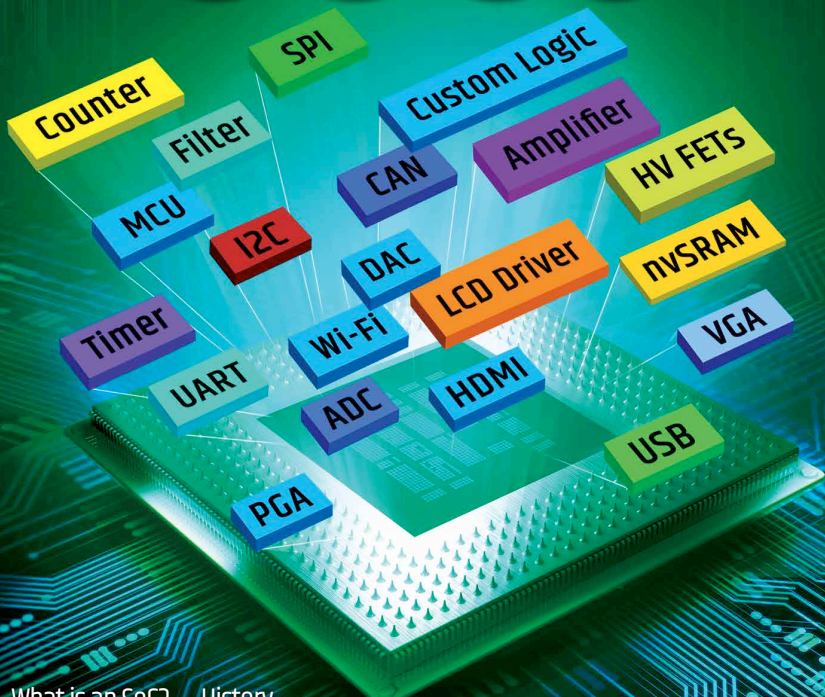
VOLUME 12 | ISSUE 05 | MAY 2017

digit Fast Track

YOUR HANDY GUIDE TO EVERYDAY TECHNOLOGY

To

SoCs



- What is an SoC? ■ History
- Typical Components - Hardware ■ Typical Components – Software
- SoC vs CPU ■ The Fabrication process ■ Architectures and their SoCs
- Popular SoCs ■ IP and SoCs ■ Next gen

A 9.9 Media Publication

Free With Digit May 2017

www.digit.in/forum

Join the forum to express your views and resolve your differences in a more civilised way.

digit.in FORUM

Post your queries and get instant answers to all your technology related questions

One of the most active **online technology forums** not only in India but world-wide

JOIN NOW

digit.in

SoCs

powered by

CHAPTERS

SoCs
MAY 2017

CREDITS
The people behind this book

EDITORIAL
Executive Editor
Robert Sovereign-Smith

Managing Editor
Siddharth Parwatay

Assistant Technical Editor
Mithun Mohandas

Feature Writer
Arnab Mukherjee

Writers
Abhimanyu Mehta
Anupama Mantri
Dhinoj Dings
Purusharth Sharma
Vaibhav Kaushal

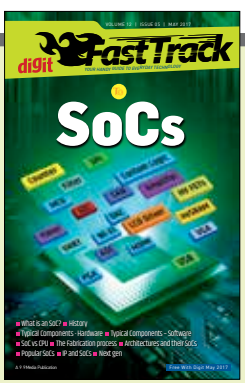
DESIGN
Sr. Art Director
Anil VK

Visualiser
Baiju NV

May 2017
Free with Digit. If you have paid to buy this Fast Track from any source other than 9.9 Mediaworx Pvt. Ltd., please write to editor@digit.in with details

Custom publishing
If you want us to create a customised Fast Track for you in order to demystify technology for your community, employees or students contact editor@digit.in

e-mail



COVER DESIGN: PETERSON PJ

Turbocharged and tiny

Ubiquitous, that's what they are. Open your eyes and you'll spot one, you probably have at least one on your person or around you at any given time. It may not be immediately apparent where and how they play a significant role. Yep we're talking about the all-powerful SoC or System On Chip to the uninitiated.

You probably mistook SoCs to be just processors. A processor or microprocessor is just one part of the average SoC. There's a lot more squeezed into an SoC's tiny stamp sized area and it requires quite a fair amount of digging to completely fathom the magnitude of what's happening at that microscopic level.

That's where we come into the picture.

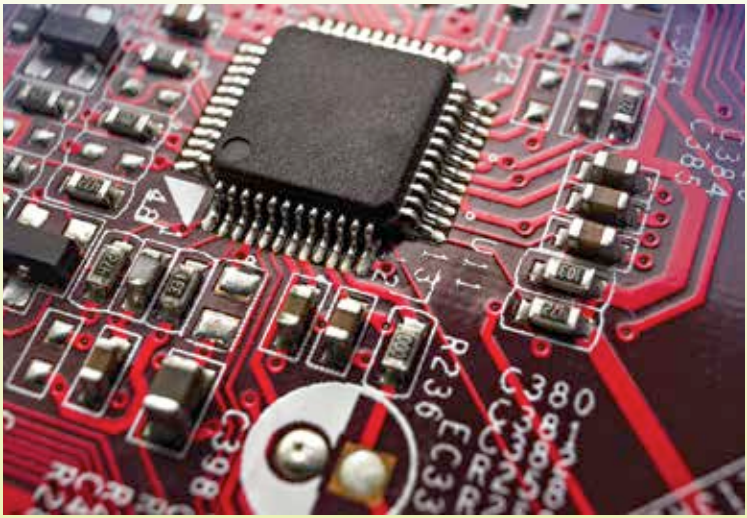
We start off with an in-depth explanation on what SoCs are and how there are very similar circuits that often get mistaken for each other. There are SiPs, SoPs, MCMs and many more circuits which all come in similarly small packages and do practically the same thing, however, we delve into the nuances of what makes each of them stand apart from one another. Then we move on to a history lesson into how SoCs came to be, what necessitated their formation and how they've grown over the years. We then get into the nitty gritties of each hardware component that goes into making an SoC. This is where you'll learn how an SoC is essentially an entire computer sans the display, keyboard and mouse. All that power crammed into a really small space.

What's hardware supposed to do without the accompanying software? When we use the term software, we really mean the whole shebang – from the OS Kernel to the drivers to the runtimes to the APIs, to the frameworks and finally culminating in the application itself. Each of these play their important role in the entire process.

Once we're done with the hardware and software lessons, we check out the different technologies that actually make it possible to shrink massive circuits down to the tiny size of an SoC. How SoCs are manufactured and what the future holds for lithography technologies. We then get into all the popular SoCs that power everyday devices. From the all-favourite Snapdragon to the underdog MediaTek, they have been built for either power or efficiency or as we've seen lately, for both.

Hopefully, after reading this FastTrack, you will be able to appreciate more about what an SoC entails rather than choosing favourites based on brand name. We end this FastTrack by peering into the crystal ball to see what the future of SoCs look like.

Let us know if you liked this FastTrack and would like to see more of these topics. And if you have suggestions for future FastTracks then we're all ears. Drop your feedback and suggestions on *editor@digit.in*.



SYSTEM ON A CHIP

From being something esoteric to becoming a household phrase, the SoC has sure come a long way.

The computer was a slow and gradual outcome of a long and winding road paved by research, microprocessors and digital logic. The basic circuits that made the implementation of various gates (AND, NAND, XOR, etc.) were the scaffolding on which the idea of a computing machine was first devised. The challenge has always been in communicating with the machine in our own way – while the machine communicates internally to complete the task assigned by us. The computer industry was experiencing the dawn of its potential in the 70s

when another concept was born. Why not make controllers and microprocessors that are specific to a task that we need them to perform? Why not make this so compact that it can be fitted inside the small devices that we want them to drive? The Silicon Valley was just getting born. Giants like IBM were showing no signs of stopping and were debating that the future indeed lies with the processors. This was the time when the concept of System on a Chip was first devised. It is very hard to trace the true origins of this concept – for it is vast and vague in its implementation. It started with something minuscule and later became the driving force behind major billion dollar industries. Little did the people of the 70s know that this elegant wizardry in building the tiny circuits and connecting them to specific peripherals could become a limitless and endless gateway to making gadgets more compact and functional. In this issue, we lay focus on the concept of System On a Chip which truly had humble beginnings but has vast potential in the future. We are nowhere close to tapping the full potential of this concept, for the concept has broken the boundaries of digital logic and entered the field of bio-engineering. We shall try to understand what an SoC is and where the need for it rises from. First, we shall briefly discuss it's architecture and design and explore it in further detail later. Then we shall take a look at what is, and what could be.

A primitive digital watch. The digital watch was the first device to witness the revolution

What is a System on a Chip?

A System on a chip is an IC (integrated circuit) that integrates the components of a computer and other electronic systems. It may contain various types of signal functions and is not limited by any one type of functional capability. The importance of SoCs are that they are compatible, cheap to manufacture and can handle processes with low power consumption. Due to this reason, they find their application in almost all modern day gadgets like phones, handhelds, media players, calculators etc. The idea is to build a circuit as per requirement. Assume the requirement to be that of making a functional timekeeping digital watch (As was the case of the first SoC in the 1970s). This function is carried out by combining a basic timekeeping circuit to an LCD screen and voila! You have a digital watch. The question that comes to mind is how is this different from a micro controller. An SoC integrates micro-controllers with advanced peripherals like GPUs, Wi-Fi etc. A good way to put it would be to say that an SoC is for micro controller what a micro controller is for processors. There is no one correct build or blueprint for an SoC. The build entirely depends on the objective that is expected to be achieved by the SoC. In certain cases, an SoC is not adequate to get the job done. In such cases SiPs (System in package) are used. We shall learn more about them later. Let us see what components an SoC comprises of.

Inside an SoC

A SoC does not have a fixed blueprint, however there are certain components that every SoC has.

1. A micro controller, microprocessor or a digital signal processor. This is the core of an SoC and provides the basic memory to connect with the other peripherals.
2. Memory Blocks which according to the requirement may be a selection from RAM, ROM or even flash memory. Again, it entirely depends on the task at hand for the device.
3. Oscillators and phase loops to keep time. Major breakthroughs have been made in this field and now we have circuits which are capable of keeping time using simple digital logic. This, in theory, would double the efficiency of SoCs.
4. External peripherals like USB, Ethernet, Fire Wire.
5. Power management circuits

   A Bus is also an important component in an SoC and the selection of the

Samsung plans to use Mediatek's SoC

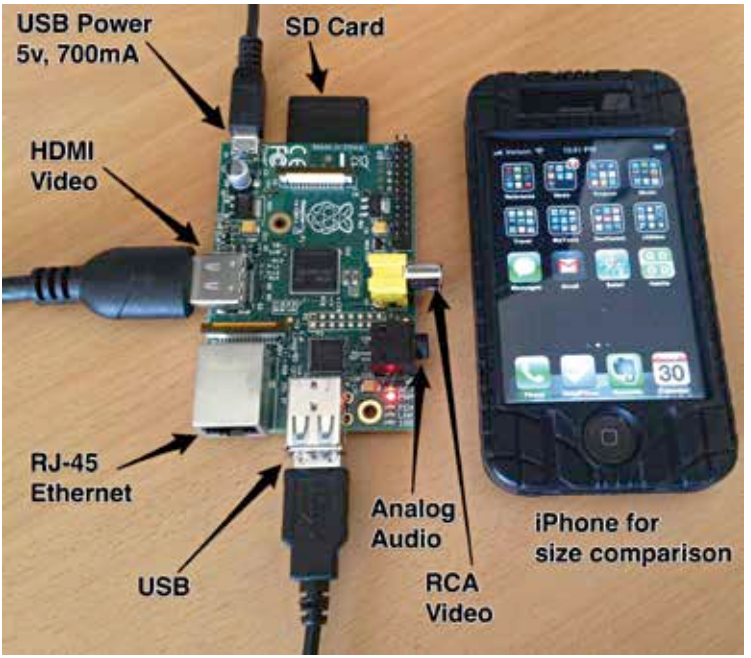
type of bus is crucial to the implementation of the system and its performance as we shall see in the later chapters.

Why do SoCs exist?

We know what a powerful system the Raspberry Pi is. It is your own DIY mini-computer and it can be programmed to do tasks which were considered huge a mere decade ago. A good example of this is the guy who programmed the entire firmware of the Sony PlayStation One to the RPi – complete with USB controllers and everything. The existence of such a device would surely blow the minds of the developers in the 80s who earned their bread by decreasing memory requirements by the KB. But the driving force behind the Raspberry Pi is an under-appreciated SoC, the Broadcom BCM2835.

Do not be fooled by their tiny size. SoCs support even multi-core processors and GPUs. In fact this field is where the major breakthrough is happening these days. So how is an SoC any different from a motherboard? The answer to this question is quite simple. The motherboard houses different chips for different components. On the other hand, SoCs have all these components on a single chip.

Another special feature of SoCs is that they are super low on power consumption. This is the one feature that makes them irreplaceable parts on devices like smartphones and mini-computers. As a matter of fact, you just need 5V supply to run the RPi via Micro USB. That's effectively a computer powered by a USB phone charger.

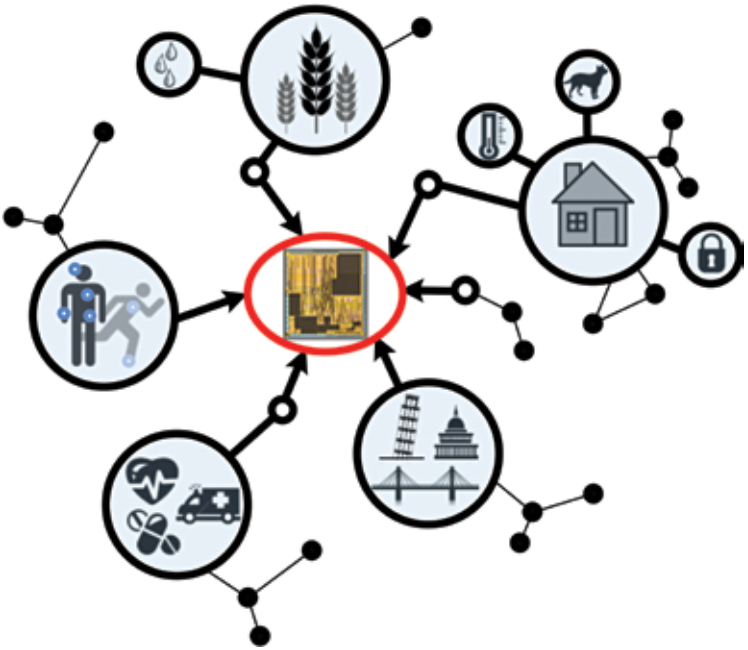


An example of I/O devices connected to the RPi

Application of SoCs

Since the 80s, if a device is small and battery powered, there are major chances that it is powered by an SoC. Smartphones, tablets, even the Nintendo Switch and the PS Vita; everything is run on these powerful mini-monsters. And they are indeed monsters when it comes to processing power. For example, the Microsoft Surface Tab was powered by Tegra 3 SoC. Again, the power and processing of an SoC depends entirely on how much processing it was built to handle i.e. the task that the chip is expected to execute.

Besides CPU, GPU and memory, an SoC can also host another important component called a Northbridge, which handles communications between CPU and other components of SoC. This is gives the SoC the ability to include features like cellular transmission, 4G, Bluetooth etc. Most SoCs are powered by ARM processors. They are chosen as they have the suitable architecture to deliver high performance with low power consumption. The major drawback of SoCs is the cost involved. Although simple SoCs



Almost any task can be achieved by an appropriately designed SoC

may be manufactured at dirt cheap prices, the more sophisticated ones are pretty expensive as they include some expensive components. However, advances are being made in this field at a drastic speed. The future holds cheap, ultra fast and ultra compact SoCs which could even be programmed with Linux based environments. The possibilities would be endless as you would be literally able to make any device of your choice – given you have the hardware to get the job done.

SiPs and SoC's

So, essentially, a system on a chip is a circuit that integrates all the component of a computer onto a single, integrated chip. It is used in small but complex electronic devices, such as smartphone or wearable computer. As SoC is both hardware and software, it consumes less power, gives better performance and is more reliable than multi-chip systems. It is specially designed to slot-in the numerous computer components in one unit. Some of the major SoC applications are

The USB 3.0 SiP

- Speech signal processing.
- Image and video signal processing
- Information technologies
  - PC interface (USB, PCI, IDE etc.)
  - Computer peripherals (printer control, LCD monitor controller,.etc.)
- Data communication
  - Wireline communication
  - Wireless communication

There are three types of SoC in general

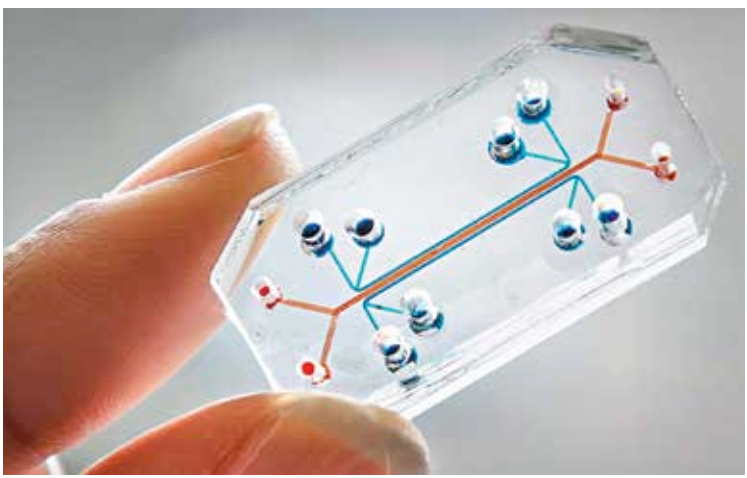
1. SoC built around micro controller
2. SoC built around microprocessor
3. Specialized SoC, designed for specific applications

There is a separate category of programmable SoC, which contain a CPU core alongside mixed-signal arrays of configurable integrated analog and digital peripherals.

An alternate for SoC is called SiP (System in package). It comprises of a number of chips in a single package and is made when it is not feasible to construct a SoC. A SiP is a further level of integration where multiple dice are integrated. Since SoC increases the yield of fabrication and its packing is more simpler than SiP, SoC are considered to be more cost effective.

The Future

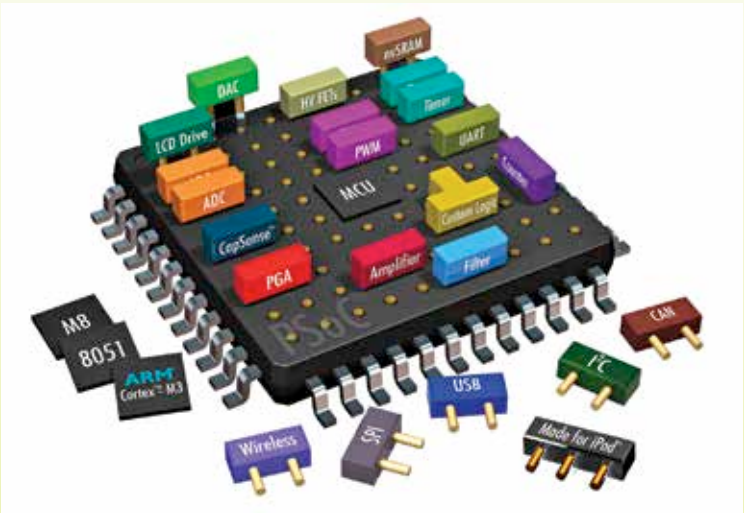
SoCs are ever developing and becoming smaller, faster and sharper. As we shall see in the consequent chapters, the future is going to be studded with devices and gadgets that have been improved and created using sophisticated SoCs. There is a study that might be able to simulate an entire human body on a single chip. As bizarre as it may sound, we have already invented and discovered ways of making DNA like computers and use DNA as memory. Only time shall tell the direction this research leads us to.



