had less capability than a Texas Instruments or Casio calculator (the classic FX100) of the 1980s. Silicon chips are now being produced that have transistors which are only a few atoms thick, and now, in the year 2015, we have Graphics Cards with 4 Gigabytes of memory, that require a 1 Kilowatt Power Supply (more power than most home microwave ovens). 4 Gigabytes of memory just for a Graphics Card. that's more memory than the main processors had in most desktop computers, even only five years ago!

Initial analysis of the rapid development of computing over the past few decades brought us "Moore's Law" - which states that the speed of computers doubles every 18 months. Recently, however, this empirical "law" has begun tailing off, but a much more fascinating empirical law has been found, by Professor Koomey. Using statistical analysis of the power of computers dating back to 1946 vs efficiency, Koomey observed that, in essence, for a particular (fixed) workload, the size of the battery you need will halve every 18 months.
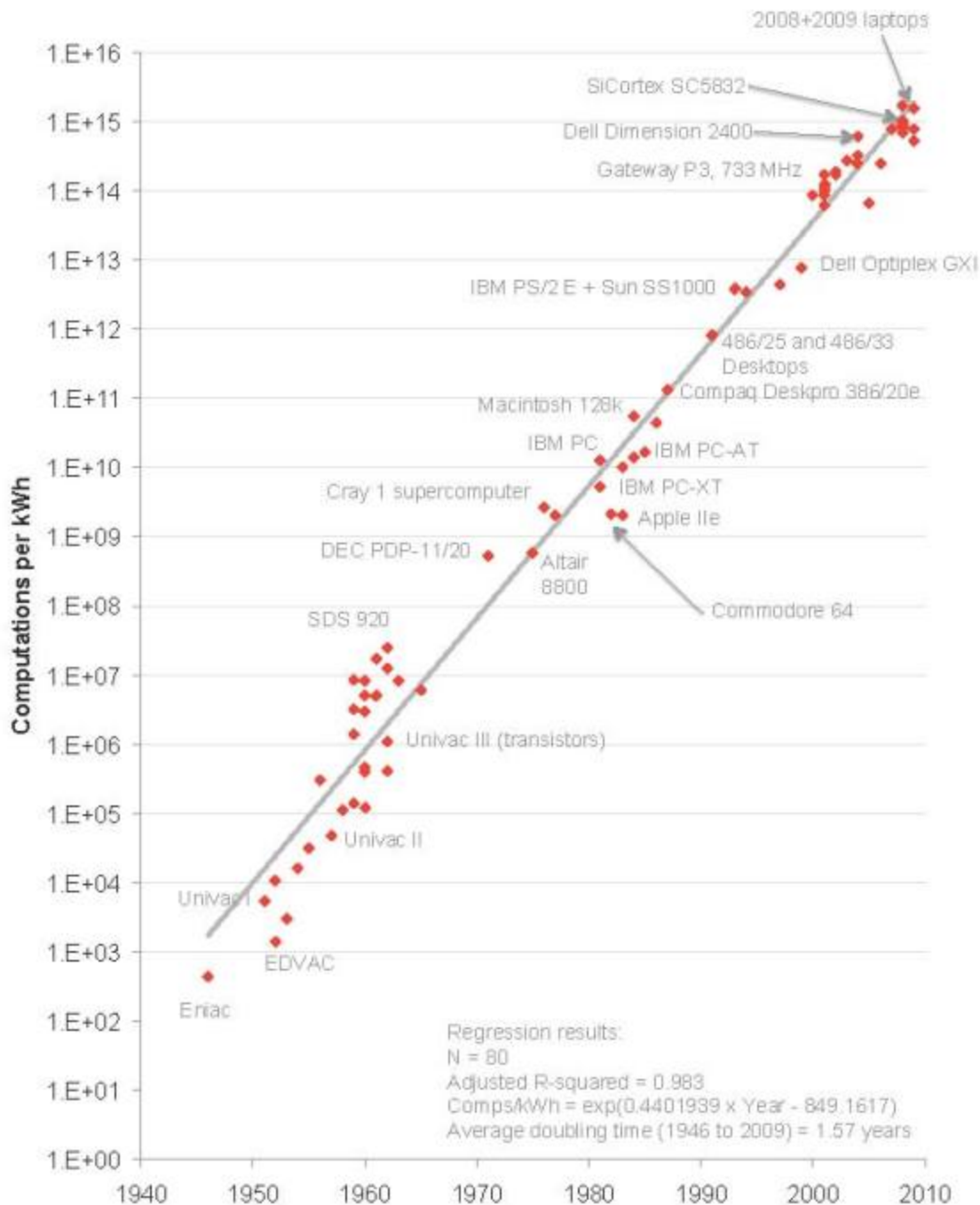
So whilst many are chasing the "ever-faster" dream, seeking the fastest most powerful computer that they can afford, at some point, the speed of computers for everyday use has to - or had to - have been passed, at least by now, surely?

## What is "good enough" computing?

Somewhere around 2009 the phrase "good enough computing" was used to refer to computers that, whilst not sufficiently powerful to fulfil the absolute latest and greatest specialist tasks, nor satisfy the craving and hunger of those who live at the edge of technology's fast lane, would basically "do the job", for the majority of tasks that the average person would ever need. They could buy a computer now, in other words, and, as long as they didn't need to do anything else, they could keep reading email, browse the internet, write documents and print them, watch films, listen to music, and chances

are that the same computer would be there, on the same desk, gathering dust behind its cables, in three possibly even five years time.

*It's worth pointing out that such people are a manufacturer's nightmare...*

Now, in 2015, "good enough computing" has been cross-examined, and found wanting - perhaps not for the right reasons though. The key problem of having a three to five year old computer is not so much that it can't do the job it was designed to do: if the computer was not connected to the Internet it could continue to be used for its designated tasks until it suffered major component failure (possibly in 8 to even 15 years time).

The problem is that the kinds of web sites that most people visit and want to use are being designed with *modern* computers in mind. Even some recent smartphones are more powerful than high-end desktop computers of a decade ago. The latest version of Google Maps, for example, when using the "Street View", overwhelms a recent version of Firefox running on a computer with 8 Gigabytes of memory and a Dual-Core Dual-Hyper-threaded 2.4 Ghz processor, causing it to reach 100% CPU and lock up the entire machine.

But that's not so much the real problem: the real problem is the inter-dependent nature of Software Development. Upgrading even just one application often brings in a set of dependencies that can result in the *entire operating system* needing an upgrade. And the longer the duration since a software upgrade, the less likely it is that one single application may be upgraded without huge impact and inconvenience. With no knowledge (or convenience) on how to upgrade software or hardware, most people pick the simplest solution...

*Sadly, for most people, the solution to "upgrading" is to throw away a perfectly good computer.*

So here we have the beginnings of an opportunity: what if people could upgrade the main part of the computer (including its on-board Operating System) simply by pressing an "eject" button, and, within seconds, press in a new, faster upgrade? Apart from being able to upgrade both the hardware as well as the software, they would save money by not having to throw away the entire device, but also, if the "upgrade" didn't do everything they wanted, straight away and perfectly, they could always press the button on the side and temporarily put the old computer back in the chassis.

As an aside, to those people for whom "good enough" simply "isn't", it has to be pointed out, that in today's "always-on" world, anyone needing access to faster resources than a "good enough" computer can provide, can always access "Cloud Computing" farms, using their "good enough" computer as a remote terminal. With many cloud computing farms now being operated in high-efficiency energy-efficient locations, it actually makes economic and environmental sense to operate a business in this fashion, especially given the extra cost of electricity that businesses are charged.

## The environmental impact of Silicon ICs and Computer Manufacturing

It is not really fully appreciated, even in the tech industry, quite how environmentally hostile and hazardous the development of computing devices, especially the silicon ICs, really is. It's the Elephant In The Room: nobody really wants to admit that their way of life, with all the benefits, is critically dependent on mining and refining some of the most toxic substances on the planet. Perhaps Chemists should know better, but traditionally it is chemists, performing incredibly complex deeply computationally-expensive simulations ever since computers first became available over half a century ago, who are one of the largest users of computing power.

The clue to the environmental toxicity of Silicon ICs is in the word "doping". The rare earth metals that are used are, by definition, extremely rare. They are usually found in deposits that have radioactive isotopes surrounding them. To make matters worse, the Foundries that make Silicon ICs need unbelievably vast amounts of pure water.

So we have a situation where our way of life - our interconnected-ness with the rest of the world - is dependent on highly toxic materials as well as depriving people of their water supply. That the factories are usually situated in remote areas does not make the problem go away: it just means we don't get to hear about it.

Incidentally it is a similar story with cars: a company called Divergent Microfactories, noteworthy for producing a 750HP supercar with an environmental impact 20% of anything else ever previously built (and that includes its entire lifecycle), noted that 80% of the environmental damage of any vehicle is done even before it is sold. And there are people who *genuinely* believe that converting an existing eco-hostile vehicle manufacturing strategy over to hybrid or full electric without making any other kinds of design or manufacturing changes, is somehow "better for the environment", when copper is already scarce, the neodymium used in magnets requires 1,000 litres of boiling sulphuric acid to refine just 1kg of neodymium, and the lithium used in the batteries is in essence a highly toxic explosive pollutant if it ever comes into accidental simultaneous contact with water and air.

*So why, then, if this is the case, are current manufacturers not doing more to reduce environmental impact?*

## Corporate Irresponsibility: the consequences of maximisation of profit.

In an extremely well-written book called "The Other Side of Innovation", great care is taken to praise the amazing efficiency that results from following an effective corporate manufacturing strategy. The success of a mass-produced product *critically* depends on reducing the financial cost of components and assembly, and much more, so that the final product price is affordable and competitive.

However: as "The Other Side of Innovation" points out, with that drive to efficiency comes a penalty. The people and the processes that are selected and focussed on efficiency are *not* suited to either innovation or to environmental strategies. In fact, the book goes to some lengths to illustrate how *not* to interfere with the "Efficiency Engine" as it is called, on the basis that it is primarily the "Efficiency Engine" that is responsible for generating the profits on which the continued existence of the entire company critically depends.

