Final Report - InterIIT JLR

Team 83

December 2023

Contents

1 Introduction

1.1 Current Industry Trends

- **The Rise of Software-Defined Vehicles (SDVs)** - Many OEMs are increasingly coming up with vehicles that manage their operations, add functionality, and enable new features primarily or entirely through software. Not only this, this trend is forecasted to grow from a 2022 Market Size of $35.8 Billion to a 2032 Market Size of $249.8 Billion, a percentage growth of about 600%. To date, over 30 top OEMs have taken the initiative to develop their own versions of these SDVs. Another report by the Boston Consulting Group estimates that the auto industry will gain more than $650 billion in value by 2030 thanks to the advent of software-defined vehicles, constituting a significant portion, accounting for approximately 15% to 20% of the overall automotive value.

- **The User-Centric Revolution** - Apart from the evolving tech used in meagre "operational" features, the automotive industry has shown great interest in packaging more and more software and hardware features that improve the driving experience for the users. AI-powered advancements, smooth incorporation of smartphones, advanced display technology, and a heightened emphasis on cybersecurity are shaping the future of in-car entertainment systems. These developments are revolutionising the driving experience, enhancing its enjoyment, efficiency, and safety for consumers around the globe. The in-vehicle infotainment system market is expected to go from $20.50 Billion in 2021 to $48.62 Billion in 2030. Modern automobile consumers are embracing a new perspective as regular software updates that enhance services become the standard. They are not merely purchasing the latest car model expected to endure for approximately five years; instead, they are acquiring a connected, intelligent mobility device on wheels. This smart vehicle enables them to work, socialize, and enjoy entertainment while continually evolving and improving over time.

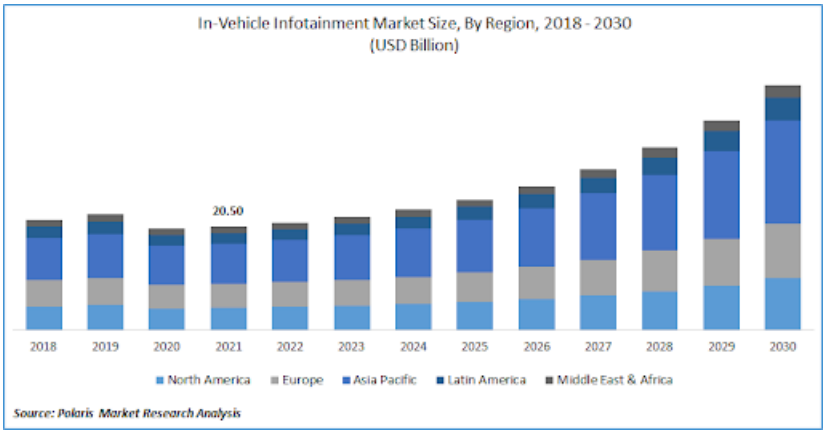


Figure 1: Market size by region

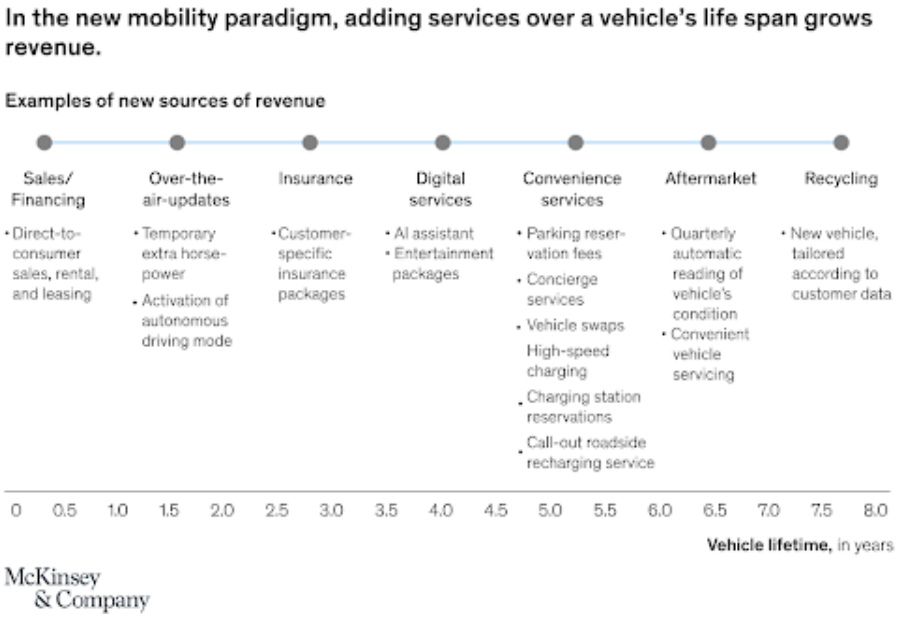


Figure 2: Sources of revenue

- **The Era of Hyper-Connected Intelligent Vehicles** - Vehicles have become more connected than ever before. With the rise of 5G and Internet of Things (IoT) technologies, Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), Vehicle-to-Cloud (V2C), and Vehicle-to-Everything (V2X) connectivity has become possible. This has given these vehicles the added ability to communicate with other software systems and collect data from their surroundings. The global connected car market is worth roughly $88.42 billion in 2023 and is expected to grow to $191.83 billion by 2028.