Believe it or not: A human body on a chip

In the future, SoC equipped nanobots might act as programmable antibodies to fend off previously incurable diseases. SoC video devices might be embedded in the brains of blind people, allowing them to see and SoC audio devices might allow deaf people to hear.

SoCs are used in smartphones and handhelds and that is not going to change anytime soon. However, there are many problems we still face in making them as efficient as we would like them to be. We shall take a look at such problems.

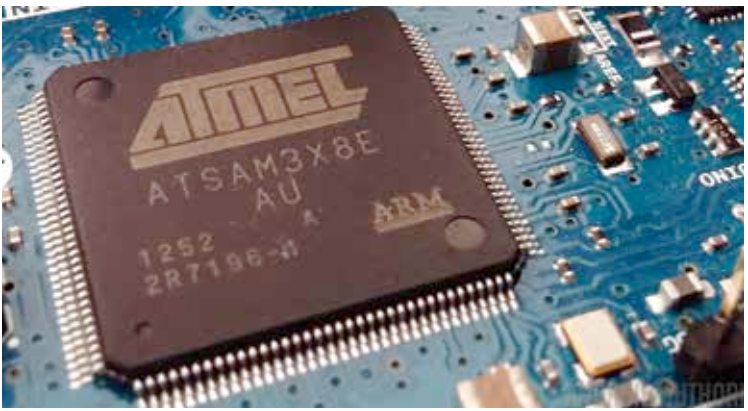


HISTORY OF SYSTEM ON A CHIP

The SoC is essentially the heart of many electronic gadgets and the journey has been quite the ride.

We know by now what an SoC essentially is. An SoC is an integrated circuit that integrates all components of a computer or other electronic systems. It is not limited to just binary signals but may contain digital, analog, mixed signal and even radio frequency functions all in one single IC. A microcontroller may be defined as a system that integrates a microprocessor with

several other peripheral circuits and memory. The tendency of the tech industry to make gadgets more compact has not been different in the case of computers. Since the time of Charles Babbage, where a 'computer' occupied the whole size of a room, we've come to a time of nanobot computers that are microscopic. We shall look at how the first system on a chip came about to be and its evolution.



A widely used ARM microcontroller

A Watch Computer

SoC is a pretty wide term as it is the name given to any circuit which integrates electronic components. However, the first major breakthrough in this field was seen in the 1970s when Electro-Data Inc. released the 'Wrist computer' (which was basically a digital watch) designed by George Thiess and Willy Crabtree. What the designers did was reduce the 44 chip and 4,000 bonding requirement for creating a timekeeping device to a single chip which integrated an LCD display with timekeeping circuitry. This was a pretty bizarre concept for the time and the watch was bizarrely priced at $2,000 which was a hole in your pocket for the time.

The first true SoC, however, appeared in a Microma watch in 1974. The watch was designed in such a way that it integrated LCD driver transistors as well as the timing functions to a single INTEL 5810 CMOS chip. TI released a single chip LCD watch in 1976 priced just at 20$. This was a big move and a huge game changer for the timekeeping industry. It was quickly realized that the commercial challenge with the use of SoCs lies in making the tech as affordable as possible. Timex and a dozen other companies entered the

An advertisement from the 1970s for a swiss digital watch

market with different renditions of the same idea. Imagine what a smartwatch is today – the digital watch was back then. It didn't take long for the SoCs to form a niche in yet another important device category, the pocket calculator. The electronic calculator was at first considered a luxury item even though it was hardly able to perform all the complex functions that the calculators today are able to perform. However, as the technology became commonplace, the prices drastically dropped because of the low cost of the raw materials required to make digital watches and digital calculators. As soon as the cost of these devices fell below 10$ to the consumers, the big companies withdrew their interest and left the market. The same market was then filled with Asian suppliers who were able to manufacture these at dirt cheap prices . That very same market domination is what we still see today in the flood of 'Made in China' electronics.

Evolution of SoCs

It may be interesting to note that today, we have much more computing power in our phones than NASA had at the time of landing on the Moon. It is because of the single chip which functions as the brain and the heart of the cellphones and laptops. It has been a long time since the SoC revolution started. It began somewhere in the mid 1990s when semiconductor processing technology was reaching 350 and 250 nanometers, at which the major processing elements of complete system products could be placed on a single die for the first time.

This would cost you a fortune back in the day

The semiconductor industry traditionally lagged in the development of the fabric that ties the chip's individual components within the chip. SoC interconnect did not receive the type of research and development focus it required and suffered from routing congestion, physical timing closure problems, increased die size, market delays, efficiency losses, bottlenecks and frivolous performance potential. Recently the interconnect fabric technology has evolved to catch up with state of the art CPU and GPU technology.

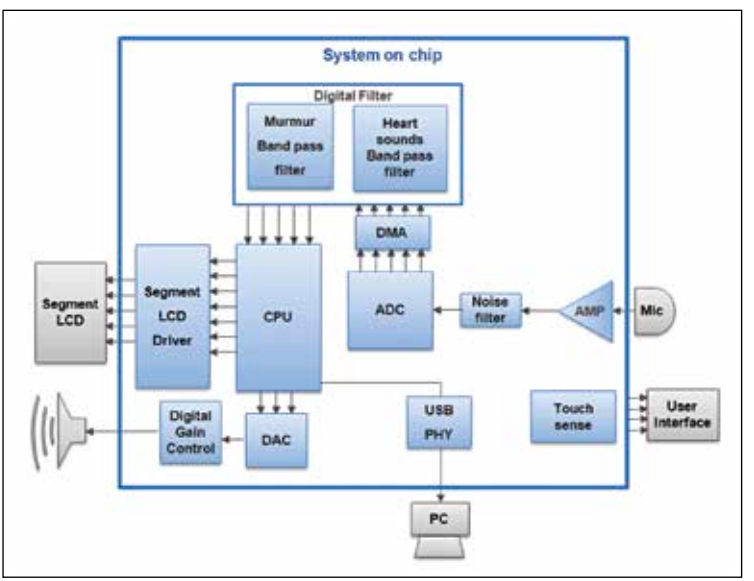
After the synthesis of the first SoCs as we just saw, several other companies and developers starting developing SoCs to suit the function of their device. The next major breakthrough came in the handheld gaming industry. SOCs were the perfect vehicle to carry this idea and were employed thoroughly throughout the 80s and the 90s in the various handheld gaming devices and even digital notepads (although their popularity never really took off). The biggest breakthrough in terms of commercial success came in the form of the device which we cant imagine living without now - the cellphone. SoCs made it possible to have mobile devices which were capable of making communication with other such devices - and eventually, do a lot more than that. SoCs now are way more sophisticated then they have ever been. But the 90s was a very important decade for the synthesis and implementation of SoCs.

Architectural evolution

In the early 70s, high levels of integration were achieved. However, the main problem was the compactness and the cost of manufacturing such integrated circuits. The architecture was relatively simple back then with a basic CPU being attached to various peripherals on board. These components like Graphical Audio interfaces, modems, codes, DSPs etc. later evolved and themselves became processing units with their own on-board peripherals. During the 80s, the complexity and functionality of these components evolved and soon, constructors began devising ways to make them universally compatible to various microprocessors. Embedded systems or Application Specific Integrated Circuits (ASIC) became a reality in the late 80s.

There were rapid advances made subsequently in semiconductor processing technologies. Recent advents like the HDTV and smartphones show an increasingly evident need for incorporating the traditional microprocessor, peripherals and memory onto a single chip. With new and more complex devices coming in the near future, there is a need for adaptive SoCs.

There are certain factors that are to be taken into account when designing an efficient SoC -



The architecture of a phonocardiograph SoC

- **Main Core**: The selection of the CPUs scheduled to the SoC are an important part of planning an SoC. The core is selected in such a way so that the architecture is efficient, well proven and easy to implement. Cost efficiency is obviously an inevitable factor to be taken in account too.
- **Inter-connectivity**: The choice of the internal bus connecting the CPUs with other components and peripherals is important. Designers have to assume the general purpose, and take re-configuration into account while selecting the type of bus to interconnect the components. Bus architecture involves planning along these lines:
  - Bus width
  - Bus pipe lining
  - Hierarchy (e.g. system bus versus peripheral bus)
  - DMA (Direct Memory Access)
  - Arbitration (scheduling algorithms)
- **Host Chip**: The selection of the host chip, the physical layer of the SoC, is a RISC/MIPS CPU in order to benefit from the modularity and well suited design provided by these CPUs. The most popularly adopted processors are ARM and MIPS processors because there is a wide range of implemented and well tested peripheral IPs for the processors readily available to any manufacturer.

Milestones

The SoC world is ever changing. There are an enormous advances being made to answer the question 'how small is too small'. We shall take a look at some of the rather interesting contemporary ones.

A Drop of Water

Recently, the first DNA/RNA cell based computers were unveiled. A single drop of water contains trillions of computers. The field of bio-computers is booming. The ability to program such small computers might form the basic foundation to building automated beings that rely on these computers the same way we rely on our DNA.

Amorphous computing

This basically combines organizational principles and programming languages for obtaining a coherent, sensible result from a series of unreliable parts that are connected in unspecified ways. Try to visualize it as making sense out of chaos and randomness. Further development of this theory will

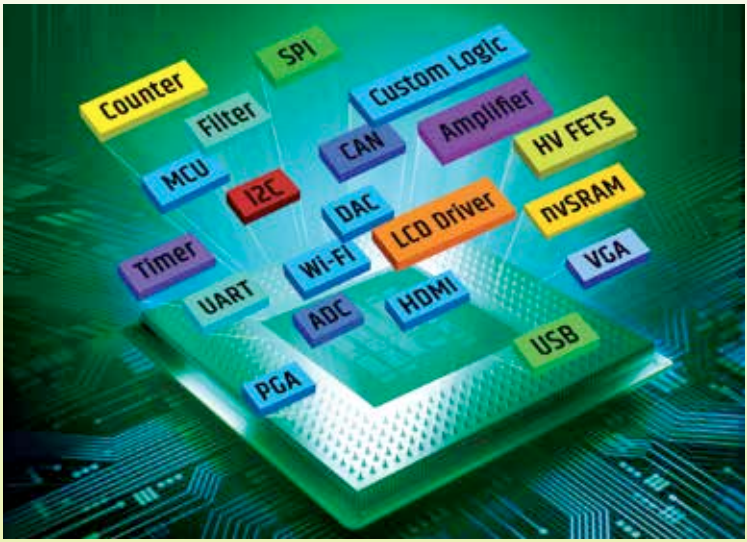
lead to construction of digital logic circuits within living cells representing logic-levels by concentrations of DNA-binding proteins.

Beating the Clock

Sun labs has been working on FLEETzero – a prototype chip with raw speed roughly twice that of today's chips. That speeds up everything by two - including our very lives. The new approach embodies the asynchronous principle that allows the logic itself to manage the time and resources required to move data. This simplifies design by removing the need for circuitry to manipulate a system clock

Imagine the possibilities with computers made of human DNA

In conclusion, it can be said that SoC and ASIC design have suffered huge evolutionary changes in a very short time span. The next important step would be to standardize the medium so that the evolution occurs at a faster pace with a large community working together on solving the same problems. The design will get better and the CPUs will get faster, thus paving the way for faster, smoother and more sophisticated devices.



SOC – HARDWARE COMPONENTS

Let's delve into the individual components that come together to give us the modern SoC.

Now that we have understood what SoCs are, their use cases and history, it's time we dig a little deeper. Allow us to use a music analogy to explain how this computing system works. When it comes to music, there are three parts to it: hardware, software and synchronization. If you're confused, allow us to explain.

When we say hardware (in terms of music), we mean the instruments. For music to flow, the instruments need to be in the right shape; they need to be tuned to perfection. Also, you must use the right instrument because if you want the sound of flute then a trumpet isn't going to help much.

Then we have the software, and this is analogous to the musician playing the instrument. It simply doesn't matter how great an instrument is if the player can't play it well. With a clueless or unskilled musician, all that comes out is plain noise.

And lastly, when we say synchronization, we mean the coherence or harmony among the players. For a tune to take shape, all musicians in the orchestra should know the notes they need to play and when to play each note so that they are in sync with the rest of the orchestra. Without this synchronization, you have wondrous cacophony.

Being a smart reader of Digit, you would probably have figured out what we mean by the analogy we just drew. If you are still clueless, here is what we mean to say: for a computing system to operate, there must exist the proper hardware in the right configuration, a well programmed software (firmware and drivers) that drives the hardware and applications should know how to operate on the hardware. Different components of the hardware and software must operate in sync with each other so that the system as a whole operates perfectly.

Given that analogy, the very goal of an SoC is to behave like an orchestra. An SoC contains several hardware components in a single package (or chip) so that they operate as coherently as possible. We are going to learn how that's done and in this chapter, we are going to focus on the hardware components that form an SoC.

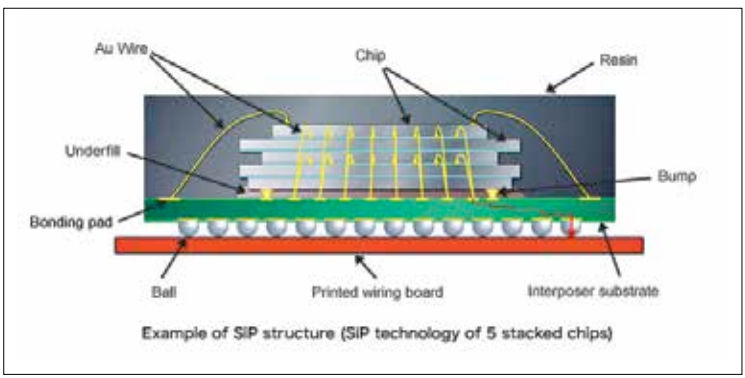
The Jargon

Let's first clear up some terms and types of integrations that are popular today. We present them to you because the lines are blurred and there are multiple terms with relatively close meanings. You might come across these terms all too often while reading this book or when researching on your own.

1. **Chip:** Another name for an IC or Integrated Circuit.
2. **Integrated Circuits:** Miniaturised form of a larger circuit where the entire circuit is built on one die and placed inside a package.
3. **Die:** A die is a piece of semiconducting material (often called wafer or substrate) over which the circuit is etched.
4. **Substrate:** The film of semiconductor material on which a circuit is deposited. A Die is a base with a substrate layer on top of it, on which

the IC has been etched. In fabrication terminology, a large thin sheet of substrate on which multiple ICs can be drawn is called a wafer. The terms substrate and wafer are sometimes used interchangeably.

5. **Package:** A circuit built on a die is often very fine and can have microscopic elements which can be easily damaged. In addition, the components of the circuit can also get destroyed by moisture or by reacting with other elements present in the ambient air. To protect against these, the die must be put inside a protective casing which is airtight, covers the die on all sides and is a good thermal conductor. Such a casing is called a package. Normally, when we pickup an IC, we pick up the package. Inside the (typically) black cuboid lies the real circuitry (the die). The pins coming out of the package are connected to the die while packaging.



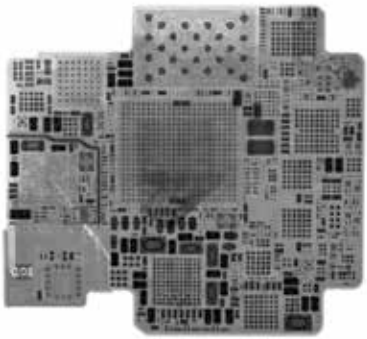
SiP is MCM stacked vertically

6. **Passive Component:** Passive components on a system are the elements which do not change their beshaviour based on another electric signal. Capacitors, Resistors, Sensors, LEDs, Speakers would qualify as passive devices.

That was the basic terminology related to SoC, SiP, SoP and MCM. If you are wondering what these are, read on.

1. **SoC:** A system-on-a-chip – one package contains multiple units which do different tasks. You've read about this a few times by now.
2. **MCM:** MCM or Multi-Chip Module is a name which sometimes predates SoCs but has a very similar function. MCMs too embed multiple circuits that can do different jobs on the same package. Circuits in an MCM are normally horizontal.

3. **SiP:** System-in-Package is an MCM which is stacked vertically. Traditionally MCMs were horizontal – if there were multiple chips on the same PCB, MCM would mean bringing them on the same substrate layer and connecting them on a single package. But that means wastage of space on the PCB. To counter this, different ICs could be stacked on top of one-another and connected vertically rather than horizontally. Such a vertical MCM is termed SiP. You can think of a SiP as a 3D MCM!
4. **SoP:** In terms of area utilised on a PCB, 90 per cent of the space is occupied by passive components and their interconnections. SoP is the process to scale them down into thin-film components and fit them on one chip. Apple Watch is one fine example of an SoP.



Apple Watch – a Mini Computer on your wrist is a SoP – it has everything as part of one single pacakage

5. **SoM:** System-on-module are complete systems on one module. Here the module is generally a single PCB on which multiple components (which might constitute a SoC) are attached. These units are normally complete systems and are designed for specific purposes. You can think of the Macbook's logic board as a SoM – it contains the processor, graphics unit, CPU, RAM, I/O controllers all on one PCB. You can attach other peripherals to it and it behaves as a complete computer.

If you read those terms carefully, you might have noticed one thing: they are all very similar (except for SoM, which is a full PCB, not a single package). In this regard, the word 'SoC' is more of an umbrella term. An implementation where you can fit in multiple circuits on a single package can be termed as an SoC. However, the way we generally try to define the new in terms of the old and yet want to differentiate among them, the terminology around SoCs has not evolved in a very friendly one.

What hardware do SoCs consist of?

A lot of different modules come together to form an SoC and there can be a lot of variations therein. For example – the lift controller requires one type

of SoC while mobile phones require another type. The type of hardware that an SoC contains depends on the usage scenario. There is no need for a graphics processor on a SoC which controls a lift while it would be an integral part of one which runs a mobile phone. We are going to list the most common ones here and explain what they do and where they can be found.

Microprocessor

Microprocessor, sometimes referred to as just the 'processor' is a small, generic processor with all the functional units. Microprocessors have their own instruction set using which they can run generic programs. They do not have a specific function (like a DSP, which is tuned for only processing signals), instead they can take a series of instructions and can execute them one after another.

Any SoC which is supposed to execute generic programs would contain them. Such SoCs can be used to execute application programs. Typically, a microprocessor needs an OS to function. It need not be a full-fledged desktop-grade OS and can be something that fits in the ROM (memory of the microprocessor). Depending on the capabilities of the entire SoC, it can be a feature-rich OS as well (think Android or iOS).

If this made you think about mobile phones, then you are right. They are also found on music players, smartwatches and miniature-size PCs (like Raspberry Pi).

Microcontroller

Microcontrollers are a superset of microprocessors from a design perspective. A Microcontroller normally consists of a microprocessor, some ROM and some RAM memory along with programmable I/O controllers. The way microcontrollers normally differ from microprocessors is:

- The microprocessor included with a microcontroller may not have a wide variety of functionality found in common gadgets like smartphones or smartwatches. They lack and extensive instruction set and features.
- The amount of RAM and ROM are very less compared to what we are used to in our day-to-day gadgets. They are just enough for the use-case they are designed for.

Microcontrollers can be generally found in appliances. Washing machines, music systems, ovens, refrigerators, car monitoring systems are fine examples of microcontroller-based SoCs. They do not run full-size OSes

or generic programs. They cannot process different types of data flowing into them. Instead they can do a small set of functions and are designed for only those. For Example, a household music system cannot run a fully functional web browser application but knows how to process FM Radio signals, ask the CD player module to read data, display track information on the LCD, process signals received from the infrared remote etc.