But this is not all: worse than the necessity to not interfere with the efficient mass-production of products is that to sustain the ever-downwards drive on pricing comes an imperative need to continually sustain profits, at all and any cost. For the Directors to order otherwise actually exposes them to liability under Company Law! The current solution to this problem is to continue to design "better" i.e. "more attractive than the previous model" models in a continuous chain of near-identical products that disrupt the design, factory and manufacturing process as little as possible, with as little staff retraining or innovation as possible (that is not easily obtainable from external sources or resources), but also continue to sustain the business model long-term, keeping everyone in the chain gainfully employed. Older products are quietly abandoned: even security updates terminated even if there was a public committment promising to supply them, because every penny spent on every software engineer working on the security updates for the older products is "profit utterly wasted" in the eyes of the Pathological profit-maximising Corporation (Professor Yunus explores this concept more fully in his book "Creating a World without Poverty").

Translation (i.e. consequences) of the above paragraph: eco-conscious products have no place in a profit-maximising world, primarily because it is *perceived* that there could not possibly be enough profit in them.

Fairphones (a Cooperative based in Amsterdam) appears, initially, to be a counter-example to this, where, from no prior experience, they are doing the best that they know. Based on commonly-understood Cooperative principles, Fairphones began by working to source ethically-produced materials, and to ensure that the factory employees in China have pensions and good living conditions.

However, they did not take into account the unique challenges of the smartphone and computer manufacturing industries. As a new company, with little experience in the actual environmental impact of smartphones, they forgot to take into consideration that the software is entirely illegal (Copyright violating), that it is supplied illegally under NDA without respect for the Libre (Free) Software License by the manufacturer of the smartphone's main processor (in the case of the Fairphone 1 it was a low-cost Mediatek SoC), that the factory making the hardware have absolutely no access whatsoever to the source code, and that as a result not one single end-user may upgrade or even fix security problems in the software on their own legitimately-purchased hardware, *even if they were technical experts*. End result: every single Fairphone ever sold is now hopelessly out-of-date, has major security vulnerabilities, and, despite the actual hardware being perfectly functional, is now not only useless but is a hazard to its owner's privacy and security.

To Fairphone's credit, they appear to have learned this hard lesson and are taking it into consideration for the next phone that they produce. Latest news is that they intend to make the design "modular", exactly as, this white paper, ultimately recommends.

*The devil in modular design is, however, in the details*.

## People don't care about social issues, but they care about themselves.

Fairphone owners are the exception to the rule. The first Fairphone was brought about successfully through crowdfunding, specifying a Minimum Order Quantity (MOQ) of 10,000 units in order to get pricing down through bulk purchasing and to reduce the impact of setup and teardown costs on tooling. Ultimately they sold over 60,000 units. Even with that (apparently) huge volume of orders, the actual sale price of the first Fairphone was extremely high: USD 433 on an unsubsidised price - without a phone contract. The Fairphone 2 has an even higher price: EUR 530. Amazingly, despite the high cost, at the time of writing they are just over two thirds of their way to a 15,000 MOQ.

For the majority of smartphone users, shelling out EUR 530 up-front and waiting for possibly up to a year for delivery is simply too much. In India (and now West Africa) we have Google bringing a low-cost "Android One" phone based on the "good enough computing" principle, selling for USD 83 in West Africa.

In the Western world however the critical price threshold for "consumable" electronics is the figure "99". In the UK in 2006, a tiny clam-shell laptop was available in Maplin's and other stores for 95 UK pounds. This product, rebranded throughout the world under over 30 different names, was codenamed the "Skytone Alpha 400" and it was the first real low-cost successful portable computer to come out since PDAs made their first debut in the 1990s.

Now, incredibly, since around 2012, thanks to companies such as Allwinner who brought us the world's first 1ghz USD 2.50 processors, we have tablets coming out of China,

Taiwan and Hong Kong for under USD 35 retail. Ingenic and ICubeCorp, two other SoC manufacturers, also now have USD 2 and USD 3 products: ICubeCorp's processor is even quad-core! Allwinner now have an IPTV processor which has such high integration (on-board Ethernet, HDMI and SATA) that a product (known as the Orange Pi) can be made using the Allwinner H3 at a sale price (quantity one) of only USD 15 plus 4 shipping from China. It's a quad-core processor, running at 1.6ghz, and has 1Gbyte of memory.

In summary: people are, as a general rule, attracted to the highest features for the amount of money that they can afford that will allow them to do what they want: communicate, work and enjoy life. In the mass-volume arena, exceptions to this are, in the Western World (and increasingly in China itself), when the price drops below a certain critical "disposable income" threshold, but also when the perceived (or actual) quality is sufficiently high that people consider a higher - much higher - price to be fair and equitable. That used to be the case with Apple products (however, it is probably more likely that people were attracted to "products brought about through the Steve Jobs Vision").

Compared, then, to Apple, Dell, HP, HTC, Samsung and the hugely successful no-brand products rolling out of China, sold literally in the tens to hundreds of millions per year, Fairphone's MOQ of 15,000 is a drop in the ocean. Social studies, unfortunately, confirm that taken en-mass, individual purchasing decisions are driven by self-centred interest, not by larger social issues. Small fortunes are spent by large corporates to take advantage of this well-known phenomenon, in the form of marketing and advertising campaigns.

*Unfortunately, then, eco-conscious computing as a primary sales tactic is also a minority sales prospect.*

The hard lesson, here, then, is to come up with a strategy that is both environmentally-conscious *and* saves people money. Interestingly, a modular design, where the main computing unit could be shared across multiple products, saving people over 40% of the cost of buying two computer devices, would *achieve exactly that*.

# Modular Computing

The general theory behind modular computing is that it is inherently upgradeable and expandable over time. As the needs of the user change, so can the computer itself, without disruptive pricing or useability interference. As we've seen so far, there are quite some barriers to overcome.

However, modular computing is not new. It has huge precedent (dating back several decades) and is beginning to make a come-back. It is therefore worthwhile taking a brief look at the historical uses of modular computing, then covering some existing uses, then going on, bearing in mind the goal to create "affordable desirable eco-conscious long-term computing", to define a standard, compare it against existing ones to ensure its viability, robustness and "market uniqueness", then, as it is a future standard, envisage some scenarios that would, over the next decade, test its likely success.

## A Brief Potted History of Modular Computing

When computers cost over USD 10,000 it was an insane proposition to create them as a single board. Not least, such a board would have taken up a space of over 2 square metres, because they were typically constructed from hundreds of 8 to 24 pin "discrete chips". The 16 Megabyte Memory Board of the Apollo Computer, for example, was an entire square foot of circuit board.

It therefore made sense for a manufacturer to create a "backplane standard", into which (in some cases) not only the main computer "card" could be plugged but memory cards and peripheral cards could be plugged as well.

One classic example of the benefits of modular computing, working in conjunction with good software, was a story from an Imperial College Professor, who in a prior incarnation as a Computer Salesman, demonstrated a huge 1ft cube computer packed with interface cards, where its main selling point was the ability to save its current state to storage if it detected a power-loss event. Entirely by accident, during the demonstration, the professor walked around a desk and not only pulled out the power cord but also managed to impart enough momentum so that the computer fell off the edge of the desk. Now, *as it was making its way to the floor*, the software correctly noted the power-loss, saving the demonstration to hard disk storage just in time for the computer to hit the floor, bending the metal cage and causing all the peripheral cards to fall out. The professor said that he calmly picked up the computer, bent the cage back into shape, re-housed all the cards, plugged in the power, whereupon the software *restored the demo* and he was able to carry on. The unintended demonstration did more for the sales of the product than his slides. If this product had been a monolithic device, chances are its impact would have destroyed several critical (inflexible) components.

Moving on to the success of the IBM PC, we saw the introduction of the 8-bit "XT" memory bus standard. This standard allowed several cards to be addressed by the main processor. It was later extended, when the Intel 286 and above came out, to a 16-bit standard called the "AT" Bus. Video Cards, Modem Cards, Sound Cards, Floppy and Hard Drive Cards - in the late 1980s and early 1990s it was commonplace to buy upgrade cards and to install them yourself, not least because the cost of the computer in the first place was so insane, it was financially unjustifiable to purchase everything in one go, but it was commonplace to transfer perfectly good peripheral cards when buying an upgraded base unit, especially the high-end (expensive!) Audio, Graphics or Hard Drive Cards available on the market.

IDE Drives were the first attempt at cost-saving (over SCSI Drives): IDE Drives were merely the IBM "AT" Bus farmed directly through to the circuit board of the hard drive itself, removing the need for the (expensive) addition of a SCSI controller board, because the "AT" peripheral card could be incredibly simple (and low cost), simply "passing through" the signals from the main motherboard directly through to the chips on the IDE drive's circuit board.

The IBM "AT" Bus is, amazingly, still with us today. The first form which made it into true "portable" form was the PCMCIA Memory Card. The PCMCIA Standard is, in essence, entirely and solely the 16-bit "AT" standard: nothing more. It was introduced when the first laptops needed an easy way to be upgraded. However, some bright engineer noted that, historically, "AT" peripheral cards are "memory addressable", and, thanks to the design of the "AT" Standard, making memory-addressible Modem, Ethernet and later USB "cards" became a straightforward proposition. It then became possible to sell, for a whopping USD 2,000, a laptop that had only the most basic of interfaces, but that had 1 or 2 PCMCIA (later CardBus) slots that allowed end-users to buy only the extra peripherals that they needed, or even upgrade them later. Dial-up Modem technology in particular suffered from a particularly fast turn-over in speed increments, going from 9600 baud to 115200 baud within a relatively short period of time.

Today, however, the only remaining uses of the old IBM "AT" Bus are in the PC-104 Industrial Standard and in the form of Compact Flash Memory cards. Even Compact Flash Cards are beginning to be superceded by the smaller "SD" and "MicroSD" card form-factor.

Other examples of Modular computing are found in the mainframe arena: "Blade Computing" takes the modular concept to an extreme, placing an entire modern

computer onto a backplane stuffed with connectivity that is designed for massive parallel compute tasks requiring huge I/O bandwidth.

However, in the rest of the computing world, the constant pressure to reduce price has led to hermetically sealed units. Apple products are basically non-repairable. The plastic is hermetically sealed using ultra-sound so that there is literally not even a seam to open in order to replace components: the product *literally* has to be destructively cut open in order to effect repairs.

More recently however there has been a consumer backlash against this kind of irresponsible design ethic, based on the perfectly justified annoyance at having to entirely discard devices rather than effect repairs (for a reasonable fraction of the cost of a replacement equivalent device). Many people tolerate broken screens on their smartphones only whilst waiting for "the next contract upgrade opportunity".

So within this backlash is a perceived market opportunity to sell much more "modular" electronics appliances, just as they were available world-wide, such a historically incredibly short time ago.

## Existing Modular (mostly smartphone) efforts

The first real modular portable device in smartphone-sized form-factor available on the market was from BugLabs, around 2006. The company has since morphed onto the "IoT" bandwagon, but they were definitely the first company to bring us modular upgradeable hand-held elctronics. Their primary focus was, however, on "engineers" - on allowing engineers to rapidly prototype a portable device, by buying different "modules" - a 3G modem, or a different LCD screen, or a WIFI module. Their business model was based around making it easy to customise not only the software but also the hardware. As such, it wasn't really of interest as a mass-volume product.

*The Lesson: design the product right from the start to be mass-volume, but save development costs by making it truly open (libre) and thus attractive to geeks.*

Several "stealth mode" startups have since been developing modular smartphones. A Finnish company, staffed by former Nokia employees, recently lost the battle against market and investment forces, selling their designs to a 3rd party rather than bringing them to market as unique products.

The Puzzle Phone is an effort to divide phones into three main parts: a backplane (which holds the LCD and long-life components), the main CPU module, and a battery module which has secondary electronics (presumably WIFI, Bluetooth, camera and others) that are likely to change rapidly over time. The tagline on their web site "Repairable: because you do not get rid of your car if the wind-shield breaks" is particularly telling. However: they claim "open-ness" but there is no link through to an actual Libre (Free) Software Community on the web site. Also, although the hardware standard is also claimed to be "open" there is no evidence of links on the web site through to an actual standard that is open for public review.

It turns out that the Fairphone 2 claims to have modularity. This appears to be more about making it easier to consider opening and repairing the internal components than it is to make a true "modular marketplace". As a compromise strategy, from a Cooperative that is only aiming for the 10k to 100k sales markets (at the time of writing), it's a good one. To their credit they're working hard on transparency in the supply chain, asking the hard questions about whether the materials are conflict-free. So their sales strategy is not just about receiving a phone, it's about allowing people (the end-users) the opportunity to make a difference to the quality of life of other people involved in the manufacture of the products being purchased.

The one modular phone design that everyone in the tech industry is talking about, however, is a former Motorola Research Initiative, now known as Google "Project Ara". Project Ara promises to deliver a truly-upgradeable smartphone, pretty much exactly as envisaged by a former University student who set up the PhoneBloks modular advocacy forum. The basic principle of Ara is that there is a (replaceable) backbone, onto which tiny "modules" can be hung, including WIFI, camera, 3G, 4G and beyond as they become available. Even the LCD may be removed, sold on ebay and replaced with an upgraded unit. It promises to be the best thing ever: a revolution in smartphones where you can start with a USD 50 pricetag and upgrade all the way to USD 500 if you so desire.

Below the surface, however, we see evidence of some severely poor design decisions, ironically caused by Google's team basically having access to more money than any other team would normally have, thus putting them in a situation where, instead of solving challenges creatively, they've applied financial brute-force instead. The ramifications of this type of development methodology are far-reaching and severe, and are related to the world-wide monopolistic business practices which the average end-user never sees any direct evidence of (except in the form of a higher cost of product ownership).

Key to Project ARA is an entirely new standard, called MIPI UniPro (Trademarked, Copyrighted, Patented and, last but by no means claimed to be least, "open"). Anyone familiar with trademarks and patents should already have that sinking feeling, but for those that are not, the story needs to be unfolded. Patents were intended to encourage innovation: they're a government-sanctioned monopoly, intended to allow the patent holder - the innovator - a limited amount of time in which to financially exploit the market opportunities they "clearly deserve". It is little-known that enshrined into patent law is the right of anyone to make an implementation of *any* patent, so that the new inventor may make improvements. This is the only sign of the original intent of Patent Law: that it is there to *encourage* innovation.

The reality is known to be radically different: patents are now the near-exclusive domain of immense multi-national corporations. Individuals very rarely successfully use patents for their intended purpose, nor do they financially benefit from them. Patents are used instead as "weapons" and multi-nationals have to hold thousands to tens of thousands of patents (and pay yearly fees for them) as a "Nuclear Obliteration Option" against any other large multi-national that might even *consider* getting away with asking for royalties. Compound the sheer complexity of writing successful patents, along with the fact that patents are not properly reviewed for uniqueness let alone "inventiveness" by Patent Offices overwhelmed with technical details as well as sheer volume, and we end up with the insane high-profile cases of patent wars witnessed recently between Apple, Oracle, Samsung, Google and many others, as well as the rise of the "Patent Troll": corporations that exists solely and exclusively not to create inventions but to buy up patents, seek out "infringers", extort money (with government sanctioned approval) that is deliberately calculated to be just less than the cost of a litigation defense, and to instantly engage in litigation if the victim refuses to pay.

So Google worked with a limited number of companies to develop MIPI UniPro chipsets, in order to maintain secrecy of the project. Also, due to the sheer cost of developing chipsets (USD 10m to 50m is not uncommon a development cost) the number of possible partners is pretty limited. However, that itself is not the problem: the problem is when patents are involved in the mix, as well. Once even one single "partner" submits a patent, all and any other potential partners - regardless of the potential cost reduction for end-users - *automatically* bow out of considering *any* involvement, due to the simple fact that the patent royalties demanded by the "first" innovator would put them out of business. Rather than risk the prospect of committing USD 10m to 15m only to have zero customers buy their price-uncompetitive product (because of the additional financial burden of paying the patent royalties), they simply "bow out", leaving the entire field open

and exclusive to that "first" innovator. And when an "innovator" has an open and exclusive field - a monopoly - guess what happens to prices?

It will be about twenty years, now, before those first patents expire. So, logically, we can expect price-reductions in "Project ARA" components... in pretty much two decades time (by which time the project will either likely have failed or will have been proven to be irrelevant).

The second problem is related to the monopolistic and cartel business practices that were exposed within the LCD manufacturing industry a few years ago, when many high-level executives were arrested. We *know* that such cartel business practices are in place... so *why* did Google pick an exclusive, patented known-monopolised and cartelled communications interface (MIPI) as the fundamental basis of its smartphone backbone?

The implications - just for LCD sourcing alone - are that anyone considering making a small run of ARA-compatible smartphones (despite the promised "open-ness" of the ARA phone) - is inherently excluded. LCDs for smartphones are usually custom-made in special runs to exclusively fit a particular phone manufacturer's requirements. Those suppliers will be extremely reluctant to jeapordise their extremely lucrative relationship with the end-product manufacturer by giving you, the small-volume orderer, *any* kind of access to the components that were specifically manufactured for another client.

So there are, basically, accidentally-cartelled but logically and rationally exclusive relationships between the top smartphone manufacturers and the very few LCD companies in the world. Orders of existing designs *might* be possible beginning with a MOQ of 100,000 units. Custom designs by a complete outsider *might* be possible with a MOQ of 1 to 5 million units, with a hefty cash deposit. Imagine Fairphone contacting one of the top LCD manufacturers, for example, and asking for a custom from-scratch order: a special run of 15,000 conflict-mineral-free LCDs. This would be simply flat-out laughed at, once a puzzled LCD saleman put the phone down on them, because the LCD salesman would know the overwhelming cost of development of such a customised product, but could in no way tell a potential customer that.

Then, also, getting hold of the ARA-compatible chipsets to create a compatible product would, again, be near flat-out impossible. If you went to the manufacturers of these chipsets today, directly, then apart from them getting very nervous because you hadn't signed an NDA, and apart from them being fearful of upsetting their relationship with Google, they would ask you for a cash up-front order, MOQ of 10,000 to 50,000 chips or above.

*The lesson: under no circumstances build a modular standard based around newly-designed, newly-patented standards. make it from decades-old proven standards, instead*.

So, sadly, whilst Google has promised "open-ness", and has a standing open invitation and call for participation in Project ARA, there are sadly financial *disincentives* for both small and large entrepreneurial businesses to get involved, thanks to patents as well as cartelling business practices. Google's project *might* become ubiquitous in around 2 decades time, but until the patents expire, and if they can come up with a solution to the LCD cartelling business practices, Google themselves will need to be the ones that pay for the Research and Development of modules, extensions and new products.

## What are the defining characteristics of a successful eco-modular standard?

Thus far a fairly bleak picture has been painted, of ongoing failures, industry stagnation, user frustration and much more. To even consider that there would *be* a solution that

could successfully "run the gauntlet" of such ambitious aspirations would seem to be near-impossible. However: Arthur Conan Doyle's character Sherlock Holmes famously said that "when the impossible is eliminated, whatever remains, *however improbable*, must be the truth". Thus, taking inspiration from this quote, and applying rational reasoning without judgement as to the "level of ambition" or otherwise, we can clearly and rationally envisage a potential candidate standard, and, with the right creative attitude, test it out.

## Inspiration from other Industry Standards

The first thing to do when developing a standard is to examine other standards, to see how they succeeded (or otherwise). Not least, there may be an existing one that could be used, instead. The web page for the [EOMA Standards] (http://elinux.org/Embedded_Open_Modular_Architecture outlines) has a specific section dedicated to the analysis of alternative standards. The first thing to note: *all* of them - without exception - are "Industrial" style standards. Q-Seven. PC-104. None of them are designed for the mass-volume market, where issues such as user-handling, robust casework and ESD precautions are all critically, critically important.

The second key issue is to choose a case and a connector. In order for the project to be achievable *at all*, in a global market where the cost of creating a new pair of connectors and associated housing, plus tooling, plus casework, could be well in excess of USD 0.5 million, as well as require an extremely demanding MOQ of 1 million units in cooperation with a factory in China, re-use of *existing* connectors, housings, sockets and assemblies seems to be a pragmatic solution. Not least, the re-use of pre-existing offerings fits with the ethics of reducing the environmental impact of the final products, and to begin a product family by having a huge environmental impact by having entirely new tooling made when there are perfectly good ones out there already, seems entirely at odds with the ultimate goal.