Digital Signal Processor (DSP)

A DSP is used to convert analog signals to digital which can be understood by microprocessors, microcontrollers and any other components in the system (SoC). DSPs are vast in their definition, implementation and use-cases - ranging from processing mobile signals, audio and video processing (encoding, decoding, converting, compressing, encrypting etc.) to reading pressures, temperatures and medical imaging. These signal processors run mathematical algorithms designed for the use case they serve.

There is an interesting side to DSPs when it comes to SoCs – many algorithms that DSPs execute on a hardware level can also be run on microprocessors at a software level – so you would not need another piece of hardware for decoding audio on your mobile; your quad core processor could run it. Why then are DSPs built as hardware chips? It's because most of these operations are compute-intensive mathematical operations. Running them on the generic microprocessor would waste energy, can get slow if the OS schedules the processor to do something else or because the algorithm requires parallel processing of multiple samples of data. Running a signal processing algorithms on the microprocessor can heat it up. This is why DSPs are separate from the main processor.

The list of use cases for DSPs is huge. DSPs are used in audio and video signal processing (sampling, encoding and compressing), speech recognition, reading natural environment (part of weather forecasting), seismology, medical imaging, radars and sonars (hence, scientific and military implementations) and a lot more. If there is a system which involves reading analog data and processing it in some way, a DSP is most probably part of the package.

For example, a missile targeting system might contain a GPS, an image processing system, a heat sensor, pressure sensor for calculating its position relative to its target. All those systems depend on a DSP to convert the information into digital format so that the main processor can run its algorithms to calculate the position and take further actions.

ADC/DAC - Analog to Digital and Digital to Analog converters

These circuits can convert digital signals to analog ones and vice versa. Usual applications include audio and radio signal encoding and decoding. Sometimes they are not part of the main SoC chip but reside as an external chip connected to the SoC on the same PCB. They can also be part of another chip. For example, a mobile can have an ADC/DAC audio chip for handling music playback and recording via earphones. The chip can be part of the DSP embedded into the SoC or stay on the same board as a chip outside of the SoC.

Memory

SoCs are essentially mini-computers which can work on either a specific format of data or they can be generic. While the use-case determines the amount, there is no denying that almost all implementations of SoCs are going to require some amount of memory.

Memory for an SoC can be of a few types:

1. **ROM** or Read Only Memory is supposed to be the kind of memory which can only be read from but not written to; they should ideally be hard-wired on a chip. However, the term is commonly used as a reference to EEPROM.
2. **EEPROM** stands for Electronically Erasable and Programmable ROM. In normal operations EEPROMs operate in read-only mode but can be erased and reprogrammed with new contents. The reprogramming process is a slow one and is not done 'often'.
3. **RAM** is used as the operating memory and allows the processor to access data faster than from disk. It acts like a buffer for storing both, executable as well as non-executable data stored on the disk.
4. **Flash memory** is what you find on most compact storage devices like SD cards and pen drives. Data stored on them persists beyond a power shut-down.

Depending on the type of SoC, it may or may not have a separate RAM or Flash memory e.g. an SoC built for an air conditioner might not need a RAM or Flash memory while a smartwatch is going to need both because it needs to run applications that demand both storage and memory larger than what a processor typically has. Your smartphone has all of these types of memories.

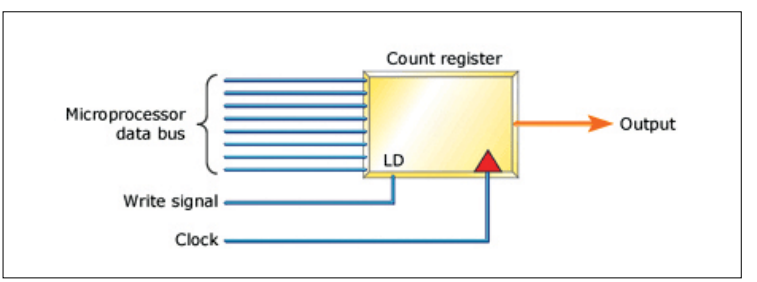
Timers

Timers are one of the most useful parts of any computing system. To measure the time taken between any two events is a crucial part of a lot of operations in the real world. We cannot imagine a scenario where timers do not make an impact. Timers can either count up (like a stopwatch) or count down (like a countdown timer). Keeping track of current time after the time has been set, measuring time taken for a task to be completed, cancelling a task when the task has been executing beyond expected length, starting a task at a given time are all done using timers in microcontrollers.

However, that is not the only use. One of the most important uses of a timer comes up in rather sophisticated SoCs which perform multi-tasking. An electronic timer can be used for raising interrupts on the hardware level. These interrupts are captured by the processor and an OS routine is invoked. This routine is called an interrupt handler. If the microprocessor of the SoC is a single-core processor, then the running application is first suspended (the execution is paused and the state is saved so that it can be resumed) before invoking the routine. This routine then runs multiple checks against the current state and decides what to do. A

NOTE

There is yet another use of a timer from a pure-hardware viewpoint which makes the timer the most important part of any chip – clock ticking. This is the "hertz" in the specification-sheet. A timer which counts very fast would be called one with higher frequency. All digital equipment depend on clock ticks of their respective chips. This tick is an electric signal produced inside the IC which tells the units to perform the next step. This ticking depends on a digital timer. Timers are thus part and parcel of almost all ICs.

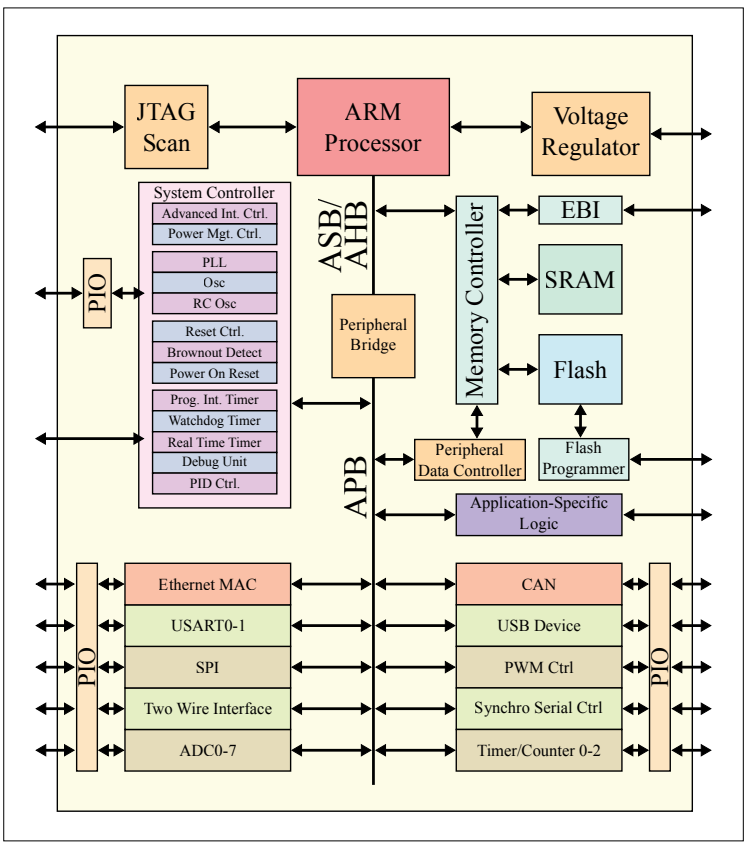


That's how timers work

task scheduler may also be invoked which can decide to kill non-responsive tasks or one that has been running for longer than anticipated. This helps maintain stability of the system on a software level and avoid crashes.

System Controller

The System Controller is normally the part of an SoC which hosts multiple circuits responsible for the SoC's state management. It is a component on sophisticated SoCs built for more generic purposes, which can mimic a full-fledged computer, e.g. a smartphone. For SoCs having a system controller, some timers may be designed as a part of the System Controller. Apart from timers, system controllers also contain oscillators and debug units.



Generic ARM SoC Block diagram

The Bus

The individual modules within an SoC need to communicate with each other. The system bus is what connects them. The bus is a set of wires and protocols. All modules attached to a bus should adhere to the wiring as well as the protocol. Without the bus, the modules cannot send or receive data. For example, AMBA protocol is a popular bus architecture designed by ARM systems.

Memory Controller

Let's say the processor needs to read "128 Bytes of data from address *0x00DEA0FE* from the RAM. While to us humans, that might look like a valid request, such a request is not enough to actually fetch the data. This is because the protocols to read ROMs are different than that needed to read from RAM which are again differ from protocols for reading from persistent storage units. To fulfil a read or write request, the low-level protocol for storing and retrieving data needs to be known. The Memory controller is the unit responsible for this work.

Interfaces

An SoC normally has multiple interface controllers depending on the use case. For example, a mobile phone SoC would contain interfaces for USB, Wi-Fi, LTE Radio, Bluetooth, Display (graphics), SPI (Serial Peripheral Interface) etc. These interfaces are connected to the processor via the system bus and are connected to their respective h/w units (e.g. a Bluetooth transceiver) as well.

Many of these units can be called SoCs in their own right. For example, a wireless connectivity system on a mobile can contain the transceiver hardware for Bluetooth, Wi-Fi, GSM/LTE as well as the DSP system for these technologies – all of this can be outside the main SoC.

Power Regulators

In an electronic or electrical system, everything depends on electricity and in a country like India, voltages are not guaranteed. While there are circuits outside of the SoC taking care of the power supply, an SoC needs to have a voltage and current regulator built right into it so that it can save the rest of the circuits from getting fried due to voltage spikes or causing an error when electrical energy is below the required ranges. They are responsible for maintaining voltages and current levels. These circuits can also be con-

nected to temperature sensors to halt the system in case the temperature rises above a threshold. If you have ever heard of phones shutting themselves down because the temperature was too high, or witnessed your phone shutting down because of low-battery, these gentle-chips were doing their job!

Graphics Processors (GPUs)

These are not found on all types of SoCs but only on the ones which need to perform graphic-intensive work. Smartphones are one of the most common SoC implementations which contain Graphics Processors. Graphics processors are not used on every system which needs to display something. A system with a simple LED display would not need one. Also, not all systems outputting data on bitmapped displays need a graphics processor. For example, a simple weather reader displaying weather and time on a bitmapped display does not need a dedicated GPU.

Graphics Processing units are useful on SoCs which need to perform tasks such as playing games, processing shadows, etc. If the system does not process 3D data, a dedicated GPU is not always required; they are power-hungry and can generate a lot of heat.

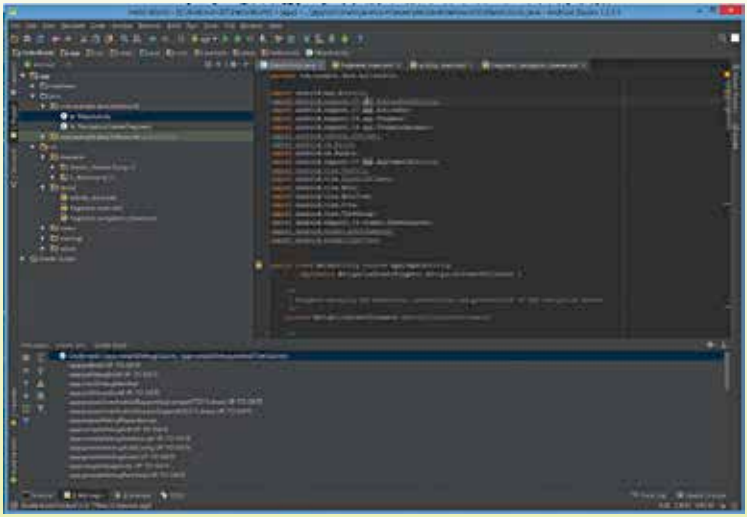
On-Chip Debug and Self-Test systems

Unlike software products, you can't just 'rewrite the code and push an update'. If any chip has a hardware-level bug, the company would have to fix the bug and resend the updated package. If the package was used to build a consumer gadget then it's even worse.

In the early days, debugging a chip meant that the tests were run using bond-out processor which had to be physically placed where the processor was supposed to be. Modern SoCs contain an On-Chip debug circuits which allow the developer to put the chip in debug mode where the behavior of processor changes and instead of depending directly on electronic clock-ticks, they start receiving ticks via the debugger unit which allows the programmer to inspect the variables and system state closely.

Summary

While we are running out of space here, we believe we have introduced the most generic hardware you can find on an SoC. But that's just the tip of the iceberg. If you were to start getting deeper into the terms introduced in this chapter, you would find yourself in an ocean of information. That said, we are going to move to the next chapter – about software on SoCs.



SOC – SOFTWARE COMPONENTS

There's a lot more happening under the hood of your favourite gadget. Here's a primer into how software plays a part.

Let's begin learning about SoC software by writing some for an SoC, shall we? We know that most functions of a smartphone phone are done on an SoC which contains the microprocessor, RAM, some ROM and Flash storage. A smartphone is the best SoC platform we can start with.

You might think that the simplest way to write software for an SoC would be to fire up Android Studio/XCode and create an App that displays the famous "hello digit" program (did you notice what we did there? :P). But that's not how software for SoCs are written!

Apps written for a mobile platform would not qualify as a software component for a SoC. Apps (as the name suggests,) are application software. They need a complete operating system and execution environment to run. E.g. an android app would need a execution environment for the application code (Dalvik or ART) to run. If you were to take the binary code from that app and attempt to run it on the SoC processor, you would not get the desired result. It would appear as crash! For a piece of software to be called a software component for the SoC, it should be closely tied to the SoC. Let's dig a little deeper.

Layers in Software – a primer

With all the buzzwords thrown around, it is easy to be trapped into thinking that an SoC is the mobile phone and SoCs were designed only to be used in mobiles. We have already expressed that to be untrue – that mobiles need more hardware than just the SoC and that there are SoCs implemented in devices other than mobiles. However, smartphones are the most popular implementation; thus, we will start with them.

An SoC is a system and by system, we mean a computing system. Software written for any system that can perform generic tasks are logically layered. The part of a software which interacts directly with hardware is said to be lower-layer software. It normally consists of the kernel and drivers. These parts know how to talk to the hardware (which is why they are called lower-level code). All other software depend on these low-level software components. For example: a Bluetooth driver knows the protocol its hardware unit follows and thus can ask it to send a pairing request signal. The UI elements which allow you to pair your wireless headsets would not be able to do anything if the driver did not know how to tell the Bluetooth controller to initiate pairing with a new device.

On the same level reside other critical components of the OS which includes process schedulers, memory managers, interrupt handlers, network stacks, display managers etc. Depending on hardware components built into the SoC and the implementation and usage, these parts may or may not be present. The kernel (another name for core) of an OS contains code that can interact with the microprocessor. Drivers for devices attached to the SoC also execute with low-level privilege.

A word about privileges in hardware

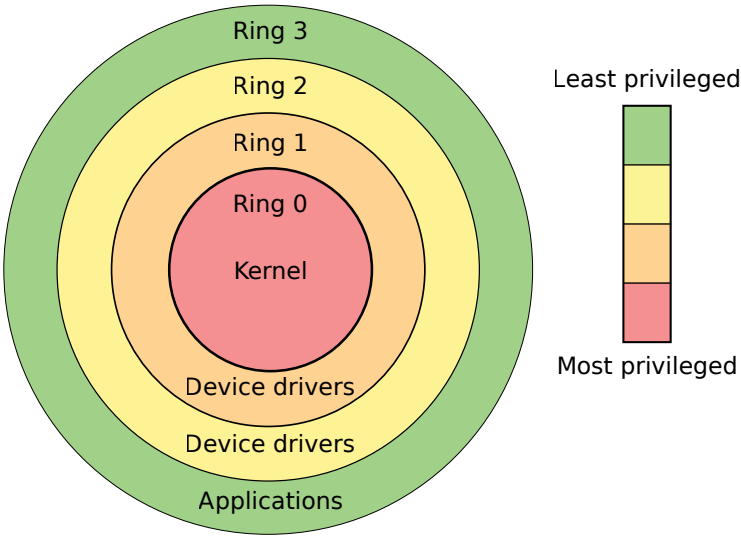
Privileges are levels of protection built into Microprocessors which allows or disallows a piece of software to send signals to the hardware. For example, a Bluetooth device driver must be able to send electric signals to the radio circuits to start emitting wireless signals. However, the UI component, even when presenting the options for pairing new devices should not be allowed to directly send signals to the hardware because:

a) The UI component might not know the low-level hardware details for emitting signals responsible for pairing a device.
b) The request might need to be authenticated first (e.g. the user must enter a password for connecting to a new device) and thus needs other parts of the OS to come into play.

In the second case, the UI components might need to be stopped or suspended for the time and the OS must first authenticate the user. The UI component might crash if it encountered an exception when trying to initiate. For this use case (and plenty others), the code which can control hardware is usually run in 'kernel space' or 'privileged mode'. Such code communicates with higher-level software components using a predefined and well-documented API presented by the OS on behalf of the driver.

There are multiple levels of privileges built into microprocessors. The more advanced and generic the microprocessor, more the number of privilege levels. If you really want to understand how OS interacts with hardware, you can go through the Intel's Software Developers' Manual located at: https://software.intel.com/sites/default/files/managed/39/c5/325462-sdm-vol-1-2abcd-3abcd.pdf. Note that this is a 4700 page manual and is an advanced guide for people who want to write low-level code (Privilege levels are described in section 6.3.5) If you are looking for ARM, you can look here: http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0552a/CHDIGFCA.html Once again, the content on this page is a small part of a large book on the architecture of ARM Cortex A73 processors.

Based on the low-level code is another layer which serves as an API which the application software can call. If a program wants to read a file, it would call a function from this API. This function cannot invoke the processor to send electric signals to the storage unit or memory controller; instead it calls another method within the OS which can lay out steps to read the



Privilege rings

file, figure out the addresses of files in the storage unit, then calls upon the memory controller to read the blocks from the storage unit where the file data is known to reside. Remember that this is an overly simplified view of what happens.

The highest level of an OS is the shell. The shell can be either a command-based terminal (like command prompt in Windows or the Bash shell in UNIX-like systems) or a graphical interface. Applications written for end-users generally just make calls to the APIs which know how to communicate with the rest of the OS to get the job done.

SoC software components – what they are

The use case of an SoC may or may not utilize a complete OS which interacts with the user and presents a pretty interface. We are not going to talk about the OS, services, implementation, usage and interface with the end-user. We are going to focus on the lower levels only. It is noteworthy that SoC software can be complex or simple depending on the hardware.

With time, as Internet penetrates more and more lives, IoT (Internet of Things) will gain momentum. SoC is at the heart of IoT. All these Things make use of SoCs and are intelligent because of very sophisticated

hardware and software. Let's take a look at the software components one by one.

Kernel of the OS

The literal meaning of the word kernel is 'core'. The kernel is responsible for handling pretty much everything else on the system. The kernel knows how to use the processor in the SoC above everything else. It knows the codes to send to the processor. Therefore, you can't use an OS written for ARM microarchitecture to work on an Intel microarchitecture because the kernel uses different instruction sets. In fact, the processor would not be able to even load the OS when the system boots! The kernel can be monolithic, microkernel or a hybrid one. Since SoCs of today have become quite sophisticated, the kernel of OSes they run are big and sophisticated too.

Drivers

The kernel knows how to control the microprocessor in the system. However, it cannot always know how to control every single component connected to it. For example, if there is a Bluetooth module in the SoC, then depending on the make and model, the unit may or may not have certain abilities (think Bluetooth profiles) and support a specific version of the protocol (e.g. Bluetooth 4.0).

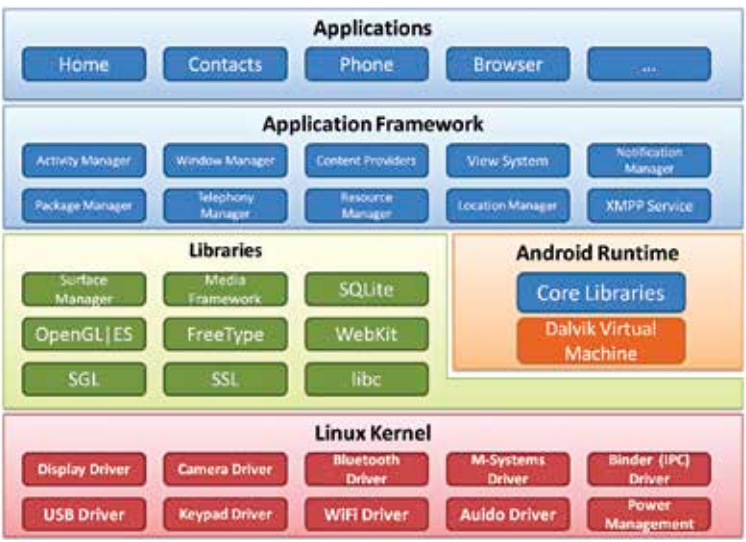
For making sure the OS can work with multiple types of hardware, drivers are created which can work with the hardware in the SoC. Drivers call certain APIs in the OS and the OS can call the registered methods in driver to control the hardware back. Such drivers are device drivers. There can be other drivers too. One such example would be a file-system driver – when you attach an NTFS formatted USB to your mobile device, the OS while with being able to talk to the hardware must also know how to interpret what's on the disk so that it can present it to you in the file explorer. Drivers execute in privileged mode but on a higher ring than kernel so that they can be controlled by the kernel. This design allows the kernel to handle crashes in drivers gracefully.

Consider this: if you plug a USB device into your mobile via OTG and the file system driver crashes then your phone would crash too if the problem went undetected by the kernel. However, if the OS could detect the crash in the driver, it could just stop the driver (and any process that was using its functions) and show a message to user saying that the USB could not be read and silently re-load the driver in the background.

Protocols and stacks

The word protocol only means set of rules. The word does not convey anything more than that. A networking protocol is a set of rules for networking while a bus protocol would mean the set of rules for using the bus. An SoC is a complex hardware setup in a compact form-factor and normally has a standard bus governing the data transfer.

All peripherals use the bus. For the OS to orchestrate communication between multiple devices on the SoC, the devices and the OS must know how to operate on the bus. In addition, there might be other protocols of data transfer for each device. Also, there can be multiple buses in a system. For example, the bus used to transfer data from the RAM to the processor



Android Environment (the runtime can be either Dalvik or ART, depending on version)

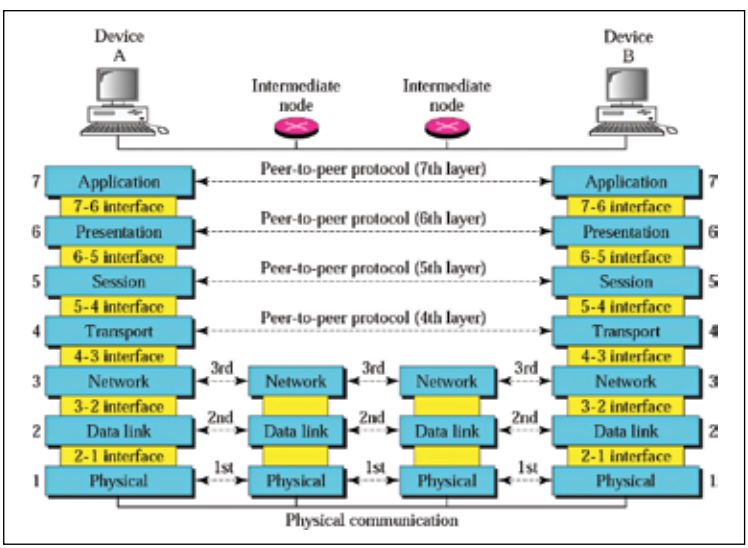
is normally separate from the one used to transfer data from a networking device to the processor. The lower-level software must know the number of buses and how to use and control them.

Let's talk about protocol stacks now. Most protocols are designed in a stacked manner, much like OSI model where highest layer on the sender's side encapsulates data in a package and lower layers encapsulate the already encapsulated data in their own respective formats. The receiver unwraps the data in the reverse order of encapsulation.

Consider this scenario, you are browsing a website using *HTTPS* on your tablet while it is connected to your phone's Bluetooth *PAN* and the phone is connected to the internet via *LTE*. In this case, the tablet is sending and receiving *HTTP* data using *TCP* protocol where the data was encrypted on the session layer of the *OSI* model and was wrapped and routed using the Bluetooth protocol by the tablet. This data was then unwrapped on the phone and re-sent using the *LTE* protocol. If the protocol stacks for *HTTP, TCP*, *Bluetooth* or *LTE* were not implemented, it wouldn't have been possible to achieve this feat!

Software creation for SoC

To write software that works with an SoC, one needs the right set of development tools, just the way you need Android Studio (in fact, the toolchain it provides) for building Android apps. For writing an SoC software, you would need a compiler which can convert your code to binary format understood by the processor. Most lower level SoC software is written in C or C++ and is compiled for the specific platform. You would need a C compiler for ARM architecture if you need to build a program that runs on Raspberry Pi but you would need the compiler for x86 architecture if you are building for Intel Galileo platform. Drivers or protocol stacks for an SoC are also written in C using the development toolchain for that architecture.



The OSI Protocol stack

If you are wondering how software controls the bus, you need to understand how buses are used. The bus is a set of conducting wires used by all chips connected to it. When a device wants to send data, it sends the electric signals via these wires. This is very simple till you question "what happens when two devices want to send the data to the processor at the same time?" If two or more components were to start sending data at the same time over the bus, all data would reach corrupted because the electric signals from multiple devices would cause interference with each other.

For this, a system of interconnection is built which governs how devices can send data over bus at the same time (different modules using different interconnections), or one after another (scheduling the usage of bus) depending on how the bus was architected. Bus design and implementation is a vast and interesting field.

In case you're wondering "why C?", the reason is "performance". The C programming language compiles to binary code and offers capabilities to control the hardware easily. Most operating system kernels are written in C and so are most drivers. It is easier to write code in C than in any platform's assembly language while intelligent compilers ensure that the generated binary remains performant.

How do Apps run?

Software we have been talking about till now are system-level programs. Though apps are not the focus of this chapter, it's time we talk about the wheels of the smartphone revolution – apps!

The word 'application' means to put something in operation. Application programs are software designed to put the hardware into operation for a task. We have already discussed that operating systems are designed to make sure that certain actions are privileged and to perform those operations, the OS exposes APIs (Application Programming Interfaces).

Applications are normally compiled for a certain OS and architecture so that it can use the capabilities of the target OS and run code on the target microprocessor. When the application is launched, the OS reads the application file, loads executable code in memory and asks the processor to start executing the loaded executable with the lowest privilege levels. As the application executes, it calls APIs provided by the OS for its needs (like reading file from disk, sending data over network etc.).

Applications written in cross-platform languages (like Java, C#, Python, etc.) are not binary programs that can run directly on the hardware. Instead, they need the language runtime environment to be present on the OS which interprets the program and runs it. The execution for such a program happens in this manner:

1. When the application is launched, the OS reads the file and determines that the program needs a language runtime to execute. The OS determines this either by reading the file extension or the first few bytes of the file.
2. The OS launches the language environment like any other application and supplies it the original program file it read in step 1. If the runtime is not present, it can show an error or open the file in an editor program.
3. The runtime reads the file. Depending on the language runtime (e.g. Python runtime) it would first interpret the file and generate opcodes (standing for operation code and sometimes called bytecode). Languages like Java *or .NET* expect the input to be in bytecode format.
4. Runtime reads the byte-code and converts it to binary codes which can run on the platform's microprocessor. For performing operations which need support from OS (like reading a file or transmitting data over network), the runtime translates the bytecode instructions into API calls for the OS.
5. Once all execution has been done, the language runtime stops itself and exits.

In these cases, the language runtime must be designed for the platform (Processor architecture and OS).

Software used to design SoCs

So far, we have talked about software that run on SoCs. However, designing one needs a lot of software tools as well.

Design

One of the first things that is required for designing hardware is a language using which various operations of a standard electronic circuit could be described. Such languages are called hardware description languages. VHDL is one of the oldest languages in this realm while Verilog is a newer language with multiple revisions that have kept it relevant and advanced. These languages allow the designer to build the "logical" model of the hardware wherein the expressions of the language define the way circuits would communicate with each other and how electric signals flow among them. High-level synthesis programs are used to write the desired behavior which are converted to their corresponding HDL representations.

Once the hardware design on the HDL level is complete, schematic designs can be used to visually present the circuit design on a computer. A Schematic editor not only presents a graphical view for placing components, they can also be used to verify the function of circuits by simulating the circuit and running tests on them.

Once logical and schematic designs are completed, the SoC needs to be fabricated. For that, the circuit designs need to be converted into physical interconnection on a silicon die. Since modern chips have billions of transistors interconnected, manual placement of each transistor on a circuit is impractical. Layout tools take care of that. These tools are used to automate the physical design of a circuit. The complete IC can be large and may contain redundant circuits placed at different positions (e.g. Each processing core is going to need its own set of adders). Layout tools take modular circuits and use them to build larger circuits which can be verified and tested on a simulator.

Simulation and Testing

Although we are presenting it separately, testing of design at various stages go alongside the designing process. A logical problem on the HDL level would mean that both schematic design and layout of the chip would change. For each level of design, the testing and verification is done using tools designed for that level of design. For example: a HDL testing tool would be able to verify if the circuit defined using HDL can perform the desired

Schematic Design of a Circuit

tasks perfectly or not. If the testing tool detects a problem, the programmer can update the description and run tests again.

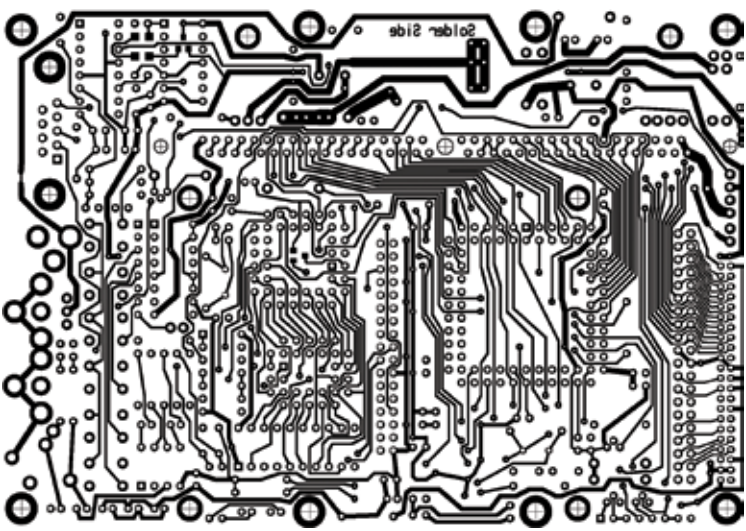
Schematic tool would test the circuit and can show results visually. Then there are layout tests which interestingly do not look for logical errors. They instead test for performance issues caused by delays in paths, look out for lengthy wiring used in layout, check if congestion exists on routes within the IC and so on.

Tools for the creation

Electronic Design is not popular as a hobby because it requires expensive equipment for producing results. Due to this, software available for EDA (Electronic Design Automation) are either commercial tools or educational software with limited features. We are going to present the ones which you can download and use for free without having to sell all your possessions or internal organs. They should be enough to get you started with EDA.

For Verilog/VHDL:

1. DVT Eclipse: *https://www.dvteclipse.com/* (Time-limited trial, Academic License)



A PCB Layout

2. *https://www.edaplayground.com/* – A free of cost Verilog/VHDL editor in the browser.
3. ModelSim: *https://www.mentor.com/products/fpga/model/* (Academic License available)

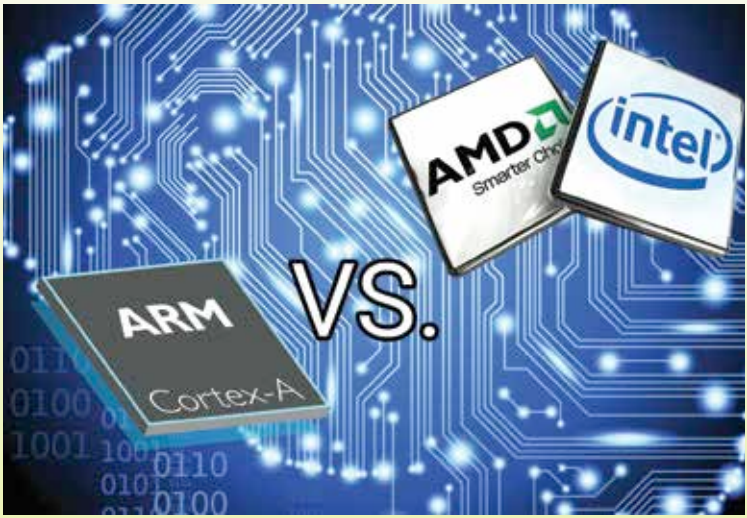
For Schematic Design:

1. KiCad: *http://kicad-pcb.org/* (free, cross-platform for Windows, Linux and Mac)
2. Eagle: *http://www.autodesk.com/products/eagle/overview* (trial, academic license)

For Layout:

1. KiCad: *http://kicad-pcb.org/*
2. TARGET: *http://www.ibfriedrich.com/en/index.html#productsPage* (Free version available)
3. Eagle: *http://www.autodesk.com/products/eagle/overview* (trial, academic license)

We hope you will get your hands dirty!



SOC VS CPU

Often mistaken for one another, the SoC is quite different from the more powerful CPU. Here's how.

So far, we have been saying that an SoC is a complete system on a single chip. We have also conveyed that smartphones run on SoCs. And smartphones can do everything that a desktop or laptop computer can. It can:

1. Run a sophisticated OS - Evidence suggests that you can run a Linux OS built for desktop on your phone.
2. Use and facilitate multiple networking technologies - The phone can talk over Bluetooth, Wi-Fi and mobile communication technologies (2G, 3G, EVDO, LTE etc.).
3. Perform graphical computing – GPUs of mobile SoCs are being used to power large mobile displays with more pixels than our full HD televi-

sions. These GPUs are being used to play games and that means mobile GPUs can do some decent 3D computation as well.
4. Connect a mouse and keyboard – You can connect a mouse and a keyboard, either by using OTG cables or Bluetooth.
5. Be used as a camera – for taking pictures, videos and video-calling features.
6. Be used as a compass or a GPS receiver or as movement trackers.

If you were to perform all of this on a PC, you would typically need to connect multiple peripherals (keyboard, mouse, display, webcam, a GPS module etc.) for getting the work done.

Now let us out the components of a PC.

1. Processor
2. RAM
3. Graphics processing unit (discreet or embedded into the processor)
4. Storage devices (HDD/SSD)
5. The main board (motherboard)
6. Fans
7. Wires for connecting the components
8. Power supply unit (SMPS)

And what does a SoC contain?

1. Processor
2. RAM
3. An integrated GPU
4. Storage (Flash memory)
5. The circuitry for interconnecting the above-mentioned components (acting as the main board)

Since an SoC does not need any wires for connecting the components (they are already connected to the internal bus), a power-supply unit or fans, a SoC is almost as functional as the PC. That's not to say that SoCs don't need power at all. The power supply is external in this case and most SoCs are designed to be used in a low Thermal Design Power situation. So they don't need extra cooling which a fan provides inside a PC.

However, this is not the complete story. There are capabilities unique to each of these device classes.

Good things about the PC

One major area where a PC beats the SoC is upgradability. You can change the components of a PC and upgrade it. A better processor, more RAM, faster

storage or the next-gen GPU – you change what you need, when you need (while compatible components are available in the market). This is simply not possible on an SoC. An SoC is a single package with all those components built-in. If you were to upgrade something, say RAM, on an SoC, you would need tools which can cut microscopic wires, remove dice from the package individually and solder the new RAM back on package without damaging other components. Oh, and you would need to close the package too! You see, upgrading a SoC is no common a feat and is almost never done. Current gen SoCs are etched on silicon so forget soldering altogether.

Liquid cooling on demand for PCs

The second area where the PC beats the SoC is heat-dissipation and performance. When you tax your PC with some serious compute job (like rendering an Adobe After Effects movie), it might run for hours to get the movie rendered. In the process, it will utilize the CPU (the processor) to its max, do a lot of read-writes on RAM and may put your GPU under pressure too. These are power-hungry devices which can heat up to 70-80 degree Celsius easily. Do that on an SoC and you would probably damage something inside it.

Each component in a PC can use heatsinks and liquid cooling to keep them working. SoCs contain all the components of the PC on a single chip

– that leaves no room for internal heatsinks or extra spaces to allow cooling these components separately. Heating up of any component inside SoC would also heat up other components. So, if you are playing a game on the mobile which does not tax your processor but only the GPU, it is still the same chip which is getting hot. If all of them are put under load, the temperature would rise quickly and the voltage regulation unit would shut down the chip once the temperature threshold value is reached.

SoC versus CPU

For once, let's compare apples with oranges. We are going to do the unfair comparison of a SoC vs the CPU. Before we proceed, let's clarify the unfairness. SoCs are complete systems on a single chip, while CPUs are only processors with some cache and no RAM or Storage (modern Intel CPUs do have a graphics processor built-in though). In addition, the SoC can house DSP modules of various kinds too. It is, thus, a tad unfair comparing these two. But let us tell you – it's interesting!

Note: While it is common to address processors inside SoCs as CPUs as well, we are not doing that here. When we say CPU, we mean the processor you find in a Laptop, Desktop or a Server.

When SoC beats down the CPU

There are certain areas where CPUs can't stand up to SoCs:

1. The SoC contains the entire system and it can be tasked with anything that is achievable by a PC. The CPU, obviously, cannot get that done alone.
2. SoCs can contain DSP modules which makes them super useful. Audio synthesis, imaging, video and analog communication need DSPs which can be built right into the SoC. In many cases, the CPU might not have those capabilities, especially communications and audio synthesis systems.
3. Size is obviously an issue addressed by SoCs. Many smartwatches are full blown computers. It might be interesting to know that the Apple Watch has more processing power and memory than the system that runs Curiosity Rover on Mars. All that power, on your wrist.
4. Power consumption is something SoCs were designed to address. Mobiles, Tablets, Watches and mini computers – they all are supposed to work on as less power as possible. A part of that is because of the problems heat would pose, but it's a win for SoCs nonetheless. An SoC for a tablet would be fine with 12-15 watts doing all kinds of

jobs while a CPU would typically demand more (ranging from 12 to 90 watts).

The power of CPU

The CPU might be bulky, power-hungry and only one component of a full system but it has its own benefits.

When you have too many cores

1. The CPU might not have DSPs but it can have multiple instruction sets. Ever wondered why a system sporting a BitLocker encrypted hard-drive performs nearly as fast as a non-encrypted one while also running programs without trouble? That's because a CPU can have a dedicated AES instruction set to encrypt data on-the-fly while the general-purpose cores are doing their job. All of this while the built-in graphics processor can compute 4K pixels at 30 frames per second for an OS running inside a VM using virtualization technologies built into the CPU. They can also simultaneously compute on arrays of similar data on SIMD cores built right in (SSE4) and run floating point math using the FPU circuits while

managing memory and data input-output from multiple components and peripherals. All that task, simultaneously with each clock tick is no minor feat. SoCs do not implement that many instruction sets and their control system is not designed for such workloads.