So, again, as outlined on the EOMA Standards page, several potential candidates were evaluated, including Mini PCI (not to be confused with Mini PCIe), ExpressCard, Compact Flash (mentioned previously), MXM, 200-pin SO-DIMM and other memory-module standards, and PCMCIA (mentioned previously). From 2010 to 2011, both re-purposing of PCMCIA and re-purposing of CompactFlash were initially chosen, and draft standards drawn up. Re-purposing of ExpressCard as a 26-pin standard was added around 2012. The next four years were spent evaluating and refining these three standards, and picking one to focus on for a proof-of-concept.

After seeing that there were no pre-existing standards, one had to be created. However there was a key lesson learned, again from the same Imperial College University Professor previously mentioned, who was keenly aware of, for example, why the X.25 standard (the backbone of the JANET inter-university network) did not supercede the RS232 standard despite RS232 being, speed-wise, so amazingly poor in comparison.

X.25 was a 9-wire serial protocol, with receive and send data lines, as well as two "interrupt" lines (receive and send) to distinguish between data packets and "control" packets. There was, however, only one "Clock" line: receive only. "Clock" was expected to be generated by an external (powered) source, to be received at both ends, and for each party to synchronise both send and receive to this external clock.

Within the standard, however, it was "recognised" that there might be the possibility that a receiver might not have the "interrupt" hardware line. So, in *software*, it was permitted to send a special "Escape Sequence" - a data code - that was recognised as being a "marker" to indicate the beginning of "control" packets.

All is well so far... except let's now analyse that. The problem is that you don't necessarily know whether the receiving end has a "control" line or not. It might... or it might not. Therefore, to make absolutely sure, it is necessary to *ignore* the hardware control line... and *only* use the software "escape sequence", just to be on the safe side.

The upshot of this simple logical analysis is that the hardware control lines are *utterly wasted pins*. Meaning, that on the 9-pin X.25 connector, one of those wasted pins could have been used as a "Transmit Clock", reducing the Bill of Materials for deployment of an X.25 network by as much as USD 20 because a powered external clock box could be replaced entirely by a simple USD 2 cable. Thus, through a simple design flaw which could have been caught at the standard's design phase with simple logical analysis, the opportunity for X.25 to be a competitor to RS232 was lost.

*The Lesson: under no circumstances allow ANYTHING to be optional in a Hardware Standard*.

Upwards-compatibly expandable at the hardware level - yes. Optional - no.

This is where standards such as 96boards, Q-Seven, EDM, ULP-COM and so on make matters extremely complex for designers to consider supporting (or implementing) them. Imagine a scenario where a designer has to choose whether to use MIPI or LVDS for the LCD interface. The standard says that the CPU Board may have either of these... so how on earth is a designer supposed to choose which CPU Board to use? What happens in the future if LVDS becomes obsolete (it won't), but the base unit was designed around LVDS, and the only CPU Boards in the locality of the purchaser are now MIPI? What if there are two or more such combinations of "optionality" (as is very common with these standards). What happens if, in the future, the CPU Board's main SoC, which was initially chosen for its highly-competitive price point, goes end-of-life, but there are no new SoCs with the *exact* combination of interfaces chosen over a decade ago when the product first came out? And yes, in the Industrial Markets, choosing components with a ten-year manufacturing committment is extremely common, because Industrial equipment is genuinely expected to last that long, if not longer.

So it's not a joke: it's an absolute nightmare that seems to be repeated time and time again, from which modern Standards Designers repeatedly seem unable to learn from, despite the fact that it is quite straightforward to do the scenario-analysis and come up with a standard that greatly simplifies its long-term deployment and thus inherently increases its longevity and acceptance. The COM Express Standards Team managed it, so why can't any of the other embedded industry standards teams?

Another flaw to avoid is: picking the wrong voltage levels for GPIO. A key aspect of successful standards is that there is a "reference supply" voltage, against which the "high" logic level is expected to be measured. Electronics circuits "1s" and "0s" are represented by voltages above and below a certain threshold. For example, CMOS defines these as being "above 0.7 times the reference voltage is a 1" and "below 0.3 times the reference voltage is a 0". There are two possible solutions therefore. The first is to fix the reference voltage, and the second is to make the reference voltage variable, and supplied by one of the pins on the connector (from the CPU Board). 96boards chose a fixed voltage (of 1.8 volts). Sensible standards choose a variable voltage, supplied by the CPU Card.

The reason why a fixed voltage is a problem is simple: a fixed voltage requires level-conversion. It doesn't matter what voltage is picked: it will *always* be wrong. Let's say that you pick 1.8 volts - you just automatically excluded (or made cost-ineffective) any CPU Card using a CPU with a voltage *other* than 1.8 volts, because that CPU will require bi-directional level-shifter ICs *on the CPU Card*, the cost of which (in terms of space as well as financial cost) could easily exceed the available budget. Also, given that many pins are dual-purpose (as specified in the standard), it becomes so extraordinarily

complex to work with a CPU other than one that has the exact specified voltage level (of say 1.8 volts) that it is utterly cost prohibitive to consider. And, with geometries reducing all the time (with CPU voltage signal levels correspondingly reducing as well), any standard that mandates for example a 1.8 volt GPIO signal level is *automatically* placing itself on an unpredictable count-down to expiry. Unpredictability in standards means uncertainty. Uncertainty means doubt. Doubt means nobody will use it.

*The Lesson: have the CPU Card supply a GPIO Reference Voltage, as part of the standard, and have Level-Conversion (if needed) handled by the carrier boards*.

This lesson turns out to be something that the industry already knows and has supported for decades. Looking at nearly every I/O chip out there, time and time again it can be observed that there are typically more than one power supply pin. One is usually for the chip's "Digital" circuits, another for its "Analog" circuits (if there are any), and, invariably, yet another is for its *I/O circuits*. This last voltage is where the "GPIO Reference Voltage" comes into play. CSI Camera modules require anywhere between 2.8v and 3.3v for the Digital and Analog supply, but the I/O voltage can be anywhere between 1.8v and 3.3v. The SN75LVDS83b can have an I/O voltage of anywhere between 1.8v and 3.3v. I2C EEPROMs are designed to take an I/O voltage of between 1.8v and 5.0v. So this is, far from being a "new concept", actually an *Industry Standard Practice*. Recognising that and making it part of the standard now makes sense, as does the definition of CMOS logic levels being 0.3 times and 0.7 times the Reference Voltage.

Finally, we have the issue of choosing which standard interfaces should make up, as a whole, the standard itself. Should USB be chosen? If so, which variant? What about SD/MMC? What about Ethernet, or SATA? What about audio? and the hardest to choose (and justify) is which type of video output should be included. For EOMA-68, the first draft of which was drawn up in 2011, it has taken *four years*, along with the analysis of almost a hundred different SoCs, to settle onto the chosen interface set.

One key factor which allows the EOMA68 standard to be truly called "future-proof" is in the difference between upwards-negotiation at the hardware level and the failed concept of "optionality" (as discussed above). Take SATA as a potential candidate. SATA is 4 wires: two differential pairs. However, at the hardware level it has upwards-negotiation, from the lowest speed of 1.5gbit/sec for SATA-I, right the way up to 6gbit/sec for SATA-III over the *same four wires*.

Another (better) example would be PCI-Express or SD/MMC. Both of these standards require a "lowest level" - a minimum number of wires. PCI Express for example can operate at a single lane, and SD/MMC can operate with only a single wire for data (in "SPI" mode). However, both these standards can, if the *hardware* detects extra lanes, *automatically* upgrade to (or more specifically: negotiate the use of) the faster speeds. In other words these standards were *designed* with removable plug-in cards in mind, right from their very inception.

The critical thing here, however, is that if such standards are used, it is imperative that the CPU Card provide support for all speeds of any given standard, and all those speeds below, up to the maximum available. The reason is simple: the "Carrier Boards" cannot expect to have the cost of any potential "conversion" imposed on them. Imagine for example that the OMAP3530 was deployed (allowed). The OMAP3530 has a very strange USB2 interface which *only* supports the 480mbit/sec speed. It cannot support the slower speed devices. That would mean that just because USB 1.1 might not be properly supported *EVERY SINGLE CARRIER BOARD* would be forced to provide "USB1.1 to USB 2.0 conversion" just to cope with the fact that someone *might* plug in a CPU Card with an OMAP3530 processor on it. As that is totally unacceptable as it would impose a significant extra cost on the BOM of every single carrier board product, the standard *has* to set the rule that the CPU-side must support all speeds in between the arbitrarily-chosen maximum.

One final flaw that was spotted in at least one of the standards (96boards) was the provision of triple-function pins. This is an absolutely fatal and final nail in the coffin of any standard. Basically it dictates that each and every SoC utilised *must* provide those *exact* same three functions, because the cost of providing external bi-directional multiplexing components on the CPU Card is so high as to make the entire exercise utterly pointless.

Multiplexing has become pretty much a way of life in the embedded SoC space. Silicon ICs are manufactured by placing the bare die onto a tiny PCB, where the PCB is then dropped into a plastic (or a ceramic) case which has metal pins sticking out of it. However, before the lid is put on, there is one final task left: connect the pins to the PCB. This is typically carried out, in BGA pins by *literally* hammering the pin onto the PCB, using the heat caused by the friction generated by such a tiny area being hit so hard as a form of "welding". The problem is that such hammering can often cause cracks in the PCB or the silicon die. The more hitting, the higher the chance of failure. Thus, logically, the more pins, the higher the cost of the resultant product (due not to the actual cost of production but because of the increased number of failures!)

Also, where space is a premium on these smaller embedded devices, it would at first seem to make sense to make the pin pads as small as possible. half a millimetre or less, even. It turns out, however, that when you do this it's simply impossible to cost-effectively manufacture the PCB with such tiny spaces. Many China PCB manufacturing and assembly factories simply can't cope with anything lower than 0.8mm. Their equipment isn't accurate enough (or the BGA ICs are placed by hand). At 0.8mm, when the BGA balls (which are made of solder) begin to melt, the surface tension to the copper pad and all of the BGA solder balls pulls the entire IC into alignment. However, below 0.8mm the chances are high that even the slightest misalignment will result in two BGA balls melting and flowing together.

So, basically, to keep both manufacturing price *and* the size down, the number of pins needs to be no more than around 450 pins, the size of the processor no more than around 20mm on a side, and the pin "pitch" as it's called absolutely no smaller than 0.6mm. This in turn means that many of the functions offered by an SoC need to be "doubled up"... or, in the case of some Texas Instruments SoCs, "octupled up" (eight possible functions per pin).

The important thing to note here, however, is that even amongst the same offerings from any one SoC manufacturer, *there are no multiplexing standards*. So to expect any two SoCs to provide even two identical I/O functions on the exact same pin is just utter madness. About the only de-facto standard that seems to have been agreed on is that one of the functions of a pin will always, always be a plain "GPIO" - General-Purpose Input/Output. It's worth emphasising however that that *doesn't* mean that any one GPIO pin will be "Interrupt-capable" (see GPIO section, later, for details). EINT, it turns out, is often treated as being one of the multiplexing capabilities.

*The lesson: under no circumstances offer more than two multiplexed functions per pin, where one of which MUST be GPIO.*

With these additional technical lessons as requirements in place, we can finally choose the interfaces for the example standard, EOMA-68.

## Example standard: EOMA68

EOMA68 is the first EOMA standard, which reuses the legacy PCMCIA casework, housings, sockets and assemblies. It has only 68 pins, over which it is particularly challenging to ensure that useful functionality is provided. With care - and luck - it has proven to be possible. The nice thing about reusing legacy PCMCIA is that, firstly, the

tooling does actually still exist, and secondly, it's big enough (exactly credit-card-sized for very good reasons) to be able to fit an entire fully-functioning computer without having to make too many compromises on either space or features.

The general idea behind EOMA68 (and indeed all the EOMA standards) is that you literally fit the *entire* fully-functioning independent computer onto a removable, robust "card", thus turning devices (such as laptops, tablets, desktop computers, Digital SLR Cameras, Media Centres and so on) into nothing more than "Peripherals". For those people who are familiar with the Motorola Atrix "Lapdock", which has been famously hacked to run with low-cost "USB-Stick" style computers for around a total of USD 120 (including the cables), it's a similar concept, with the difference being that all EOMA68 compliant "docks" are *designed* from the very start to accept all and any third party CPU Cards instead of just one single manufacturer's proprietary product.

*Compact Flash is a "Memory Card" standard, compatible across multiple third party products, so why not a "Computer Card" standard?*

Let's summarise the requirements here.

- The standards chosen must not be encumbered by patents or monopoly practices.
- They must be fully supported by Libre-licensed Software
- They must be long-term, both in the past and in the easily-predictable future.
- If necessary they should have some form of flexibility (speed negotiation)
- They should ideally come from a "removable" plug-and-play background
- No GPIO pin should provide more than two functions (including GPIO itself)

Additionally, the following features need to be covered or addressed:

- Audio
- Video (in and out)
- Storage
- General-purpose I/O (including interrupt capability)
- Analog (ADC / DAC) has to be considered
- Peripheral ICs (sensors etc.)

Covering each of these in turn we can assess how they may be provided.

## Audio

This is one of the trickiest to provide. Analysing dozens of different SoCs, there simply are no standards. I2C might be considered to be an acceptable standard, but there are two variants (5 pin and 7 pin), and what about the Audio ICs at the other end? Some might not work if they are expecting to receive more data than the available pins. Additionally, some of the ultra-low-cost SoCs simply do not have I2C (at all). Others provide audio via analog on-board ADC and DAC Audio circuitry.

In the end, the decision was made to not include audio as part of the EOMA68 standard, but that it could be provided as a USB Audio device. In the cases where the latency normally associated with USB2 audio was unacceptable, there is always the end of the PCMCIA card, where audio connectors as well as HDMI (which includes audio) can be placed.

## Video

This, again, was one of the trickiest to consider. Should "video in" be considered to be part of a general-purpose computing standard? Given the number of wires in, for

example, CSI, and given that MIPI is so new, the decision in the end had to be "no", but that it could be provided if desired over USB (or at the end of the CPU Card).

Video output was much tricker. An analysis matrix had to be considered, based primarily on the resultant BOM (cost of the chips), with the video output support ranging all the way from 320x240 (or even less) for really small hand-held devices, right up to 1920x1080 @ 60fps for large-screen displays. The reason why the BOM had to be taken into consideration as the primary driving factor is that at the lower-cost end, the inclusion of any "video conversion" chips has a far-larger impact on the price, percentage-wise, than it would at the higher end.

So the decision had to be that the interface types that were available at the lower-end of the spectrum had to take priority. With that firm decision in mind, and using the awesome and very recent resource, http://panelook.com, it was very easy to determine and justify that RGB/TTL had to be the interface of choice.

Through http://panelook.com it was possible to ascertain, through the number of active panels at the lower resolution end of 320x240 up to around 800x480, the most commonly-available interface type was RGB/TTL. LVDS, MIPI and so on simply were not in the running, at all, at those resolutions. By around 800x600, single-channel LVDS starts to come into play, up to around 1400x900. MIPI and eDP are still an *order of magnitude* less active (less available in production) than even LVDS, despite word to the contrary from the sources that wish to tell you what is going to be the latest-and-greatest standard. However, at these higher resolutions, the cost of the LCD itself (USD 30 to 65) dwarfs the cost of a conversion IC (USD 1.50 to USD 5), so above the 800x480 threshold it really is immaterial as to what ultimate LCD interface is utilised on the carrier board. Thus, at the 800x600 range and above, the LCD may be arbitrarily chosen for any carrier board product based on current price and availability (remembering though to include the cost of the converter IC in the assessment).

Again, however, a 2nd (or even a 3rd) Video Output could (and has, in one of the Reference Designs) be provided in the form of HDMI or CVBS connectors, via the user-facing end-plate of the PCMCIA casework.

Also an issue that needed to be decided was based on the capabilities of the lower-cost SoCs. A decision was made to restrict the maximum resolution, currently set at 1366 x 768 (but may be increased to 1440x900), so that the USD 2 IC3128 and the USD 2.50 Ingenic jz4775 could be used. However, this would appear to limit higher-end SoCs which are capable of operating at 1920x1080 resolutions, so this was reserved for a future (backwards-compatible) upgrade, using the physical height restriction of PCMCIA "Type II" Cards at a 3.3mm height.

The only remaining question then is to decide how many pins (colour resolution) are to be provided: 24-pin, 18-pin or 15-pin. Initially, in 2010, the full 24-pins were provided on EOMA-68. However, as the standard developed and more alternative interfaces were needed, this was reduced to 18-pin, on the justification that many low-cost LCD panels only provide 6-bit colour anyway, and that many people cannot tell the difference in brightness levels below 1 percentage point. Remember: the rule is to provide *good enough* computing. It turns out that many mid-range LVDS panels (especially the 1024x600 and the 1366x768 ones) only have 6-bit colour resolution anyway (3 lane LVDS, 6 bits per lane), so going beyond that would be pointless anyway.

## Storage

Initially, storage was either specified to be provided via on-board NAND Flash, on-board SD/MMC, or via USB-connected devices. In the initial version of EOMA-68, SATA was also included, but as the cost of SoCs which did not include SATA kept on dropping (to

as low as USD 2 for a quad-core processor), inclusion of SATA (which would require an additional USD 4 in components to provide, when the SoC itself was only USD 2!) became harder to justify. SATA was therefore dropped in favour of adding support for USB 3.0, in 2012.

Around the same time, however, it was decided to make it possible to connect an SD/MMC device over the EOMA68 interface, with up to 4 lanes. This could either allow for storage using eMMC, or for an additional SD/MMC storage card slot, or even for any of the various SD/MMC compatible peripherals to be connected (such as the various embedded WIFI modules).

So with the inclusion of USB 3.0 (which goes up to 5 gigabits per second whereas SATA was up to 6 gigabits per second), high-speed storage needs are taken care of, if the SoC has USB 3.0 capability. With USB 3 being backwards-compatible with USB 2 all the way down to USB 1.1 and 1.0, even the lower-cost SoCs still have access to external USB storage, albeit at slower speeds. and that's fine too: people pay what they can afford, and they can always upgrade later.

SD/MMC itself has an interesting development history, spanning a couple of decades. It started life out as a 4-wire bus (strictly speaking an enable / select line plus 3 wire: clock, data in, data out) that got expanded to 2, then 4, then 8 bi-directional data lines, and along the way added auto-negotiation for increased data rates, in a similar way to USB, SATA, PCI, PCIExpress and many other standards which involve backwards-compatibility as a core fundamental feature of their design.

This backwards-compatibility - aside from being something that the EOMA68 can take strategic advantage of - is critical for end-users of SD/MMC storage (and peripherals) to be able to access their data over time, both from the perspective of purchasing an SD card and using it for 5 to 10 years in a wide range of both old and newer devices, as well as being able to buy a product which takes SD cards and not upgrade it for 5 to 10 years. If there was no auto-negotiation at the hardware level *none of this would be possible*.

Only very recently has SD/MMC 4.0 had backwards-compatibility for the original SPI removed, leaving a small conundrum for EOMA68, in the form of whether to support SPI or not. In the end, it was decided "not", for simplicity, and to add a "proper" (explicit) SPI interface to EOMA68 on separate wires... but to hint that SPI, being only 25 mbits/sec and no greater, can easily be implemented as "bit-banging" on any arbitrary GPIO of the interface. There is even a Linux kernel module which does exactly that.

## GPIO

As previously hinted at, GPIO is where some care has to be taken, surprisingly, for something that seems so simple as communicating "on" or "off". Firstly: GPIO is often used to signal "interrupts", as in a notification that an event has occurred. For example, an EINT may be generated by a peripheral that was previously in a special low-power mode that it has woken up, and that the main CPU should now talk with it to extract the data that it now has available. This saves power both on the peripheral as well as the main CPU... but only if the CPU has "EINT wakeup" capability.

Such capability can be quite costly to provide on a CPU, so it is often a limited resource. It is therefore not wise to allocate every single GPIO as being "EINT". The Allwinner A20 SoC for example, although it has over a hundred GPIO pins, only has 30 of them marked as "EINT" capable. With many of those being multiplexed to multiple functions, it seemed prudent to only allocate a maximum of 4 of the GPIO pins as "EINT" capable.