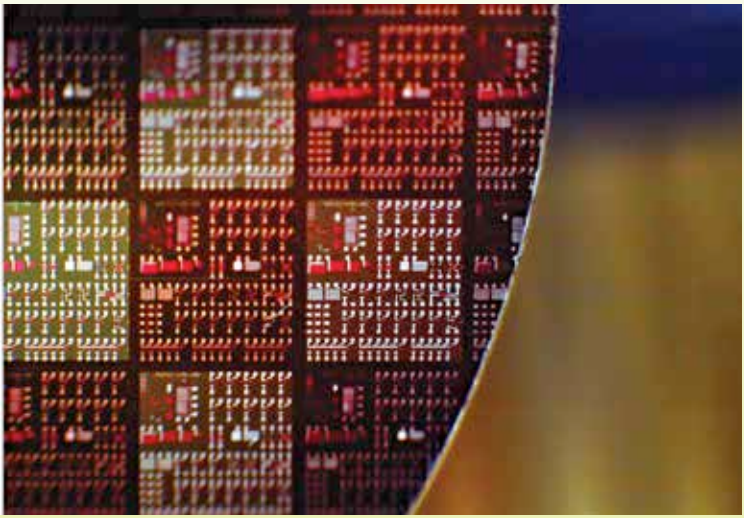
2. Since CPUs can get lots of work done using different instruction sets as well as higher frequencies, they are used to build super-computers and servers.
3. If you have tiny amounts of RAM, it would limit the work the CPU can do. Systems low on memory cannot work on large data sets at fast speed s because suspending data to disk and bringing them back is a non-trivial wastage of time and processing power. Although SoCs contain good amounts of RAM, when you want to create a system that can address terabytes of RAM, you turn to the CPU (*http://www.dell.com/in/business/p/poweredge-r930/pd*). Being able to host and address terabytes of RAM is a luxury SoCs cannot have.
4. CPU is portable in implementation: You can upgrade the system by upgrading to a better CPU without tearing down the rest of the system. Or, you can add or replace RAM and the CPU would work fine.

When both fail

SoCs are full systems and CPUs are beasts around which full systems can be built but they are not ideal for all use-cases.

For example, what would a microwave oven use for controlling heat and timing? A CPU, or a SoC? Probably neither. A simple microwave controller would need a ASIC chip (*https://en.wikipedia.org/wiki/Application-specific_integrated_circuit*) because that would be sufficient to display small amounts of information on a LED screen, run a timer and sound the beeper. Both CPU with its processing power and SoC with myriad functionality would be an overkill. If, however, it was to be a IoT device, an SoC would probably make sense. A car's dashboard system can do without an SoC too unless it is a self-driving car which needs an array of sensing tools and algorithms to drive through traffic. Also, they need GPUs to handle compute tasks.

There are very few computational requirements in today's world which do not (or cannot) use either a SoC or a CPU. Such requirements are either too small or too large and complex to be fulfilled solely by either of the two types. While larger requirements can be solved by large computing clusters with different types of processors (DSPs, GPGPUs and CPUs), smaller ones can be taken care of by ASIC implementations.



THE FABRICATION PROCESS

So, how is the chip that makes many of your gadgets work, made? Take a closer look right here!

Semiconductor device fabrication refers to the process by which integrated circuits present in many gadgets are created. It's a sequence that involves multiple steps. Generally, the entire process from start to having the chips ready to ship would take some six to eight weeks. They are done in specialized facilities called fabs.

A brief history

Before getting into the different steps involved, let's take a look at the brief history of fabrication.

Back when feature widths were way bigger than 10 micrometer, purity wasn't as big an issue as it is today in semiconductor manufacturing. However, as more and more circuits got integrated, the cleanrooms – the spaces of manufacture – became cleaner than before. These days, fabs get pressurized using filtered air so that even the tiniest particles could be removed. These particles, if left unchecked may come to rest on the wafers, leading to defects. Also, the workers in a fabrication facility these days have to wear special suits so that the devices could be protected from human contamination.

During the early days, in the 1960s, device manufacturing was largely an American enterprise – with companies in Texas and California on the forefront. But now, the global business that it is, it has spread to other parts of the world including Asia and the Middle East. It's common enough for the world's leading semiconductor manufacturers to have facilities around the world. For instance, Intel- the biggest manufacturer in the world owns facilities not just in the US but also in Asia and Europe.

Cleanrooms everywhere

Some of the top manufacturers in present-day world include Samsung(Korea), IBM(US), SMIC(China) and Texas Instruments(US) to name a few.

The wafers

A typical semiconductor wafer is made from pure (extremely pure) silicon. This silicon is turned into mono-crystalline cylindrical ingots. The diameter of these ingots could be up to 300 mm. They are sliced to form the wafers. Each wafer would be about 0.75 mm thick.

The steps involved

Processing

In fabrication, the different processing steps come under four catego-

Many steps to get to this point

ries: deposition, removal, patterning and modification of electrical properties.

- **Deposition** - This refers to any process which grows, coats or in some other way transfers a material on to a wafer. Some of the commonly used technologies are chemical vapour deposition (CVD), physical vapour deposition (PVD) and molecular beam epitaxy (MBE).
- **Removal** - This refers to any process using which material is removed from the wafer. An example is the etching processes – either dry or wet.
- **Patterning** - This is the step by which the deposited materials are shaped or altered. It's usually called lithography. In conventional lithography, the wafer gets coated with a chemical. After this, a machine termed 'stepper' would focus, align and move a mask. This exposes particular portions of the wafer lesser than the short wavelength light. The exposed regions would then get washed away with a developer solution. Once the etching (or similar process) gets done, the leftover chemical is also removed using a technique called plasma ashing.
- **Modification of electrical properties** - One of the key elements of this step is doping transistor sources and drains. The doping processes would be followed by furnace annealing. In advanced devices, the process would be rapid thermal annealing. Annealing helps the implanted dopants to be activated. These days, the step also includes reducing a material's dielectric constant in low-k insulators. This is done by exposing to ultraviolet light in UV processing (UVP).

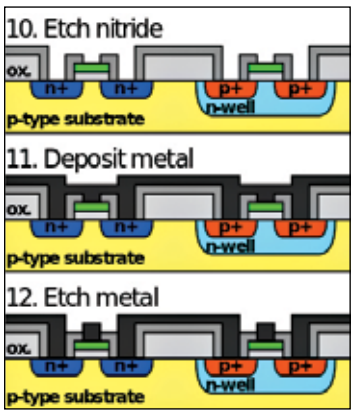
Modern chips could contain up to eleven metal levels. To produce this may require more than 300 sequenced processing steps.

Front-end-of-line(FEOL) processing

This is the process by which the formation of transistors happens on the silicon. The raw wafer, in this case is engineered by growing a highly pure silicon layer that's practically defect-free. This is done by a process called epitaxy. In more advanced logic devices, even before the silicon epitaxy, certain tasks are performed which improve the performance of the transistors that are to be built.

One such method involves introducing a straining step in which a silicon variant like silicon-germanium(SiGe) is deposited. Once that's done, the crystal lattice would become stretched to a certain extent. This would result in a better electronic mobility.

Another method is termed silicon on insulator technology. In this, an insulating layer is inserted between the raw silicon wafer and the thin layer of silicon epitaxy. The advantage of this method is that the transistors created would have lesser parasitic effects.



Deep stuff is FEOL!

Gate oxide and implants

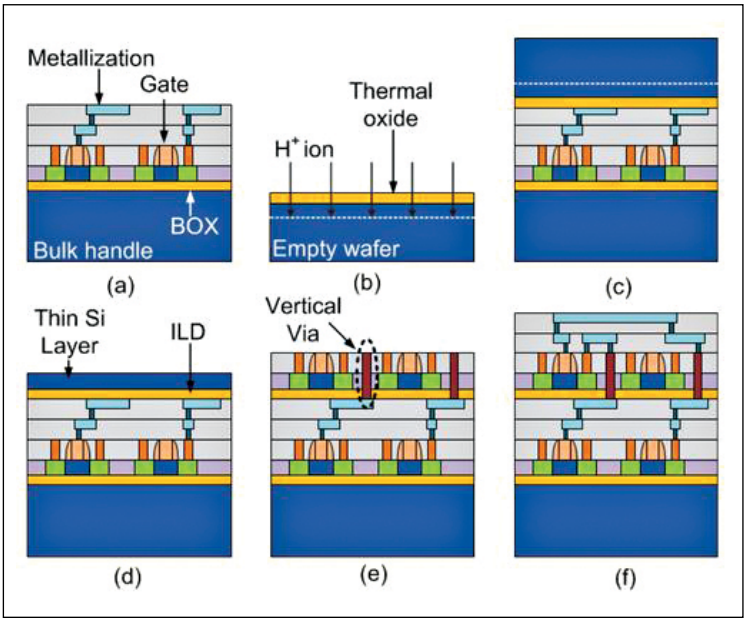
The step next to front-end surface engineering involves growth of the gate dielectric, patterning the gate and also patterning of the source and drain regions. It also involves implantation or diffusion of dopants. This is so that the required complementary electrical properties could be obtained.

In the case of dynamic random-access memory devices, the storage capacitors too are fabricated at this point. They are usually stacked above the access transistor.

Back-end-of-line (BEOL processing)

Metal layers

After the creation of the different semiconductor devices, they need to be interconnected so that the required electrical circuits could be formed. This happens over a series of wafer processing steps that are collectively called BEOL.



Just as the diagrams indicate, kinda technical.

This processing includes the creation of the interconnecting metal wires which are isolated by the dielectric layers. Traditionally, the insulating material has been a form of silicate glass. However, in recent times, the use of low dielectric constant materials have come to fore.

Wafer test

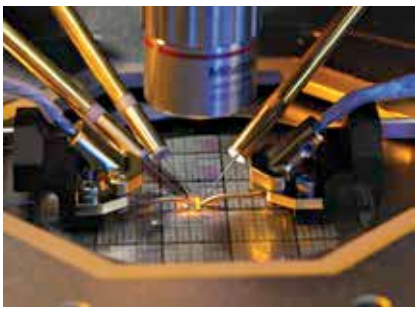
The demand for metrology in between different processing steps has gone up thanks to the heavily serialized nature of water processing. For instance, to have a strict control over the thickness of gate oxide, thin film metrology that's based on reflectometry or ellipsometry is used.

To verify that the wafers didn't get damaged by the processing steps that went before testing, they use the wafer test metrology equipment. If it's found that a large number of dies on a wafer have failed, the whole wafer gets scrapped- this way, the costs of further processing could be avoided. Virtual metrology-which uses statistical methods has been used to predict wafer properties. This negates having to perform any physical measurement.

Device test

After finishing the front-end process, the semiconductor devices undergo a multitude of electrical tests which help determine if they properly function. The proportion of the devices on the wafer that are found to function properly is called the yield. Usually, the manufacturers are secretive about the yields. However, it's said that the yield could be as low as 30%. One of the many reasons for a low yield is process variation.

The chips on the wafer are usually tested using an electronic presser. The presser has tiny probes that would be pressed against the chip. Each bad chip that's identified would be marked with a dye drop. Electronic dye marking is done if the wafer test data gets logged into a central computer database and the chips get sorted into virtual bins pertaining to predefined test limits. The binning data thus obtained could be logged or graphed on a wafer map. This helps trace any manufacturing defects and also to mark the faulty chips. The map could also be used for wafer assembly and packaging.



Testing it, one wafer at a time!

After packaging, the chips are tested yet again. This is because there could be anomalies like missing bond wires or the analog performance getting altered by the package. This test is indeed the final test.

Testing times in a fab could vary from just a few milliseconds to a couple of seconds. The test software is usually optimized to reduce the testing time. Multiple chip or multi-site testing is also a possibility-many testers possess the resources to run nearly all the tests in parallel.

Power on, test!

To facilitate speedier testing and bring down testing costs, chips are usually

designed with testability features like a built-in self-test or scan chains. Some of the designs which use special analog fab processes feature wafers that are laser trimmed during the testing. This is so that tightly-designed resistance values that the design specifies could be achieved.

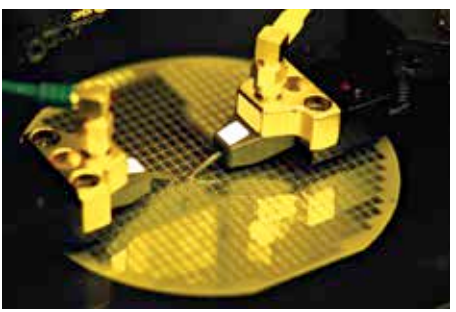
A good design would attempt to test and statistically manage corners-which are extreme silicon behavior brought about by high operating temperature and also other extreme processing steps. Most of the designs would be able to cope with a minimum of 64 corners.

Dicing

After testing, a wafer usually gets reduced in thickness. Then, the wafer gets scored and broken into separate dices. This process is called wafer dicing. When it comes to packaging, only the functioning and unmarked chips get packaged, of course.

Packaging

There are three steps involved in packaging-whether ceramic or plastic: mounting the dies, connecting die pads to pins on the package and then the sealing of the dies. To link pads to pins, tiny wires are used. Such wires, traditionally were composed of gold. This brings about a lead frame involving solder-plated copper. Since lead is poisonous, lead-free lead frames are mandatory these days.



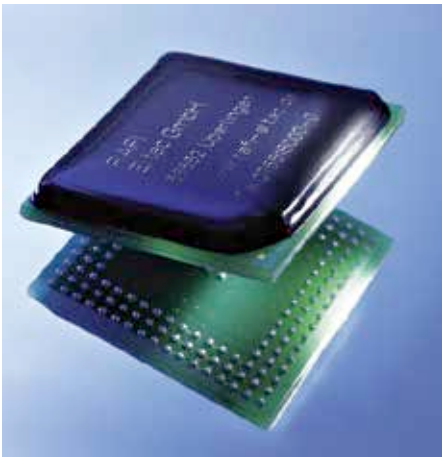
Dicing: no game of dice, but a precise science

Yet another packaging technology is chip scale package or CSP. In the case of plastic dual in-line package, it's many times bigger than the die which is inside. However, CSP chips are almost the same size as the die. This means that a CSP could be constructed for every die before the wafer gets diced.

The packaged chips also get retested. This ensures that the chips didn't get damaged while packaging. It also ensures that the die-to-pin interconnect operation was done correctly. Once the retesting is done, a laser would etch the name and numbers of the chip on the package.

A method of fabrication

Multiple technologies can be used for the manufacturing of SoCs. To detail all of them would be beyond the scope of this article. But since the mode of fabrication gives insight into what goes behind the scene in making a chip, it's worth knowing in detail at least about one key mode of fabrication.



Packing in all that power.

Standard cell

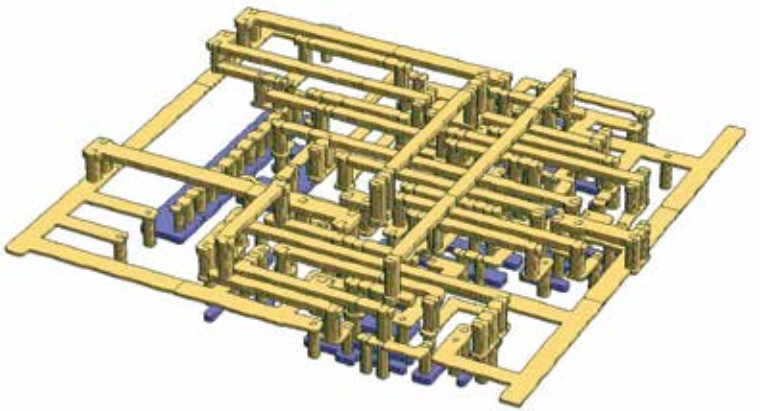
Standard cell methodology is used to design application-specific integrated circuits or ASICs. This incorporates mostly digital-logic features. The method is also a great example of design abstraction- which means a low-level very large scale integration or VLSI layout gets encapsulated in an abstract logic representation (for example, the NAND gate).

Standard cells belong to the general methodology class called cell-based methodology. It demarcates the designers' functions. For instance, one designer could focus on the logical function facet of the design while another would focus on the physical or implementation facet.

Thanks to the standard cell method, designers have been able to scale up ASICs from rather simple and single function ICs that involve many thousand gates to highly intricate SoC devices that include many millions of gates.

How it's constructed?

A standard cell essentially means a bunch of transistors and interconnected structures which provide a Boolean logic function. Examples include OR, AND, XOR and XNOR inverters. Or it could provide a storage function- like in the case of a latch or flipflop. The simplest of these cells represent the elemental NOR, NAND and XOR Boolean function. But that's not to say that cells of a greater complexity aren't commonly used- for example, the D-input flipflop or the 2-bit full adder.



One smart methodology

The Boolean logic function of the cell is called its logical view: the functional behavior gets captured as Boolean algebra equation or a truth table. In some cases, it's captured as a state transition table.

In the normal course, the primary design of a standard cell gets developed at the transistor level. It appears as a schematic view of a transistor netlist. The netlist refers to a nodal description of the transistors, which detail the transistors' connections with each other and also that of the terminal ports with the external environment. A schematic view could be produced using different Electronic Design Automation(EDA) and Computer Aided Design(CAD) programs. These programs generally come with a Graphical User Interface(GUI) that aids the generation of this netlist.

Additional CAD programs could also be used by designers-like Spectre of SPICE which could effectively simulate the netlist's electronic behavior. This is done by declaring the input stimulus(current waveforms or voltage) following which the circuit's time domain or analogue response is calculated. Using the simulations, it can be verified if the netlist implements the required function and also predict other relevant parameters. Examples of such parameters include signal propagation delay and power consumption.

It should be noted that the logical as well as netlist views are useful only for abstract or algebraic simulation, but not for device fabrication. This means that the standard cell's actual physical representation must also be designed. As far as common design practices go, this view- called the layout view is the lowest level of design abstraction.

The VLSI layout

From a fabrication standpoint though, the most significant view is the standard cell's VLSI layout. This is because it's the closest that things get to a manufacturing blueprint for the cell.

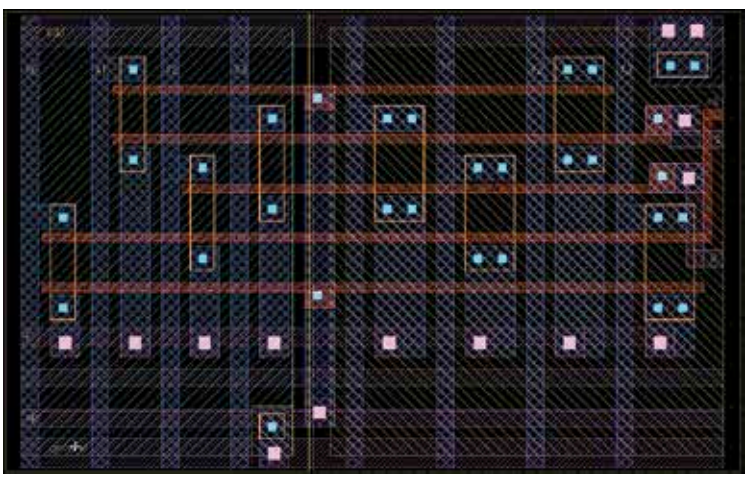
The layout typically has base layers that correspond to the various structures of the transistor devices. It will also have the interconnect wiring layer as well as 'via layers'- they club together the multiple terminals of the transistor formations. Generally, the interconnect wiring layers are numbered. They would also give precise via layers that represent connections between specific sequential layers.