One slightly strange GPIO pin is also offered: a Pulse-Width Modulation (PWM) GPIO pin. This pin needs some explanation. PWM can be quite fast, but also regular. If it was to be provided using a software timer, the amount of CPU resources needed to "merely" change the pin from 0 to 1 and back again at fixed, regular intervals, would be disproportionately considerable to the simplicity of the actual task. Given that this standard is to be used in portable devices, where the LCD backlight's brightness is regularly controlled by exactly such a "PWM" signal wire, it seemed prudent to add one PWM pin to the EOMA68 specification.

The only other main issue, as already discussed above, was to have the CPU Card specify a "Reference Voltage" that the GPIO must comply to. It turns out that many if not most peripheral chips (such as the Texas Instruments LVDS converter IC) actually have two (sometimes even more) power lines: one for the main power, one for handling of digital GPIO, and sometimes there is a third (separate) power supply for analog sections of the IC. So the anticipated "level conversion" which would be expected to be provided on the carrier board actually turns out to be a non-issue, because the peripheral ICs often turn out to be designed with variable voltage supply in mind. That Texas Instruments SN75LVDS83b IC for example takes a 3.3v Digital supply for its main functioning, but the GPIO section can be powered directly from the "Reference Voltage", allowing a direct match of the voltage levels with those coming from the CPU's GPIO. The STC3115 Battery Monitoring IC also follows the same concept, and the various I2C EEPROMs available on the market also seem to follow the same de-facto standard variable power level protocol.

## Analog ADC / DAC

Whilst it was considered to be nice to have, in the end there were no scenarios that could be envisaged where it would be beneficial to have on-board ADC or DAC on EOMA68. There exist ADC / DAC peripheral ICs (usually SPI-based to achieve a faster data transfer rate), and there exist Embedded Controllers with on-board ADC and DAC channels that can also emulate USB client interfaces and/or have SPI and/or I2C, some of them for well under USD 1, which would perfectly well provide the job as needed for any specialist ADC or DAC tasks.

## Peripheral ICs (sensors, etc.)

As previously mentioned above, USB is the "interface of choice" for high-speed external (and internal) peripherals. Audio, WIFI, printers, keyboards, mice, external storage: there even exists USB Display devices so that extra screens can be connected to a computer.

However in the embedded world there are literally tens of thousands of small sensors available, which use less well-known interfaces such as I2C, SPI, One-Wire Bus, CAN Bus, RS232, RS423 and so on. Out of these choices, the most commonly-used ones in the "embedded" market were chosen to be provided: I2C, SPI and RS232. I2C because it is a multi-decades-long simple and proven standard used for temperature, gyroscopic, accelerometer and thousands of other sensors. SPI because it is another decades-long standard that is used for slightly higher-speed data collection than I2C, and RS232 because there are a wide range of GSM, GPS and other Radio devices that still use RS232.

USB, it is worth emphasising, has a very long history, of over 2 decades of development and improvement. The latest improvements, from its humble beginnings of only 10 mbits per second, bring it up to 10 Gigabits per second in the USB 3.1 dual-lane scenario. It is hard to envisage, over the next ten years, which kinds of "good enough computing" scenarios could not possibly be covered by such insanely-fast data rates.

# Summary of Interfaces

So EOMA68 has the following interfaces:

- 18-pin RGB/TTL video output
- One set of wires that provide anything from USB 1.0 all the way to USB 3.1
- A second USB interface that is from USB 1.0 to USB 2.0
- One SD/MMC interface (which may go from 4-pin all the way down to SPI)
- One I2C interface
- One RS232 TTL-compatible UART interface
- One SPI Interface
- One PWM GPIO
- Four "EINT" capable GPIOs
- One "power / reset button" line
- Up to 22 GPIO pins (many of them multiplexed with the above functions)

This is actually quite a versatile standard with an extremely low barrier to entry (even the USD 2 IC3128 from ICubeCorp is capable of fulfilling these requirements, as is the USD 3 Ingenic jz4775) that can accommodate much faster SoCs including Quad and Octal Core offerings from Samsung, Allwinner, and even Intel. In particular, as the memory and main boot storage is *on board* the CPU Card, there are no complex high-speed memory bus requirements needed.

Thanks to the interface selection, design of the carrier boards is a very straightforward and simple prospect. Many of the Reference Design carrier board PCBs are 2-layer with components on only one side. All high-speed signals are differential pairs. All the complexity is in the CPU Card *not* the carrier board, making it an attractive option for low-cost product prototyping.

At the time of writing, the most recent (and almost certainly final) update to the EOMA68 interface has been the removal of Gigabit Ethernet in favour of extending the USB 3 support up to USB 3.1 capability. This it is felt is important to ensure that EOMA68 be the long-term viable standard that it is designed to be. However, it is worth emphasising, again, that it has been *four years* since EOMA68's first draft. If at any time it had been decided that EOMA68 was "ready" prior to this final revision, the lifetime of the standard would have definitely been adversely affected.

Sometimes it's best to wait, rather than release early, especially given the nature of EOMA68: once it's released as final, there is absolutely no possibility of even the smallest revision, because it would result in incompatibility between existing carrier boards and CPU Cards vs the newer ones. Whereas in the industrial world, the small volumes of an individual factory's needs may be met by upgrading standards (PC-104 now has half a dozen upgraded variants), a mass-volume standard has absolutely no such luxury of "upgrade-in-the-field". It's either right, or it's an immediate failure.

The only opportunity to provide a (one-way) backwards-compatible upgrade path is by using the PCMCIA "card height" as a mechanical barrier. This method has already been scheduled to provide 1920x1080p60 LCD interoperability at the 3.3mm height level, on the basis that any SoC which can do 1920x1080 over the RGB/TTL interface can also fit into a 5.0mm height socket and support down to the minimum resolution, but that the other way round is not necessarily true. A 5.0mm high card will not fit into a 3.3mm slot, so the possibility where a CPU Card that can only do 1366 x 768 being incompatible with a device that has a 1920x1080 screen simply does not and cannot (physically) occur.

Within this chosen set of interfaces - none of which are optional - it is guaranteed that future CPU Cards will offer higher-speed capabilities, for less power, and that such CPU Cards, even those produced in 10 years time, will continue to operate even in the very first carrier products that come onto the market. Likewise, it is guaranteed that the very

first CPU Cards will operate (albeit at much slower speeds) in products that hit the market in 10 years time. Even in the unlikely scenario where the standard screen resolution for "good enough computing" morphs well beyond the 1366x768 maximum set by the amazing current USD 2 SoC offerings, there is always the possibility of using a modern future USB Display IC to reach above the current arbitrarily-set limit.

So from this extremely simple and effective interoperability strategy, we begin to see hints that products based around the EOMA68 standard could, in fact, genuinely deliver the goods.

# Scenarios

With the standard defined, it needs some scenarios to test its effectiveness from various different perspectives.

# Software Development

The critical scenario for hardware development is: will the software keep up? Is it still possible to develop software for such a "split personality" scenario? Does it matter? what are the challenges?

The key challenge is that a CPU Card can be put to sleep, removed from one product, then when it is woken up find that its entire world has just completely changed. Everything it knew - from the screen size to the input devices and even its storage devices - has just utterly changed. Fortunately, the entire project can ride, in some ways, off the coat-tails of the prior development that has gone into the software support behind things like USB, as well as the capabilities already provided with "software suspend" features present in GNU/Linux-based Operating Systems. USB devices and GNU/Linux based OSes are *already* capable of dealing with device hot-plugging.

Additionally, behind the scenes of projects like Google Project Ara, there is significant work on software development, to ensure that the Android OS also adapts properly to hardware disappearing and being replaced.

The one area where Google Project Ara will *not* handle is screen-size changes. Very few Desktops actually cope adequately with dramatic changes in screen resolution and size, typically because it significantly impacts the font sizes, icon sizes and so on in ways that the Desktops themselves simply weren't designed to cope with without a major reconfiguration and restart. However, one of the very few Desktop Widget Environments that has been designed with such flexibility in mind is KDE Plasma, and this flexibility extends right the way down to the application level. A KDE Plasma application that began on a 1920x1080 LCD could, in theory, adapts even its menu structures to operate some time later on an 800x480 LCD, without requiring a restart.

It has to be said though that, in general, the changes in LCD resolution are the one weak spot of the EOMA68 strategy, and it can be handled (badly) in initial incarnations with a full reboot of the Graphical Interface, reconfigured to the new screen resolution.

The one thing that does have to be pointed out is that the current hermetically-sealed product industry-wide strategy is, particularly in the China supply chain, absolute endemic with illegal Software Copyright License violations. One software developer who was curious as to the extent of the illegal operation of China product suppliers gave up after the 150th report in his mailbox of yet another illegal copyright-violation. The rate was something like a 98% GPL Software License violations rate. Even large companies like Mediatek, Samsung, LG, HTC, and many other huge brand names, have had to be threatened with court cases (with the resultant impounding and destruction of product at Customs without recourse to insurance claims, due to the criminal nature of Copyright

violation) before they reluctantly commit to fully complying with the GPL Software License.

The risks associated with the criminal act of flagrantly violating Software Copyright are just too great to even consider. Asking distributors to become criminal accessories to Copyright theft - even 3rd party theft - cannot be countenanced either, yet this is exactly what many distributors such as Amazon, E-bay and other Internet suppliers are doing: acting as a 3rd party accessory to criminal Copyright infringement.

To ensure that this does not happen with EOMA68 products, a full Libre Software Development Strategy is being deployed. Not an "open" one, which is typically summarised as "Sure We Will Release The Source Code... Just As Soon As We're Done Developing The Product", but a *truly* Libre Strategy. Source code for products in development is available *now*. Software Developers are invited to participate *now*, and will always be welcome. In this way, there is absolutely no excuse for any product to be released in mass-volume quantities in stores world-wide without the full source code being available even before it hits the shelves, fully license-compliant, allowing end-users to experiment, replace, upgrade and secure their own devices, without restriction, interference or impediment, for years to come.

## Factories

The benefits of an EOMA68 strategy for factories is in being able to split the ordering of components across those that are required for CPU Cards and those that are required for base units. Both sets of components have completely different ordering strategies.