In this layout, it's possible that non-manufacturing layer may be present as well. This is for the purpose of design automation. However, multiple layers that are used specifically for Place and Route(PNR) CAD programs too generally get included in a different yet somewhat similar abstract view.

More often than not, the abstract view has significantly lesser volume of information compared to the layout. It may also be recognized as a Layout Extraction Format (LEF) or equivalent file type.

The Library

A standard cell library refers to a collection of low-level logic functions like OR, AND, latches, flip-flops, INVERT and buffers. These cells get realized as fixed-height but variable-width completely custom cells. With these



A layout to put things in perspective

libraries, the main facet is that their height is fixed- meaning, they could be placed in rows, thereby making the process of automated digital layout easier. Generally, the cells are optimized full-custom layouts. This means delays as also the surface area would be minimized.

Two of the key components that can be found in a typical standard-cell library are given below:

1. Library Database

This database contains multiple views. These often include schematic, symbol, layout, abstract and also other logical or simulation views. Using the database, different information could get captured in different formats including the Synopsys Milkyway and cadence LEF formats. These would have reduced information regarding the layouts which would be enough for automated "Place and Route" tools.

2. Timing Abstract

This abstract gives functional definitions as well as information regarding power, timing and noise for every cell. Usually, the abstract is found in Liberty format.

Some additional components that could be found in a standard-cell library are given here:

- The complete layout of the cells
- The cells' spice models
- Verilog models
- Parasitic Extraction models
- DRC rule decks

Application

Now, let's take a look at the application of standard cell.

It's safe to say that a 2-input NOR or NAND function is more than enough to form any arbitrary Boolean function set. However, in modern ASIC design, there's a general practice of using standard-cell methodology using a sizable library- or maybe multiple libraries-of cells.

Usually, the library consists of multiple implementation of the same logic function. The differences would be in speed and the area it takes up. The variety has the advantage that it improves the efficiency of the tools for automated synthesis, place and route(SPR). Also, the designer gets a higher freedom to perform implementation trade-offs.

A full group of standard-cell descriptions commonly go by the name of 'technology library.'

Technology libraries are used by commercially available Electronic Design Automation(EDA) tools to automate multiple processes. These include the synthesis, placement and routing of a digital ASIC. It's the foundry operator that develops and distribute the technology library. The library forms the basis to exchange design information between the various phases of the SPR process.

The phases in the SPR process

Synthesis

The Logic Synthesis tool uses the technology library's cell logic view to mathematically transform the register-transfer level(RTL) description of the ASIC into a technology-dependent netlist. This is so much similar to how a software compiler would transform a high-level C-program listing into a processor-dependent assembly-language listing.

It even looks like a brick and mortar library's blueprint!

The netlist forms the ASIC design's standard-cell representation at the level of the logical view. Aside from instances of the standard-cell library gates, it also contains port connectivity between gates. Use of the right synthesis techniques would ensure that the mathematical equivalency exists between the synthesized netlist as well as original RTL description. No umapped RTL statements and declarations could be found in the netlist.

The process of transforming the C-level models description to technology-dependent netlist is performed by the high-level synthesis tool.

Placement

The physical implementation of the ASIC is initiated by the placement tool. With the aid of a 2-D floorplan that's provided by the ASIC designer, locations for each gate in the netlist are assigned by the placer tool. The result would be a place gates netlist.

This netlist would include the physical location of all the standard cells in the netlist. However, it does have an abstract description of the way the gate's terminals are wired to each other.

Usually, the standard cells would feature a constant size in at least one dimension. This would enable them to be lined up in rows on the integrated circuit. The chip would contain a large number of rows. The power and ground would run next to each row. Each of the rows would be filled with the different cells which make up the actual design.

Placers follow some rules as well. For instance, a unique location is assigned to each gate on the die map. Once a given gate is placed, it shouldn't occupy or overlap the location of another gate.

Routing

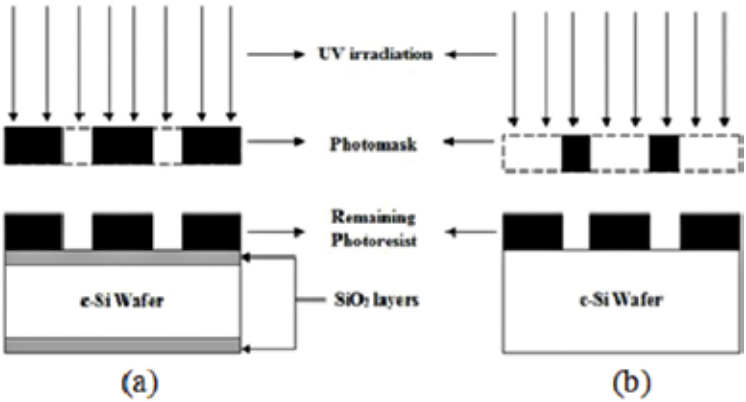
The router adds the signal connect lines and the power supply lines using both the placed-gates netlist and the library's layout view. The completely routed physical netlist would have the listing of the gates from synthesis, each gate's position from placement and from rioting, the drawn interconnects.

The verification processes

Two verification processes that are used are Design Rule Check(DRC) and Layout Versus Schematic(LVS). A rigid observance of transistor spacing, power density rules and metal layer thickness is needed for device fabrication at deep-submicrometer(0.13 0.13 µm and less). With DRC, an exhaustive comparison of the physical netlist against a set of foundry design rules happen. If any violations are noted, they are flagged.

With the LVS process, it can be confirmed that the layout indeed has the same structure as the associated schematic. Usually, this forms the last step in the layout process.

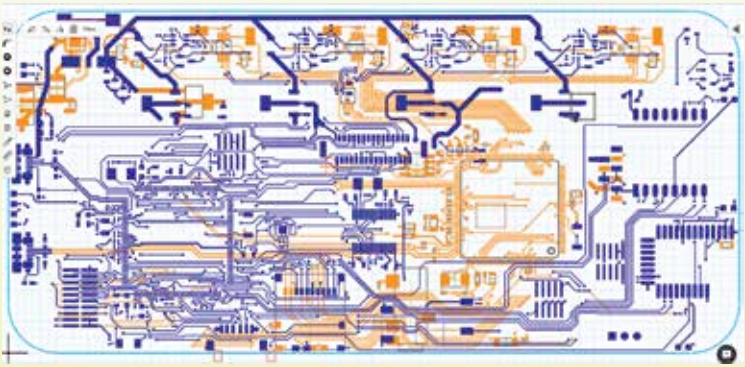
A schematic diagram as well as the view extracted from a layout form the input of the LVS tool. Then, a netlist is generated from each one, after



SPR process- quite an important process

which they are compared. Everything from device sizing to ports and nodes get compared. If they are found to be the same, LVS would then pass them and the designer can continue.

With the LVS tool, transistor fingers are considered as the same as an extra-wide transistor. This means that four transistors (each 1 µm wide) that are arranged in parallel, a four finger 1 µm transistor and even a 4 µm transistor are all considered as the same. The functionality of the .lib files would be procured from SPICE models which would then be added to the .lib file as an attribute.



ARCHITECTURE AND SoCS

A look at various CPU architectures and current generation of SoCs based on them

The first step in making an SoC is generally assembling various elements of hardware and their software drivers together. To do this, the architecture, which involves component placement and circuits, needs to be defined.

There are a number of architectures out there but only a handful of them are popular today. And even though SoCs keep getting updated regularly, most of the architectures they are based on are years, if not decades old.

Let's take a look at the popular architectures in use today.

ARM

The multinational semiconductor company based out of England are the leading manufacturer of mobile processors and SoCs in the world.

ARM originally stood for Acorn RISC Machine and was later renamed to Advanced RISC machines.

RISC (Reduced Instruction Set Computing) based designs require less transistors than the conventional CISC (Complex Instruction Set Computing) based designs that power most of the personal computers. They are more power efficient and compact making them the ideal choice for smartphones, tablets and other such embedded systems.

More than 100 billion ARM processors have been produced up to 2017 making ARM the most popular ISA (Instruction Set Architecture) in terms of number.

The ARM architecture

The latest generation of ARM architecture is the ARMv8-A. A significant improvement over the last version, the v8 adds an optional 64-bit architecture to it.

The latest version of processors based on this architecture come with a number of advancements and added features including:

- Enhanced memory models increasing the efficiency of the chip
- Architecture based on big.LITTLE configuration allowing two different cores with different clock speed to work together on a single processor
- New instruction set called AArch64 that comes with a new exception system, Advanced SIMD enhancements and a 48-bit virtual address space based on LPAE (Large Physical Address Extension) capable of extending it to 64-bit

ARM processors dominate the mobile SoC market currently.

- New RAS (Reliability, Availability and Serviceability) features
- Enhanced security features including something called Privileged Access Never state, that prevents unauthorised access to user data

The 32-bit ARM architecture is supported by a number of operating systems including all major mobile platforms like Android, iOS, Windows 10 mobile, Chrome OS, and Firefox OS.

Many desktop Linux distributions including Gentoo, Ubuntu, BSD and OpenSolaris are also supported.

Most of the current generation SoCs used in mobiles are ARMv8 based including the Apple A-x series, Samsung's Exynos series and Qualcomm's Snapdragon series.

Intel x86

Intel, founded by the famous Gordon Moore and Robert Noyce, is the world's largest chip manufacturer. They're also the inventor and proprietor of one of the most common and ubiquitous architecture, x86 and microprocessors based on it.

The x86 architecture encompasses the whole family of CPUs by Intel. Getting its name from Intel's breakthrough 8086 CPU, it was initially developed for embedded systems and small single-user personal computers. More than 40 years later, the x86 architecture is everywhere, powering laptops, netbooks, personal computer, servers, midrange workstations, portable devices and even supercomputers.

The main difference between the x86 architecture and the rest on the list is that x86 is CISC based compared to RISC architecture on which the rest are based. Generally, CISC based processors are more powerful but not efficient (check the CISC vs RISC box for more information).

x86 based Intel SoCs for mobile devices started coming out in 2012. Intel announced it's partnership with Google to extend support for its Android OS on its x86 platform, in the same year.

Intel tried to compete with ARM architecture based SoCs with its Atom series. The chip gained initial attraction but after a couple of phones from ASUS that definitely had the potential, enjoyed mild sales. Intel realised that it couldn't compete with ARM even if it enjoyed the status of being one of biggest technological company, at least not in the mobile market segment.

So, in 2016 Intel decided to exit the smartphone market and cancelled its next generation of x86 based SoCs codenamed the SoFIA and Broxton.

Maybe Intel decides to enter the mobile SoC market again, when it is ready with its 10nm fabricated processors, but right now x86 based SoCs are dead.

CISC vs RISC	
More hardware-oriented	More Software-oriented
More complex instructions, with more sizes and formats	Fewer 'Reduced' instruction set
Accesses memory directly and hence has fewer registers	Doesn't access memory directly and uses many registers.
More addressing modes	Fewer addressing modes
CISC uses microprogramming extensively and hence coding CISC processors is relatively easier	Complexity lies in the compiler, hence coding RISC processors is difficult and requires more lines of codes
Different instructions take different amount of cycles/ time.	All instructions take a single cycle to execute

Super-H

Designed by Hitachi in the early 1990s, the SuperH processor family was based on RISC and was initially designed to power Sega's gaming consoles of that time, namely the Sega Saturn and the Sega 32X.

At the time of its launch, the SuperH architecture had a number of innovations to its name, being one of the first processor to support a 'Thumb-like' instruction set and native hardware support for MAC operations.

Hitachi launched a couple of SoCs and processors based on this architecture which were at the heart of all the Sega consoles and arcade machines of that time.

More than 35 million devices running on SuperH were sold between 1994 and 1996, leading SuperH to capture more than 32% share, a significant amount, of the global microprocessor market by the end of 1996.

The most prominent of chips manufactured based on this architecture were the following

- SH-1: The first commercial processor based on the SuperH architecture, the SH-1 powered Sega's semi-popular console, the Saturn.
- SH-2: SH-2 became more popular and was not only used in Sega's next generation of consoles, the 32X and the later version of Saturn, it was also used in a number of other devices including Engine Control Unit.

- SH-3: This chip was popular in the car navigation devices and also powered up a lot of handheld Windows CE devices.
- SH-4: This iteration of the SoC was widely popular in consoles and arcade systems. It powered Sega's Dreamcast console and various arcade systems including the NAOMI, NAOMI 2, and Hikaru.
- SH-5: The SH-5 wasn't used in any gaming console but powered up a number of 64-bit multimedia devices..

The SuperH chips began lagging in the SoC race by the dawn of the 21st century, and became less popular in consoles. It was still used in variety of other electronic devices at the time, including CD players.

Hitachi stopped manufacturing these chips a long time ago, and currently Renesas, a Japanese semiconductor company is manufacturing SuperH chips. Currently, most of the patents on the design and architecture of the SuperH chips have expired making it a suitable candidate for open source hardware.

SuperH chips powered the yesteryear generation of consoles including the Sega Saturn and Dreamcast.

A new CPU was designed, reinventing and implementing the SH2, called the J2. Unveiled first at LinuxCon 2015 in Japan, the developers of the chip cited numerous reasons why they chose the SuperH architecture, including:

- Extremely low fabrication costs
- No patents
- Supports a number of existing compilers and operating systems including Linux and Windows Embedded

- A higher code density than other 32-bit RISC processors in the market

SPARC

SPARC or Scalable Processor Architecture was a RISC based Instruction Set Architecture, developed by Sun Microsystems in the year 1987.

A separate company dubbed the SPARC International was established in 1989 to promote the use of the architecture and manage the SPARC trademarks and licenses.

It was licensed to a number of manufacturers including Texas Instruments and Fujitsu. Currently its design is fully open source and anyone can use it without having to pay any royalty.

The 'Scalable' part in SPARC is there because its architecture allows one to implement it on any scale, from embedded processors to large mainframe/server processors, with each based on the same core instruction set.

A number of SoCs based on the architecture have been manufactured by a number of companies, powering up workstations and servers. Currently two companies, Fujitsu and Oracle, are developing SoCs based on the SPARC architecture.

Oracle, the current owner of Sun Microsystems, designed M7 from ground up. It was first announced in 2015 to power high end servers. The

Oracle bought Sun Microsystems and built M7, a new SPARC based processor from the ground up.

systems are powered by 32 core, 256 thread SPARC microprocessors, and have many new advancements which include the following features:

- Enhanced and secure memory protection technology called the "silicon secured memory", to stop malicious programs from accessing parts of the memory they shouldn't
- Hardware assisted encryption built into all the 32 cores
- Coprocessor capable of offloading functions and tasks from main core to speeds up a variety of tasks and critical functions like decompression, memory scan and range scan.

POWER

Named after an old microprocessor instruction set designed by IBM in the late 1980's, POWER stands for Performance Optimisation With Enhanced RISC. It was designed and manufactured by a number of companies including IBM, LSI and e2v in 1990's. There exists a governing body called Power.org, which consists of more than 40 companies and organisations to regulate licenses and trademarks of the Power architecture.

IBM came out with the first generation of POWER ISA from which the POWER PC instruction set was derived later. The term 'Power architecture' includes all generations of chips based on the POWER IBM architecture and the PowerPC processors.

The Power architecture chips were behind all the 7th generation consoles including the PlayStation 3 and the Xbox 360. The chips were also at the heart of many supercomputers. The TOP500 list of supercomputers in June showed that at least 22 of the top 50 fastest supercomputers in the world used Power chips somewhere in their design

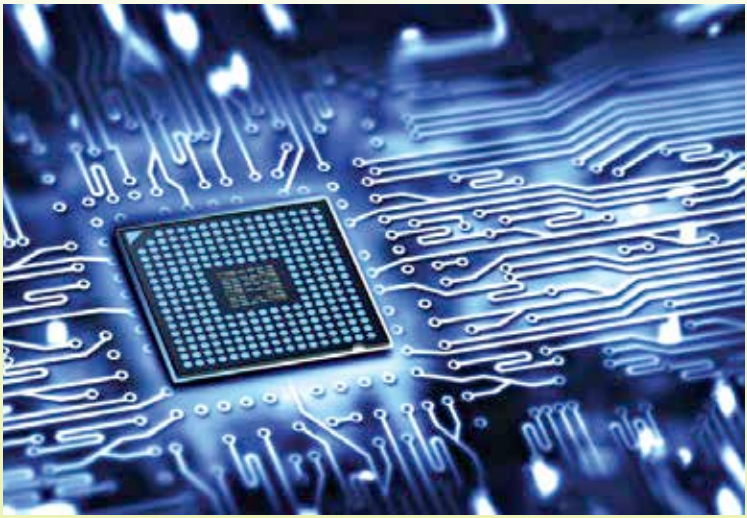
IBM came out with its latest generation of processors based on the POWER architecture called the IBM POWER8, designed specifically for today's computing world of big data and cloud computing. The multi-threaded chip comes in 4, 6, 8, 10 and 12 core variants, where each core can

POWER architecture has been used in building almost everything, from gaming consoles to supercomputers.

handle eight hardware threads at the same time. The POWER8 has a number of features which distinguishes it from its competitors. These include:

- First chip designed by IBM specifically for the big data and cloud computing market to handle vast amounts of data, processing it and storing it.
- Four times faster than the previous generation of POWER based chips, with four times the number of threads per core than the last generation
- 170% more bandwidth
- Linux-based ecosystem making it idle for big data and cloud computing solutions
- CAPI (Coherent Attached Processor Interface) port support that enables the connection of GPUs, flash memory and FPGAs directly to systems
- Compatibility to scale from the current generation DDR3 memory controller to the DDR4 standard with the help of an external component acting as a memory buffer called 'Centaur'

IBM partnered with NVIDIA for their POWER8 chip to add hardware based Java acceleration on the GPU, speeding up JAVA code by up to eight times. IBM has also been releasing servers and computers based on the latest POWER8 chip since its announcement in 2014.



POPULAR SOCS

So many manufacturers. So many SoCs.

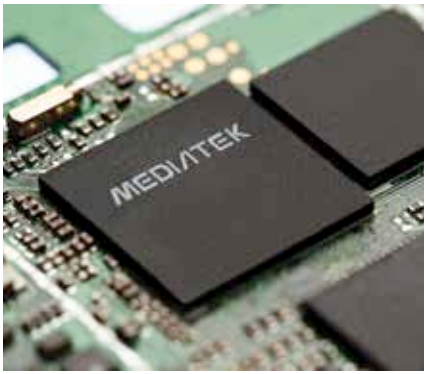
Today's SoCs are powerful. And by powerful we mean quad core processors and oodles of RAM; so much so that they're even more powerful than the PCs of just ten years ago.

There are a number of SoC manufacturers out there like Mediatek, Qualcomm, Snapdragon and Intel who supply SoCs to phone manufacturers and there are even companies like Samsung and Apple who make their own SoCs.

There are a number of parameters that define the performance of these SoCs including their architecture, number of cores, graphic processor and CPU. Let's take a look at the current generation of SoCs by various manufacturers and try to determine which ones fair the best (at least on paper).

MediaTek

A Taiwanese semiconductor company, MediaTek has been designing and fabricating SoCs for devices like smartphones, HDTVs and Blu-ray players since the last two decades. It was the third largest IC designer in the world last year and has a number of innovations to its name.



MediaTek was the first company to come out with a 10 core processor for mobiles.