The development of SoCs themselves takes around 18 months, and with the pressure on pricing, any given SoC can expect a "Supernova" style lifetime before it is overtaken - even sometimes in as little as five months - by a competitor's product. The only thing that used to save SoCs from this boom-then-bust cycle was the inherent contractual committment of the supply chain. Components have already been ordered, customers already have committed deposits, the whole process takes a few months so surely there is room to get the products sold and out the door?

However, all that ended, around 2010, with a major recession in the China Shenzen Electronics arena, caused entirely by the disruptive influence of a USD 7 SoC from an as-then little-known company called Allwinner. The previous most disruptive SoC was an USD 12 ARM11 processor from Korea, around 2008, where its closest competitor was from Samsung at around USD 16 and a MOQ of 50,000 units. This was quickly replaced by a USD 11 ARM Cortex A8 processor from a company called AMLogic. Several tablet and IPTV manufacturers sprang up based around these phenomenally-low-cost SoCs, but in 2010, when Allwinner came up with a BOM for a tablet PCB of only USD 15, including USD 7 for their new SoC, everything went completely bananas within a very short time. Nobody wanted to buy USD 50 tablets when USD 35 tablets were suddenly possible. Clients cancelled orders of tablets based on the AMLogic SoC, *despite having put down large deposits*, so, with 100% of their clients reneging wholesale en-masse on their contracts, the Design and ODM companies went bust. Consequently, suppliers who had not only bought the more expensive AMLogic SoC but also bought the various support components associated with the Reference Designs that the various tablets and IPTV products were based on found themselves holding stock on credit, with no buyers and no way to shift the stock either, so they were forced to go bankrupt, too. All in all it was a pretty disruptive time that took about 18 months to stabilise.

However, this situation could have been entirely avoided, with a split (modular) architecture. With the SoC being boxed inside a CPU Card with its own memory and storage, the fact that the CPU Card can be used in several compatible devices means that it can be ordered by a factory for huge bulk discounts, assembled and then sold as

an upgrade for all pre-existing devices as well as new ones coming onto the market. There are benefits here for the Fabs (foundries) as well, because the larger the order, the more time can be dedicated to ensuring that the chips are produced reliably. In simple terms: the larger the bulk order, the higher the yields from the foundries, which translates itno a decreased price. And if a single CPU Card can be huge-bulk-ordered by a factory, knowing that it's going to sell as an upgrade to dozens of products world-wide, it's a simple logical deduction to predict that a modular strategy *will* result in price reductions.

Then there are the devices that the CPU Cards fit into. The components on each device *simply do not change* - not unless one of them goes "End of Life". However, if a factory commits to a 5 year long-term regular buy, the component supplier can plan for a much more regular supply as well. Plans can be made to replace tooling on a regular basis instead of having to write-off (or mothball) perfectly good equipment just because, unpredictably, another fabless semiconductor design company began yet another "Supernova" SoC.

This is a drastic improvement on the current scenario, where the anticipated committment to supply is entirely and arbitrarily dependent on what components the SoC manufacturer selected for their Reference Designs! Especially given that the design basically doesn't get changed to any significant degree, but the lifecycle of the SoC is, as observed above, on a "Supernova" cycle: briefly glorious but the moment something better comes out, even from the same manufacturer, it's dead in the water.

So the EOMA68 modular strategy has clear stabilising benefits on the whole product supply chain, all of which end up indirectly as cost savings for end-users as well as less sleepless nights for Factory Management.

## Sales Teams

Current sales strategies are based around brand and around ever-increasing speeds, screen resolutions, and battery life. It's not so much a strategy as a rote formula. People have given up asking "can it be repaired" after the first broken screen on their smartphone, when the salesman responds with "did you buy extended warranty insurance from us?", or "would you like to consider a new contract and a new phone?"

From an environmental perspective, this isn't exactly encouraging, but a good salesman sells what he's given. Would it not be nicer for them, however, to sell something that would result in repeat long-term customer commitment not because the customer has no other choice to replace an entire hermetically-sealed product because it was *designed* not to be repaired or upgraded, but instead because the sales pitch was honest, helpful and, not only that, was *actually truthful*?

With modular products, a salesman can honestly say "Buy any of these CPU Cards, and any of these compatible products, and they will 'Just Work'. I guarantee it. And you can come back any time and upgrade." The truly honest salesman will, quietly, in a sotto voice, add "but if you buy the low-end CPU Cards you get what you pay for if you know what I mean", because these will likely be the ones where the 2nd USB interface, thanks to having a USD 2 SoC instead of a USD 7 one, is in fact only USB 1.1 speeds, so results in a significant slow-down of any peripherals connected to that port.

The salesman can also say to you, "if you break the screen out-of-warranty, we can loan you a spare base unit with a deposit, you just pop out your own CPU Card, put it in our spare, and we can take care of the repair for you. You won't lose any of your data, or your apps, because they're all on the CPU Card, not the base unit going off for repair".

The salesman can also say, "As you're buying our top-of-the-line models we can offer you a discount on this entry-level tablet base, that way you will have 2 base units you can share one CPU Card, you keep all your apps and data consistent, and you save over 40% on the cost of buying an equivalent Chromebook plus say a Samsung tablet."

And if the customer is a regular visitor to the shop, the salesman can say "I've got some upgraded CPU Cards with the latest Android OS on it, if you'd like to try one out next week, bring your laptop and your tablet base in with you. I guarantee that you can just pop out your current CPU Card, try out the new one, see how it runs, then if you don't want to buy it just press the button, pop out the new one for me and put your old one back in, with all your apps and data completely unaffected. And the new CPU Card costs less than the one you bought from me six months ago, and is faster, too. You can always keep the old one until you're confident that all your data and apps work on the new one. I do a good rate, better than ebay, on 2nd hand CPU Cards now, by the way, so I'm happy to give you a good price for the older CPU Card when you're done with it. See you next week?"

For the level of service and customer loyalty that results, these kinds of sales pitches are literally worth their weight in gold. And they're quite simply flat-out impossible with the hermetically-sealed products that are *deliberately* designed - from the ground up - to get you to buy an entire new product in the shortest psychologically-acceptable amount of time. Nobody really wants to buy used tablets or smartphones with broken screens or worn-out batteries (except for people who take them apart then sell the components 2nd-hand on ebay), and they certainly don't want to buy hermetically-sealed malware-ridden products where the OS cannot even be upgraded or replaced due to Copyright violations by the manufacturer.

## End-users

Some of the benefits to end-users have already been hinted at in the section covered by Salesmen, but it helps to illustrate things from the user's perspective as well.

## Families

2007 was when the author first heard about the Skytone Alpha 400, because a friend - a single mother with two young daughters - had just bought two of them as Christmas presents. The devices were utter disappointments, because the children were expecting something that would be capable of running their Flash and Java-based BBC entertainment Internet web sites. The Skytone Alpha 400 was based around a 350mhz China-home-grown MIPS compatible SoC, with 128mb of RAM. This was how it was able to meet a pricetag in the UK of only 95 pounds. However, Adobe demands an up front license fee of USD 50,000 from *individual distributors* - not the manufacturers - to provide the Flash Plugin for web browsers. In other words, every single one of the worldwide 30 or more OEM branded distributors of the Skytone Alpha 400 would have had to pay Adobe that USD 50,000 cash payment, just to be able to distribute the exact same Flash plugin. Maplins: USD 50,000. Argos: USD 50,000. Walmart: another 50,000. Best Buy: USD 50,000. for the exact same binary.

Naturally, this did not happen, with the end result that the Skytone Alpha 400 went out without Flash support, cutting off access to a large fraction of the Internet, including the BBC's Flash-based games and entertainment.

Sadly it was not until Steve Jobs encountered the exact same intransigent attitude presented by Adobe, when the first ARM-based Apple products were developed (the first being the IPAD 1), that the situation began to change. Steve would have flat-out refused to pay Adobe, especially given that the Intel (x86) Flash plugin is developed and released by Adobe without any such license demands, but Steve's huge favour to the

world's Internet Community (by beginning the end of Flash's dominance) was half a decade too late for the China-home-grown Ingenic MIPS-based products such as the Skytone Alpha 400.

The question has to be asked, then, how the situation would be different if this amazing little clamshell laptop had been an EOMA68-compliant device? It would have been a completely different story. The initial excitement of those children would have quickly turned to disappointment, but, crucially, their mother would have been able to mollify them and promise them, within a few months (as she earned a bit more money) an upgrade, knowing that the cost of an upgrade would, this time, not only be a bit better researched but also would be nothing like the cost of replacing the entire device. Perhaps she could have approached friends or relatives to ask if they would like to buy an early birthday present. But with the monolithic design strategy which gives us a lower price-tag but no upgrade possibilities, that option was sadly out of reach to her as well.

Whilst we cannot change the past, we can at least learn from it, by envisioning a better future. In this better future, those children received upgradeable computers for Christmas. Six months later, for a fraction of the initial purchase price, Grandmother did indeed purchase upgraded CPU Cards - ones that were capable of running their BBC entertainment.

Then, a few years later, the children enrolled in a new school, where they were required to have a laptop in order to complete their schoolwork. So, their mother purchased a *single* larger (9in) base unit for each of them to share, between their two CPU Cards. When not doing schoolwork, at least one of the children was able to use their original clam-shell micro-laptop. The cost-savings as well as the privacy afforded to both children by having their own separate CPU Cards are pretty clear.

We can take this story further if the two children go to University. There, they upgrade the old CPU Cards to much faster devices, saving money again, and one of them enrols in a photography class, where a larger screen is needed to display (and edit) the photographs. Then, interestingly, a new product comes out: an EOMA68-compliant Digital SLR Camera, where the CPU Card plugs directly into the side of the Camera. So, instead of removing the *memory card*, the user removes the *entire computer*. Or leaves it in place, where, at home or at the University, she simply plugs in a USB keyboard and mouse and a large screen into the USB and HDMI ports on the CPU Card, leaving the CPU Card in situ in the Digital SLR camera, and operates the editing (and emailing) software directly, with the computer (and the editing software) being entirely powered by the Camera's internal battery.