MediaTek Helio X30

The latest generation offering from MediaTek, the Helio X30 is a top of the line SoC fabricated using a 10nm process, consisting of 10 cores. So now whenever your phone wants to do some task, it will select the most apt and efficient CPU cluster out of the three according to the need. The chip also features something called a Vision Processing Unit – a dedicated camera assisting hardware. It works in tandem with the DSP/ISP to take care of various camera related features and computations, freeing the CPU and GPU of this intensive task. The chip supports upto 8 GB LPDDR4X memory clocked at 1866 MHz and runs a custom version of the PowerVR 7XTP-MT4 GPU. The first smartphone to use this chip was recently announced by a Chinese smartphone manufacturer called the Vernee. Running on 10 cores, the device is called the Apollo 2. Check out those clock speeds:

Primary CPU	dual-core ARM Cortex-A73 up to 2.5GHz + quad-core ARM Cortex-A53 up to 2.2GHz + quad-core ARM Cortex-A35 up to 1.9GHz.
CPU architecture	Advanced tri-cluster architecture
CPU structure	RISC
DSP type	Imagiq Gen 2.0
GPU	IMG PowerVR 7XTP-MT4
Year Released	2017
Devices	Vernee Apollo 2

MediaTek Helio X27

Featuring an ultra-powerful 10 core processor, the Helio X27 was the chip that powered the last generation of smartphones and is still in use. The SoC was designed keeping phones with dual cameras in mind and comes with native support for colo and mono sensor, capable of capturing three times the light compared to other SoCs using traditional sensors.

The deca Core CPU features another embedded processor, ARM Cortex M4, to take care of always-on background applications like music player.

Thanks to the three cluster architecture, the SoC is power efficient, delegating tasks to different clusters depending on how intensive they are.

Primary CPU	dual-ARM-A72 @ 2.6GHz + quad-ARM-A53 @ 2.0GHz + quad-ARM-A53 @ 1.6GHz
CPU architecture	Tri-Cluster Architecture
CPU structure	RISC
DSP type	-
GPU	ARM Mali-T880 MP4
Year Released	2016
Devices	Elephone S7, S8, UMi Z, LeEco Le Pro 3

Qualcomm Snapdragon

Qualcomm is an American semiconductor and telecommunication equipment company that has its own range of specially designed SoCs for mobile products. Qualcomm was the first company to launch a SoC featuring a 1 GHz processor for mobile phones almost a decade back in the year 2007. The latest generation of Qualcomm processors are the state of the art 800 series which feature quad and octa core CPUs.

Snapdragon 835

The Snapdragon 835 is the next generation of SoC that powers Samsung's latest flagship model, the Galaxy S8. It the first 10nm processor made by Qualcomm, a trend that its competitors are following with their latest SoCs too. The new 10 nm process makes the chip upto 30% smaller and leads to a lower power consumption and a higher performance. Sporting an Octa core processor in the big.LITTLE architecture, 4 performance oriented cores are clocked at 2.45 GHZ while the remaining 4 cores are clocked at 1.8

GHz and are more power efficient. Featuring the latest Adreno 540 GPU, the SoC is capable of rendering 4K UHD videos and is even compatible with Google Daydream.

The chip also comes with inbuilt support for a number of other features including eye based scanning systems (iris or retina) and better security settings.

Primary CPU	Kryo 208 octa core 4x2.45 GHz + 4x1.8 GHz
CPU architecture	ARM big.LITTLE
CPU structure	RISC
DSP type	Hexagon 682 DSP
GPU	Qualcomm Adreno 540 GPU
Year Released	2016
Devices	Samsung Galaxy S8

Snapdragon 820 and 821

The current generation of SoCs by Qualcomm is the Snapdragon 820. This chip is the one that powers a number of flagships including HTC 10, Samsung Galaxy S7 and Note 7, Xiaomi Mi 5 and the LG G5.

It has 2 cores clocked at 2.2 GHz and the remaining 2 clocked at 1.6 GHz .The design of the chip is based on the Harvard architecture because of which the CPU has a separate L1 cache (instruction and data both) which results in slightly better performance.

The Galaxy S8 is powered by the Snapdragon 835.

They both pack quad-core processors but this does not necessarily mean that these devices are slower. The number of cores don't matter that much in smartphones since most of the applications aren't designed to optimally use all the cores of the CPU. Also there exist very few resource intensive and demanding applications that will benefit from additional cores.

In fact, the high powered quad core processor, couples with the Adreno 530 GPU makes for a formidable duo, leading to the Qualcomm 820 getting the highest Antutu benchmark score of 136,383.

The Snapdragon 821 is the newest entrant in the market from Qualcomm and is a slightly more powerful version of the 820. It has a 10% improvement in performance from its predecessor and is more energy efficient. It is the same chip with same architecture featuring a slightly more powerful processor; the clock speed of the 2 higher powered cores has been increased from 2.2 GHZ to 2.4 GHz and the lower powered cores have been given a boost from 1.6 GHz to 2 GHz

Primary CPU	Quad-core CPU + 4x Qualcomm Krait 450/400 CPU
CPU architecture	ARM big.LITTLE Harvard
CPU structure	RISC
DSP type	Hexagon 680 DSP
GPU	GPU Qualcomm Adreno 530 GPU
Year Released	2015
Devices	Xiaomi Mi 5, Samsung Galaxy Note , S7, LG G5, HTC 10

Snapdragon 810

The Snapdragon 810 also features a 8 core CPU and is based on the Harvard architecture. Paired with a Adreno 430 GPU, the 8 core CPU consist of 4 high powered cores for intensive tasks while the remaining 4 low powered core are reserved for less CPU intensive tasks.

Primary CPU	Quad-core CPU + 4x Qualcomm® Krait™ 400 CPU
CPU architecture	Harvard
CPU structure	RISC
DSP type	Hexagon QDSP6
GPU	Qualcomm Adreno 430
Year Released	2015
Devices	Sony Xperia Z5, Z5 compact, Z3+, Xperia Z4 tablet Motorola Droid Turbo 2

Samsung Exynos

Samsung, used to rely on Qualcomm's Snapdragon series of processors until the year 2010. Then they developed their own range of ARM based SoCs, designed and manufactured by them, based on ARMv7 and ARM v8 architecture. This series of processors was dubbed as the 'Exynos'.

Exynos 9 Octa 8895

The Exynos 9 Octa 8895 is fabricated using a 10nm process, following in the footsteps of its competitors. It is 27% faster than its predecessor and is almost 40% more power efficient. The 8 cores are divided into 2 clusters, one consisting of 4 custom CPU cores responsible for handling the more intensive tasks, and the remaining 4 Cortex-A53 for less resource intensive tasks.

There are a number of other features packed in the chip including a Mali-G71 GPU that is capable of recording and playing videos in 4K UHD at upto 120fps (well these are some mind boggling figures). It comes with a DPU for features such as the iris and fingerprint scanner.

Primary CPU	4 x 2.5GHz Exynos M2 + 4 x 1.7GHz Cortex-A53
CPU architecture	Custom
CPU structure	RISC
DSP type	VPU
GPU	ARM Mali-G71 MP20
Year Released	2017
Devices	Samsung Galaxy S8

Exynos 8 Octa 8890

It was the first SoC by Samsung to feature its own custom CPU core. Built using a 14nm process. 4 out of the 8 cores are custom designed by Samsung based on the 64 bit ARM v8 structure and are used for resource intensive tasks, while the remaining 4 cores are standard Cortex-A53 cores, that are more power efficient but less powerful making them suitable for various common tasks. Coupled with one of the latest Mali GPUs, the device also supports 4k UHD video recording and playback.

Primary CPU	4x A53@1.586GHz + 4x Exynos M1 @ 2.60GHz (1-2 core load) , 2.29GHz (3-4 core load)
CPU architecture	big.LITTLE
CPU structure	RISC
DSP type	ARM NEON
GPU	ARM Mali-T880
Year Released	2016
Devices	Samsung Galaxy S7, galaxy S7 Edge

Huawei

Huawei is a leading Chinese manufacturer of smartphones and the largest manufacturer of telecommunication equipment in the world. HiSilicon is the branch of Huawei that manufactures SoCs for its smartphones and other devices. HiSilicon has purchased licenses from ARM for its cortex series of processors and Mali series of GPUs, which it uses on its SoCs.

Kirin 960

Announced early last year, Kirin 960 is the latest generation of high powered SoCs from Huawei which will power its latest flagship phone, the Mate 9. Based on the same big.LITTLE architecture that most of its competing SoCs are based on, the Kirin 960 is powered by a Cortex-A73 octa core CPU and a Mali-G71 graphics processor. Faster, smaller, more powerful and more efficient than its predecessor, the SoC has 4 high powered cortex-A73 cores clocked at 2.4 GH to take care of intensive applications like gaming and audio/video editing, and 4 lower powered Cortex-A53 clocked at 1.8 GHz which consume less power and re used for various basic tasks.

The graphics performance of the chip is taken care of by the Mali-G71 MP8 GPU which is the first GPU is the new range of ARM's Bitfrost architecture based graphics processor. The 960's CPU is 15% more efficient than its predecessor's while the GPU featured on it is 20% faster.

Benchmarking tests reveal that Kirin 960 is more powerful than even the Apple A10 Fusion chip in multi core processing (Apple beat it at single core processing). Giving tough competition to other top of the line SoCs like Snapdragon 821 and Samsung Exynos 8890, the Kirin 960 packs a number of features like a faster UFS 2.1 storage chips that is capable of encoding and decoding HEVC videos at 4K resolution.

Primary CPU	4 x ARM Cortex A73 @2.4 GHz +4 x ARM Cortex A53 @ 1.8GHz
CPU architecture	ARM big.LITTLE
CPU structure	RISC
DSP type	Tensilica HiFi Audio DSP 4
GPU	ARM Mali-G71 MP8
Year Released	2016
Devices	Huawei Mate 9

The latest Kirin 960 gives tough competition to other top of the line SoCs like the Snapdragon 821 and the Exynos 9

Kirin 950

Unveiled first in the year 2015, the Kirin 950 was the first SoC to use ARM's Cortex A72 processor in a smartphone. The SoC was at the heart of the most of the high and mid segment phones of Huawei that came out last year. Huawei's flagship phone of last year, Mate 8 was powered by the Kirin 950 and its statistics showed that Huawei's home brewed SoC gave a tough competition to Qualcomm and Samsung.

Primary CPU	4 x ARM Cortex A72 @2 GHz +4 x ARM Cortex A53
CPU architecture	ARM big.LITTLE
CPU structure	RISC
DSP type	Tensilica HiFi Audio DSP 4
GPU	Mali-T880 MP4 @ 900MHz
Year Released	2015
Devices	Huawei Mate 8

Intel Atom

Founded by Gordon Moore (remember Moore's law?) and Robert Noyce, Intel is the world's largest semiconductor chip maker. Credited with inventing

the x86 line of microprocessors, the electronic giants have been the reigning champion of microprocessors.

The last two decades saw the tremendous growth of smartphones and other portable entertainment devices. Realizing this, Intel came out with its own SoC platform dubbed the Atom. Intel entered an official partnership with Google in the year 2011 to support the Android OS on its x86 processors. Intel knew it was going to be tough to compete with already established giants like Qualcomm and Samsung, but when you are the world's biggest semiconductor chip manufacturing company, you don't have to fret much.

Intel adopted its range of x86 processors it was using for netbooks, for mobile devices and worked on reducing their power usage and increasing the efficiency.

A late entrant in the game, Intel tried to compete with the market leaders in this sphere, but ended up losing. Very few smartphones and tablets came out with the Intel Atom processors with the most notable being the ASUS Zenfone.

After investing billions of dollars and suffering huge losses, Intel shut down its mobile department last year to shift its focus on other prospects.

Intel tried hard to break into the mobile SoC market with its Atom Chips but ended up failing.

Intel Atom x3 C3130

Powered by a Dual Core 64 bit processor, clocked at 1 GHz, the Atom SoC showed good performance even though it had comparatively lower specs on paper. This was probably because of the CPU structure used by Intel.

Primary CPU	Dual-core 64-bit Atom X3 @ 1GHz
CPU architecture	x64
CPU structure	CISC
DSP type	-
GPU	ARM Mali 400MP2
Year Released	2015
Devices	Positivo First S410, Dragon Touch S8, Cherry Mobile MAIA FONE i4

Apple

Apple has been designing their own SoCs to power their own devices for quite some time now with the latest iteration A10 being one of the most powerful chips on the market currently. Designed in-house specifically for iPhones, these chips are manufactured either by Samsung or TSMC (Taiwan Semiconductor Manufacturing Company Limited).

A10 Fusion

A10 Fusion is the SoC that powers the iPhone 7 and 7 Plus and is the first quad core chip produced by Apple. The CPU features 2 high powered cores which are used for demanding applications and tasks like gaming and video editing, while the remaining two less powerful and more energy efficient cores are reserved for other menial tasks like email, texting and editing/ creating documents. These cores are setup in a configuration similar to the ARM big.LITTLE architecture.

Apple A10 Fusion is one of the fastest SoCs

Though a major difference between the A10 Fusion and other SoCs based on big.Little architecture like the Snapdragon 820, is that only one pair of cores (either the high powered or the low powered) can be active at one time making the A10 a dual core chip effectively.

According to Apple's claims, their latest chip is 40% faster and consumes 20% less energy than its predecessor, the Apple A9.

The GPU featured on the A10 is a 6 core chip that is 50% faster than its predecessor and consumes only 70% of the power.

Primary CPU	4x Apple Twister cores
CPU architecture	ARM big.LITTLE
CPU structure	RISC
DSP type	-
GPU	Improved PowerVR GT7600
Year Released	2016
Devices	iphone 7, iphone 7 plus

A9

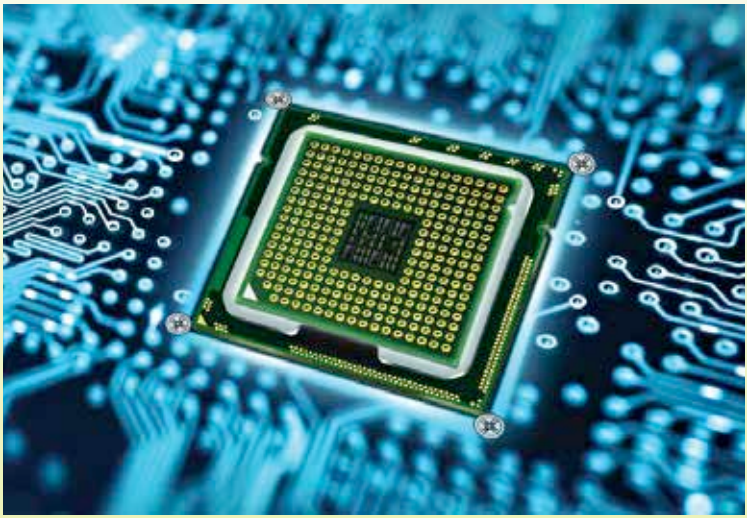
The A9 is a 64 bit system on chip which was used in the iPhone 6S and 6S Plus. Launched in 2015, it gave a tight competition to the Snapdragon 820 and Samsung Exynos 8890. The SoC featured an ARMv8-A based dual core CPUs clocked at 1.85 GHz codenamed twister. It also featured an updated image processor and an embedded motion coprocessor that services the accelerometer, gyroscope, compass and barometer.

Primary CPU	Twister @ 1.9GHz
CPU architecture	ARM big.LITTLE
CPU structure	RISC
DSP type	-
GPU	PowerVR 7XT GT7600
Year Released	2015
Devices	iphone 6S, iphone SE

Wrapping up

Qualcomm, Samsung, MediaTek and Huawei are the market leaders currently with Intel out of the race. If you compare their top of the line SoCs, and take a look at the specs on paper, you'll realize that they are almost similar. Even in the real world, almost all the SoCs fare closely, leading in some department while lagging in another.

Technology is advancing every day and these companies will continue to innovate with faster and more power efficient processors. Phones using the latest generation of SoCs (Snapdragon 835 vs Helio X30 vs Exnos 9 Octa 8895 vs Apple A10 Fusion Core) fabricated using 10nm process, are still scarce in the market. There is no clear winner in this race, not now at least, and we have to wait for phones that can fully exploit the potential of these state of the art SoCs.

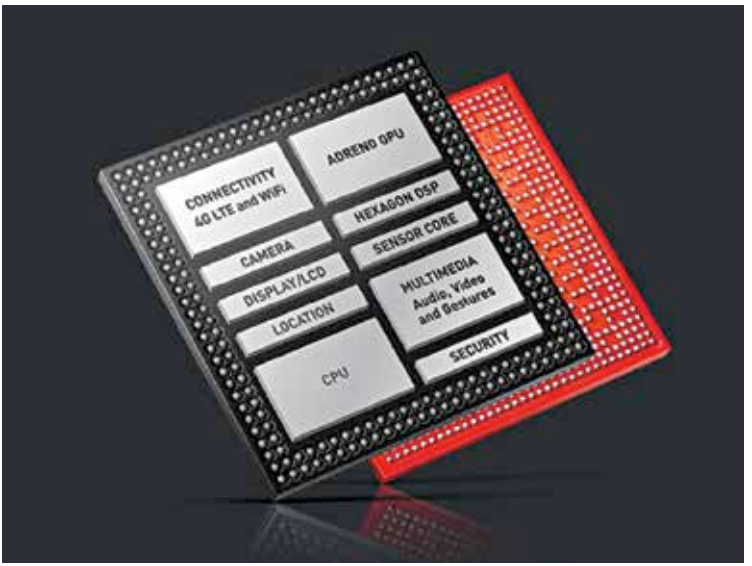


IP AND SOC

When building an SoC, you need to protect the IP from infringement. Here's how manufacturers do it.

In the modern era, there has been an immense demand for complex and highly multifunctional processor chips. This increased processing ability, which is highly sought after by consumers, requires a lot of effort in research and development. However, developers are usually reluctant to spend a high amount of resources and time in developing the technologies housed in these state-of-the-art chips. The only way to make consumers happy, while at the same time minimizing costs, is a hierarchical approach.

The trend of designing a chip from scratch has been replaced with system-on-chips (SoCs), which is seen by many as an inevitable solution to the productivity issues that plagued previous developments. SoCs essen-



A processor chip from Qualcomm

tially integrate the complete components of a previous end-product and assimilate them into a single chip. These reused components form part of an intellectual property, or IP.

If an IP remains in electronic form, it is one of either two things – a circuit description in HDL or Hardware Description Language (soft IP), or it remains a hardware chip, which includes a hardware IP core. Soft IPs are more flexible, but lack in optimisation. On the other hand, hardware IPs are less flexible, but more optimised.

An SoC usually comprises embedded processors, reused IPs, memory elements like SRAM and ROM, bus architecture, programmable blocks, mixed signal blocks and so on. SoCs may also include other complex components for improved functionality.

However, since IPs belong to other companies, SoC developers have to purchase them from legitimate vendors before incorporating components into the chip. Therefore, it becomes essential to regulate lawful use of IPs to only those who legally own the rights to use them. Otherwise, the IP developer may face loss of revenue. This is why silicon companies have incorporated IP protection measures in VLSI design flow. Security measures have also been applied to SoC design methodology.

IP Reuse In SoC

Due to the reuse of components in an SoC, the design mechanics for the product undergoes a major transition from an application-specific IP (ASIP) design to a block-based design (BBD), and then finally to a platform-based plug and play SoC design.

For a BBD, the electronic system is divided into different mechanisms and these categories are mapped into available IP components. Their placements are retained, and the routing, timing etc, are all re-optimised to aid in the overall optimisation of the entire system. Following this, the BBD is transformed to a platform-based (PBD) design. This PBD allows support for Hard IP or Hardware IP.

Overall, reused IPs increase productivity. However, certain design challenges often arise, which the SoC makers have to tackle and counter independently. These challenges include the use of heterogeneous device technologies, issues related to testing, signal integrity, voltage maintenance and time regulation. Moreover, SoCs are application specific and, therefore, the application of the chip defines the IP. To meet the need of a particular application, these IPs are frequently redesigned.