Later, this young girl gets a job as a newspaper photographer. She buys a 3G / LTE - compliant CPU Card which has a built-in 3G / LTE modem, and plugs it in as a replacement for the CPU Card in her Digital SLR Camera. As a result of this unusual combination that was never envisioned as being possible or a sufficiently likely product to justify the R&D investment by either the manufacturer of the Digital SLR Camera or the 3G/LTE CPU Card manufacturer, she is able to capture - and instantly email to her editor - footage from a spontaneous riot. Protected by the fact that the footage is available publicly in real-time and that the feed is picked up by the BBC and other world-wide news, the Police (who would normally have illegally attacked her, confiscated her equipment and imprisoned her) leave her alone. Indirectly she is responsible for saving lives, reducing both Police and citizen brutality, captures critical video and photographic evidence that results in the prosecution of criminally violent individuals who began the riot, and for her efforts she wins a prestigious journalism award.

The key here is that this is a highly unusual combination of two products from wildly disparate manufacturers who would, under normal circumstances, never collaborate to create such an unusual product as a real-time 3G/LTE-capable Digital SLR camera, because it would be viewed as a "high risk" product by the management of not just one

but *both* companies, not least because the software development alone would be well outside of the expertise of both organisations.

However the much simpler prospect of developing a simple USB-style Digital SLR Camera "base", on top of which is provided a simple "Camera app" which may be moved from one USB Card to another as the end-user upgrades to newer CPU Cards, turns the SLR camera from a "hermetically sealed appliance" into a much more interesting and ubiquitous product, with huge creative potential.

So we began the story with a family saving money by being able to share base units between CPU Cards whilst respecting family members' privacy, and ended up with a fascinating and unlikely combination of two products from completely different manufacturers on totally different sides of a single common modular hardware standard.

## A day in the life

Another way to envisage the benefits of modular computing is to follow a single CPU Card as it travels through one person's day, from device to device.

Imagine our journalist from the previous scenario: they're woken up by an alarm app running on the CPU Card, plugged into an Internet TV in their bedroom. Going downstairs to breakfast, they move the CPU Card to the kitchen TV, which, from software, triggers an automatic "breakfast TV routine" that includes showing the day's schedule, as well as offering the opportunity to download some music for the journey to work.

In the car, the CPU Card is plugged into the Sat-Nav, where it has already received the itinery from the email received the night before, and begins to give out the usual advice. With 3G / LTE downloads still being expensive, our journalist is grateful for the fact that the CPU Card has a huge storage capacity (actually a fast 256GByte Micro-SD Card), where the music for the day was downloaded using the house Broadband, and the ultra-high-resolution originals of the photos from the evening's shoot were also likewise uploaded to a work server overnight, over the same fixed-price house Broadband / Cable.

Arriving at the first photoshoot, the CPU Card goes into the Digital SLR Camera. Previews of the high-resolution pictures are reduced to a more 3G-uploading-friendly resolution and sent immediately to the Editor, who offers comments that appear directly on the Digital SLR Camera's 4in LCD. After the photoshoot, she drives to the Office, pops out the CPU Card (instead of the memory card), and puts it into a huge desktop unit with a much higher resolution screen. The editing suite software is already on the CPU Card, so she can get straight to work.

However, as it is very new, one of the senior editors calls "dibs" on the desktop unit. No problem: our journalist pops out the CPU Card, puts it into her laptop base, and carries on working, whilst the senior editor puts *his* CPU Card with *his* work into the exact same base unit, and gets to work.

At home, she flips between different forms of entertainment: a game she likes playing is easier with a hand-held games console base, but an important email comes in: she deals with it by plugging the HDMI TV in to the CPU Card's HDMI port and plugging in a keyboard to the console's USB port. Then, she likes the larger screen size and carries on playing the game on the TV. The game adapts to the larger resolution and provides a higher level of detail than the smaller LCD offers. Later she watches a film... through the same Games Console base, and skypes her sister. Easier to leave the CPU Card in the Games Console base: it's all the same software, after all. No need to make life complicated.

Can you imagine the sheer overwhelming cost of all these devices if they were all "monolithic"? Add up the number of devices and multiply that by the cost of CPUs and Memory. Not to mention the utter waste of all the screens (and CPUs) sitting idle on desks or in homes (very few people are actually capable of productively using more than one computing device simultaneously, and they certainly don't like sharing them, due to the privacy implications as well as the potential for data-loss). And the software development costs of duplicating such a scenario (most likely with some privacy-invasive but also extremely bandwidth-intensive and always-on "Cloud-based" attempt at a "solution") if there were multiple independent CPUs and associated storage in each of the equivalent hermetically-sealed products?

So with this scenario we demonstrate a greatly-simplified way to provide that elusive goal where the computer *usefully* and reliably adapts to your needs, in a way that respects privacy at the same time, at a fraction of the cost that a monolithic design strategy could achieve. We show how base units can be shared, at work, without interfering with either user's privacy or work-flow. Finally we show how versatile the CPU Card concept can really be, with a hand-held Games Console base unit adapting easily to the role normally fulfilled by at least four entirely separate (expensive) products: a dedicated hand-held Games Console, a full Desktop Tower, an Internet TV *and* a tablet.

## Remote Tech Support: cost saving on shipping.

For instances where people help family members or clients, a broken machine or even just a software problem can mean shipping costs or on-site maintenance that far exceeds the value of the computing devices that are no longer functional and well beyond the ability of the family member or client to fix.

For example: a computer that is installed in a shopping centre may require a "Health and Safety" evaluation prior to permitting the technical staff on-site. In the UK, the cost of these evaluations are typically $USD 600 and can take several days. However, a Security Guard or the Manager of the premises could in certain circumstances be exempt from such strict requirements: ordinarily however a "monolithic design" would be beyond their technical capabilities, making the possibility of cost savings and time savings a completely moot non-opportunity.

In the case of an EOMA68-compliant machine, it does not take any technical expertise to push the button and remove the Computer Card. This task may be carried out by anyone. Even if the device is partially functional a replacement Computer Card (weighing 40 grams) may be shipped to the destination and may be replaced on-site easily and within seconds. The advantage of this scenario is that if the system that is broken is even partially-functional there is very little downtime.

For those people helping friends and family remotely, the proposition is very simple: the cost of shipping a laptop for repair and investigation could cost $USD 50 (one way!), when an *entire new Computer Card* could cost around the same amount. With the additional risk of destruction and damage to a laptop during shipping, preparing and then sending just a replacement Computer Card is hands-down the clear economic winner.

## Educational Establishments

Hinted at in the above scenario is the possible cost-savings for educational establishments. There is more, however. One of the major costs for educational establishments is the software licensing of server infrastructure to manage "logins" for students, in order to keep their work private and separate. With each student literally owning their own Computer "Card", the need for logins - and a centralised server - is entirely redundant. The *computer* becomes their login identity, not only at the school but also at home. Small enough to be be transferred from desk to desk unobtrusively,

powerful enough to run Internet, Word Processing, electronics experiments, video and much more, the student becomes responsible for and empowered by the ownership of their own device.

There is also the issue of cost, as well as that of software. Many people will have heard of the failed deployment of IPad tablets to both teachers and students as a substitute for books or other equipment. Such experiments are a costly and completely inappropriate mistake, given the commercial nature of the Apple Corporation, with its focus on using its platform to drive people towards paying for its commercial services (iTunes and more). The general feeling with such devices is that whilst you paid for it, thanks to the proprietary software you certainly don't feel like you actually own it. Why is that an appropriate lesson to demonstrate to school children?

Instead, would it not be better to provide students with their own individual low-cost CPU Card, to have desktop or other experimental educational base units in each classroom, which, if they become damaged through heavy usage, are replaceable at a significantly lower cost than that of a standard desktop or other system?

And would it not be better for the students as well as reducing the cost for schools that the software running on the CPU Cards is not only entirely libre-licensed, but is commonly available and easily downloadable onto Micro-SD Cards that the CPU Card could use to re-flash itself to a known good state, in the rare cases where it becomes corrupted? No more "Calling In The IT Contractors" (at a cost far greater than that even of entirely replacing the CPU Card itself), just put in a MicroSD card and one minute later the CPU Card is once again useable by the student.

Also, there is the issue that some students may come from low income families. If we may assume that they have access to a television with an HDMI port, some CPU Cards may actually be powered over USB-OTG, whilst also allowing for a USB keyboard and mouse to be plugged in. So aside from the cost of the cables (around USD 3) and a USB-OTG Charging Hub (around USD 5), some CPU Cards actually act as wholly-independent Desktop Computers in their own right.

And, as a last resort, with the appropriate government funding, would it not be nice to have, in Libraries, an EOMA68 Desktop "Dock" with fast Internet access, where students who do not have either a TV or Internet Access at home could come in, plug in their CPU Card and do some work without charge... and, crucially, being confident that because they are using *their* own CPU Card, there is zero chance of their work being destroyed by other people's irresponsible and/or accidental malware infections (like there is at present in any Library with Windows computers, today).

## Conclusion

We begin with the premise that our way of life is now dependent on connected computing and that this way of life is, unfortunately, causing quite an environmental impact due to profit maximisation of large corporate strategy. We ask if it is possible to reduce the environmental cost of computing in a way that happens to be desirable as well as financially attractive, and we find that, surprisingly, this is indeed not only possible but also offers some amazingly creative opportunities as well, that are impractical to achieve with the current world-wide accepted monolithic product development strategies.

We show clear benefits in user privacy, reductions in malware infections due to reduced computer sharing even in public libraries, work environments and educational establishments, thanks to the user being responsible for and in control of their own Computer Card.

We also demonstrate clear benefits throughout the entire product lifecycle from end-to-end, for sales teams, factories, end-users, educational establishments, showing that there are clearly far more potential scenarios that can be envisaged than there is possibly room to describe in a single White Paper.

We also provide a small glimpse into why current efforts to introduce modular computing are not entirely successful (or have been written off by quite large corporations, already), surprisingly leaving the market wide open for a radical paradigm shift in computing appliances based around the bewilderingly simple idea of moving the entire computer into a formerly well-known "Memory Card" case.

For this to be even possible, however, Koomey's Law had to bring us computer chips that were below a critical threshold efficiency, size and price-point. That point was already reached, several years ago. It's time to bring eco-conscious modular computing to a world market.

## Links

- "The monster footprint of digital technology":
  http://www.lowtechmagazine.com/2009/06/embodied-energy-of-digital-technology.html

---

*Last edited Sun 28 Aug 2016 16:30:20 BST*