Hardware and software development is required for an IP-based SoC. IPs which closely resemble the specifications needed to work with a particular app are selected and redesigned to support the designated application. These redesigned IPs transform into Virtual Components, or VC.

IP Infringements

An SoC company has to overhaul the design of an IP extensively to change it into VCs. However, due to constraints in time and effort, the company may hire outside designers and external tools. Often, SoC developers are unable to afford their own fabrication facilities, which compels them to seek outside help. However, the inclusion of third-parties is a cause for concern, especially in the context of maintaining the right of access for the IP.

The SoC developer may have purchased the IP from a legal vendor; however, if the design aspects of the IP are leaked by the external members working on the SoC, it results in massive revenue loss for both the developer and the IP vendor.

The following points show how an IP could be violated in these circumstances:

1. The first and foremost concern for IP creators is that the SoC company itself may acquire the IP through illegal means. The company may

Samsung Exynos 8895 chipset design

decide to hack or steal the IP and in the act, cause loss of revenue for the developers.

2. An untrustworthy design team member in the IP or SoC company may decide to infringe the rights and use the design illegally.
3. Malicious use of designer tools like CAD, while the IP is still being designed or redesigned by the SoC company may also compromise the IP.
4. In case of use of an external fabrication facility, the IP may be handled by external companies and untrustworthy members.
5. An SoC designer may also misuse the IP during the creation of the chipset, or it may be misappropriated by a user of the chip while the IP is still in effect.
6. Vendors who provide IPs for the same SoC company may also infringe its usage.
7. IPs are also vulnerable from remote hardware. Any outside individual can compromise the IP and violate its usage policy.

IPs are generally at risk from three major threats. These threats include unauthorised access to the IP's system and designs, pirated copies of these technologies, and lastly, information retrieval through the use of a Trojan horse.

Preventing IP Infringement

IP developers need to be vigilant and cautious of security loopholes during

not only development, but also retail stages of their products. This move will help ensure that only those who have legally purchased the IP from a vendor can access the design and features. To ensure complete security, IP companies usually adopt these following measures:

Locking-based security: This is a simple and direct way to restrict access to the product. It protects the communication of IPs with remote devices by applying encryption-based technology, along with obfuscation and remote activation.

What this basically means is that while the IP is being transferred from the IP or SoC designers to the fabrication facility, the data is encrypted. This data is then decrypted during the fabrication process.

Obfuscation is also used for security measures wherein, the high quality original IP is transformed into a low quality one. This alteration conceals the true nature of the IP being transferred and only an authorized user can convert it to its original state.

Authentication security: Locking-based security is not sufficient as it leaves some loopholes. For instance, the SoC company to which the decryption key is given may sell the same to a third-party. However, biometric security and watermarking provides an additional layer of security. These cannot be cracked as easily as lock-based security protocols.

Some IP companies also deploy fingerprinting technology to ensure that only an authorized member can access the product. In this process, if an IP embedded on an SoC needs to connect to a remote drive, the hardware module on the chip authenticates the external device, prior to establishing the connection. This process ensures that only secure devices are accessed.

Trojan detection: While designers work on an IP, or during the creation of an SoC, the technology may get infected with Trojans, which may compromise the security of the device. In such cases, the device becomes vulnerable to Trojan side channel attacks.

Therefore, it is imperative to detect if any Trojans are present on the IP before it is transferred to the SoC developers. These detection systems would need to be super-efficient to perceive even the smallest traces of Trojan activity.

The technique deployed by the company uses measurements of actual delays in the combinational paths. A Trojan is detected if certain paths show increased delay, when compared to others.

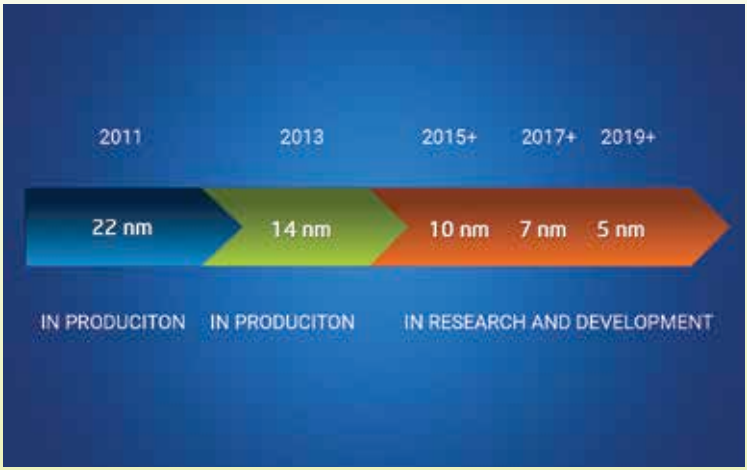
Security And Privacy Safeguards

Safeguarding IPs

With the increased level of innovative architecture in chipsets and the growing popularity of SoCs by companies, the IPs need to be protected better. Companies need to develop newer and more stringent security methods to achieve this goal.

Any major security threat poses risks to the economy of Silicon Valley companies and the chipset sector as a whole. Loss of revenue could be a potentially major setback for IP developers and may negatively impact future technologies being developed by them.

However, IP-based SoCs are on the rise and the trend does not show any signs of slowing down, at least not in the near future. The onset of IP-based SoCs gives manufacturers an easy solution for creating faster and more efficient architectures, at low cost.



NEXT GEN SOCS

Powerful, power efficient and really small. The future of SoCs holds tremendous potential.

With the advent of technology, the term System-on-Chips or SoCs has become a part of our vocabulary. While talking about smartphones and tablets one frequently uses this term. However, most people are unaware of what an SoC is.

To put it succinctly, an SoC – as the name suggests – is a single chipset which houses the main processor, graphics processor, and the memory of a particular device.

So, an SoC comes with a CPU, generally from a company called ARM, which specializes in producing low power systems and processors. Modern-

The Samsung Exynos 5 Octa SoC

day SoCs also house powerful graphical capabilities so that users can easily access and play advanced games directly on their smartphones, sans any hiccups.

SoCs also boasts multiple co-processors which are tasked with different functionalities, each of which serves a greater purpose. For instance, a processor assigned to video encoding and decoding is needed to run the camcorder functionality of smartphones. An image processor ensures that images are saved quickly and processed properly. Finally, a sound processor works to free the load on the CPU as it does the work of processing the audio quality produced by the device.

Differences Between Traditional CPU And SoC

CPU stands for Central Processing Unit. It is generally found in all computer devices and also in smartphones, tablets etc. It is the part of an electronic device which is needed to perform the various calculations and tasks.

However, to perform these tasks efficiently, the CPU requires help from other components, which include Random Access Memory (RAM), the graphics processor, and more.

A CPU setup is costly as individual equipment needs power for proper functioning. With increased power usage, the whole setup has a tendency to overheat. To maintain the temperature of the whole unit users are compelled to invest heftily in a proper cooling solution.

However, CPUs are highly customisable as users can easily swap-out and change components whenever they feel like doing so.

The SoC, on the other hand, is a single chip which pretty much contains every component. For example, the chip holds a CPU, a GPU, RAM, memory, co-processors, and also connectivity components. This results in the need for single equipment rather than multiple components as is the case in a traditional CPU setup.

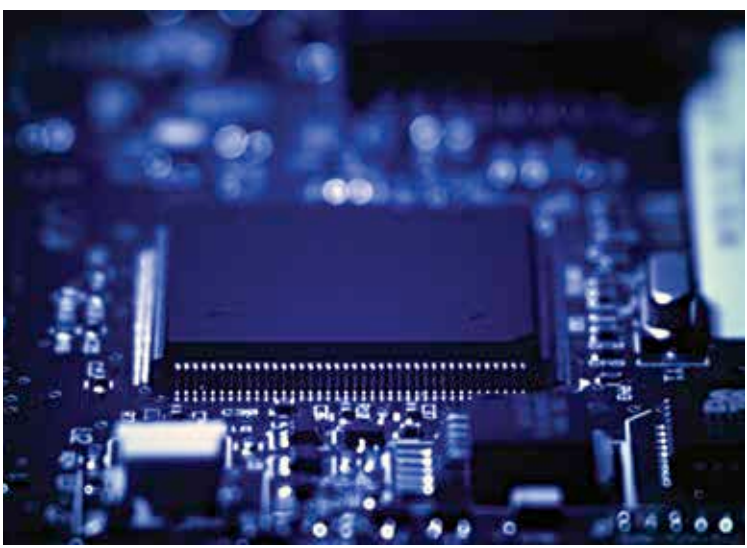
The cost of producing and using an SoC is especially low due to the inclusion of all the components on a single chip.

However, the SoCs cannot be customized by users. If any component of these SoCs gets damaged, it cannot be repaired or swapped out. In such a scenario, the user would have to replace the whole device with a new one.

Aspects Which Determine SoC Quality

In the modern world, people are no longer negligent about the key points which determine what kind of performance a particular SoC will be able to deliver. For such reasons, SoC developers like Intel, Samsung, Qualcomm, and Apple are always trying to outpace each other to ensure that their own chipsets sell more.

The primary aspect of judging an SoC is the CPU and the speed that the chipset clocks or runs at. The next-most important aspect is the number of cores on offer for processing.



A circuit board on display

The current generation of processors have up to eight cores and most smartphone users are gradually shifting to octa-core SoC-powered devices.

With the advent of graphically intensive games, SoC developers have also been forced to include a high quality GPU in the chips, which would be able to run these games without any major lags.

The other important feature which determines the performance of an SoC is its architecture and process count. For instance, most SoCs in the latest devices come with a 14nm process. However, there are some products which have been developed with a better 10nm process. These are references to the transistors which are housed in the chips.

What Is Transistor Count In SoC?

Transistor count refers to the number of transistors present in the integrated circuit of the SoC. The higher the transistor count, the more complex the circuitry is. According to Moore's Law, the transistor count for integrated circuits doubles after every two years.

The law was developed by Intel's co-founder Gordon Moore in 1965, when he identified a pattern in the development of ICs. He noticed that transistor count doubled every two years. So, the complexity of these circuits too increased twofold every two years.

In 2015, Intel became the first company to create a chip which housed over 5.5 billion transistors. It is as of yet, the highest transistor count in a single chip. However, with the exponential increase in transistor counts, new and updated technology has to be integrated into the chips. This has resulted in the rise of FinFET technology.

Why Size Reduction Of Process Is Required?

To include even greater transistor counts for faster and more efficient chips, companies have been pushing the current technology. In a bid to include more transistors, developers needed to create extra space in processors.

This is why process size has to be reduced so as to create more space for transistors, which would further improve the overall performance.

What Is FinFET?

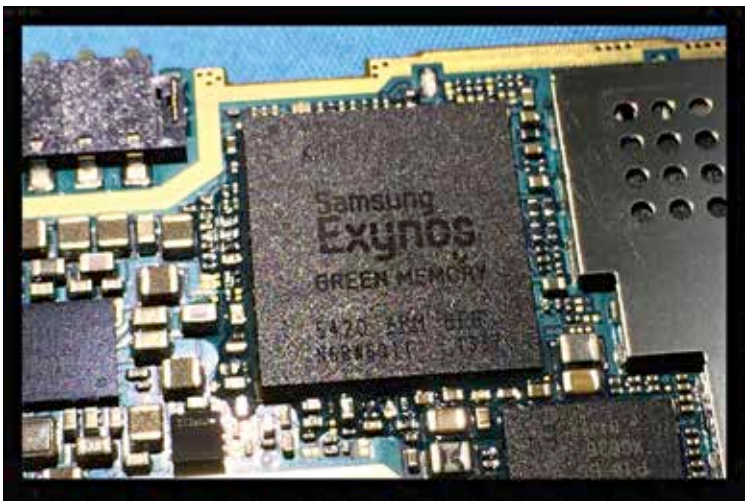
FinFET refers to Fin Field Effect Transistor which replaces the 2D transistors with 3D ones. This technology was adapted in recent years after developers faced problems in integrating the growing number of transistors to the chips. To make the chips smaller, developers scaled its

processes down to 20nm. However, after this was achieved, the integration started to crumble.

Due to this scaling down, when the voltage needed for one process was delivered to the chip, the other processes stopped working properly. To solve this issue, FinFET technology was developed.

Way Forward For SoCs

Intel, Qualcomm, and Samsung have all announced their own 10nm FinFET chips, which would reduce the process size even further from the current 14nm System-on-Chips.



An Exynos Chipset Showing the transistors and arrangement of processes.

According to Samsung, which has developed the Exynos 8895 SoC (a 10nm chip), the decrease in process size has allowed it to house 30 percent more transistors in the same area than previous gen 14nm chips.

With the manufacture of 10nm chips, developers will be able to improve performance by 27 percent, or decrease power consumption by 40 percent in devices.

However, one has to wonder how much smaller can process size get before affecting the overall functionality of the SoC. There have been reports that already with 10nm, developers are facing several issues regarding the architecture.

Use Of New Materials In SoC Manufacture

Apart from updating the architecture of chips, developers have been trying to find new alloys and elements which would further improve performance, while at the same time reducing cost of production. One such alloy, which is being used currently, is a mixture of silicon-germanium (SiGe).

Since 2000 or so, SiGe has been widely used to enhance the performance of ICs of various types.

SiGe can be processed on equipment in nearly the same way as ordinary silicon. SiGe doesn't have some of the drawbacks of III-V compound semiconductors like gallium arsenide (GaAs). For example, it doesn't lack a native oxide (important for forming MOS structures) and doesn't suffer from mechanical fragility, which limits the wafer size of GaAs.

This results in costs that are only a small multiple of ordinary silicon, and so much lower than competing technologies like GaAs.

SiGe allows for two main improvements compared to ordinary silicon:

First, adding Germanium increases the lattice constant of the alloy. If a layer of Si is grown on top of SiGe, there will be mechanical strain induced by the lattice's constant mismatch. The strained layer will have a higher carrier mobility than an unstrained Si. This can be used, for example, to balance the performance of PMOS and NMOS transistors, reducing the area needed for a given CMOS circuit.

Secondly, the SiGe alloy can be used selectively in the base region of a BJT to form a heterojunction bipolar transistor (HBT). SiGe HBT's have been demonstrated with speeds (fT) up to 500 GHz, and are commercially available with fT up to 240 GHz. The SiGe HBT also makes less noise than a standard silicon BJT.

The future of chips could also lead developers to completely abandon silicon in favour of Germanium. Researchers claim that silicon has become difficult to work with in this age of miniaturization. In a bid to create even smaller processes, Germanium in itself may present the answer. The element has superb electrical properties and innovative techniques have made it possible for companies to work with Germanium.

Upcoming State Of The Art SoC

In 2017, the SoC market has been abuzz with reports of the Qualcomm Snapdragon 835 chipset and the Exynos 8895 SoC, both of which are housed in Samsung's Galaxy S8 device. Both are based on the 10nm FinFET architecture.

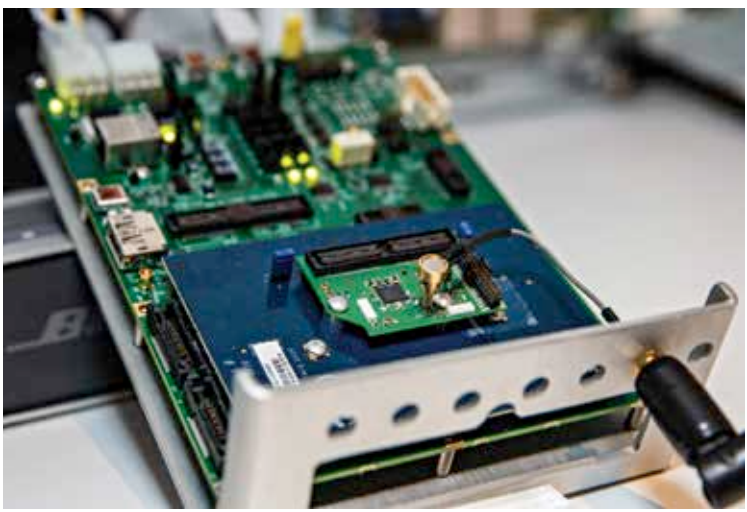
Some of the features of the Snapdragon 835, as listed in the company's website, indicate that it will offer significant improvements in its battery capacity. It will allow users to stream 4K videos. The state of-the-art chip will also come with VR support.

The processor will also come with enhanced security capabilities such as iris scanner, facial, and even voice recognition, along with the standard fingerprint sensor, which is common in most chipsets of this generation.

5G Chipsets In 2017

Companies including Samsung and MediaTek are expected to announce 5G connectivity capable SoCs by the second half of 2017.

Intel and Qualcomm announced 5G chips at CES 2017, which can transmit the GHz band that is currently being tested, as well as lower 3.3 GHz to 4.2 GHz ones.



Intel's Lightning Peak Chipset On Display.

Though Samsung is yet to make an announcement regarding its 5G chipsets, the company is expected to release comparable chips during the same timeframe. MediaTek, Qualcomm, and Spreadtrum Communications are also expected to have their own 5G mobile chipsets ready for use in next year's smartphone device lineups, though the companies may introduce samples by the end of this year.

Future Of Mobile Computing And Smartphones

The innovations in the field of computer technology and smartphones are growing at an exponential rate. This advancement is only expected to continue in the foreseeable future. Interesting research is underway and companies are exploring new technologies to come up with better products.

For instance, smartphone developers are trying to incorporate Augmented Reality (AR) into their devices.

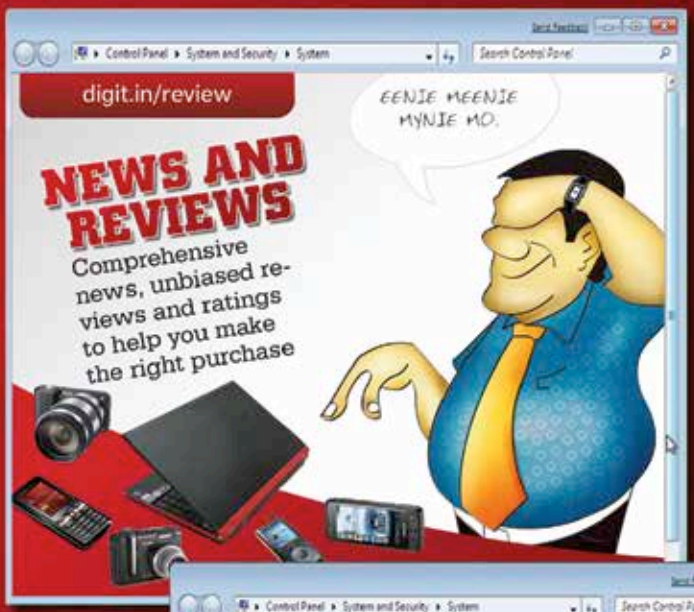
Researchers are also trying to use Organic Light Emitting Diode (OLED) technology to create foldable and stretchable displays, which would be able to change shape as per the user's preference. 3D displays and projection capabilities are also being pursued by researchers currently.

With mobile phone processors getting more and more powerful each day, developers have also started integrating artificial intelligence (AI). However, as time passes, companies may start working on including real AI into their devices, which would not only help users but will also adapt itself according to the needs of the user.

With steady innovations, soon there might not be any use for computers and laptops. Instead, they may be replaced with hand-held devices, which would be able to perform tasks and store data similar to present-day computers.

System-on-Chips have perhaps the most vital role in carrying the burden of these technological advancements. Each and every feature that developers want to add would require better processing power.

Nonetheless, this constant innovation is intriguing for the technology industry. It may interest people to see where technology progresses in the coming future.



All this and more in the
world of Technology

VISIT NOW digit.in

RNI REGN. NO. DELENG/2001/03791