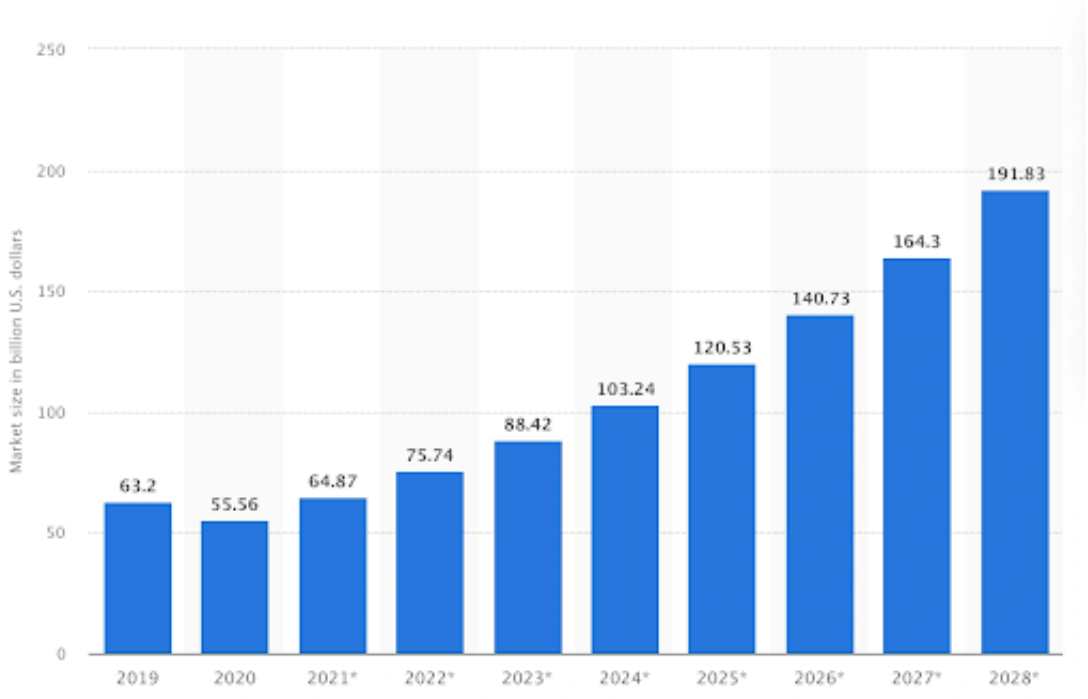


Figure 3: US market growth

- **The Autonomous Vehicle Revolution** - OEMs are rolling out advanced vehicles with autonomous capabilities that are transforming the mobility scenario. With the aid of sensors, actuators, complex algorithms, machine learning systems, and powerful processors, autonomous vehicles are causing a revolution in the transportation sector. Though today, most cars only include basic ADAS features,

major advancements in autonomous driving capabilities are on the horizon. Consumers want access to autonomous driving features and are willing to pay for them, according to a 2021 McKinsey consumer survey. Based on consumer interest in autonomous driving features and commercial solutions available on the market today, ADAS and autonomous driving could generate between $300 billion and $400 billion in the passenger car market by 2035, according to McKinsey analysis. Autonomous vehicles promise to prevent about 90% of traffic deaths, according to a report by the Boston Consulting Group.

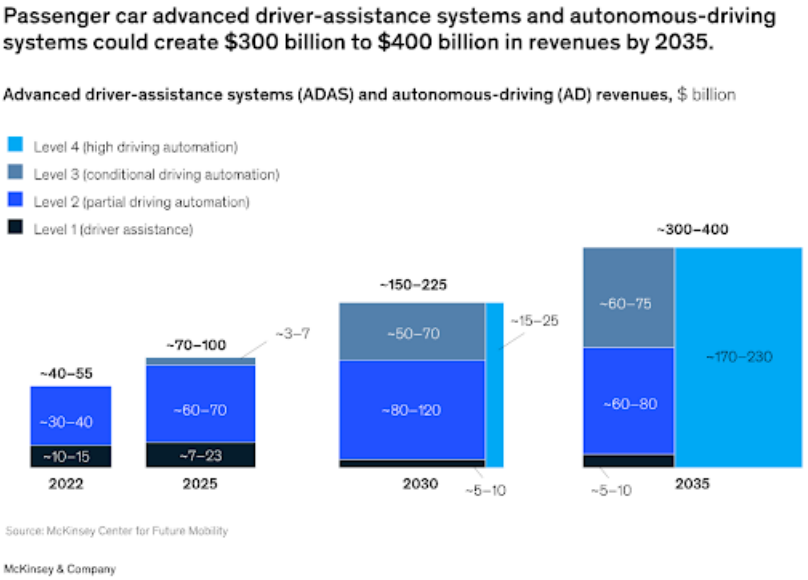


Figure 4: ADS revenue

- **The Acceleration in Adoption of EVs** - With the whole world moving towards a more environment-friendly world, the increasing awareness has influenced the people to buy more electric vehicles and hybrid electric vehicles. The transition to electric vehicles will likely have gained even more momentum, with a broader range of affordable EV models, increased charging infrastructure, and continued government incentives to promote electric mobility. Electric car markets are growing exponentially as sales exceed 10 million in 2022. The share of electric cars in total sales has more than tripled in three years, from around 4% in 2020 to 14% in 2022.

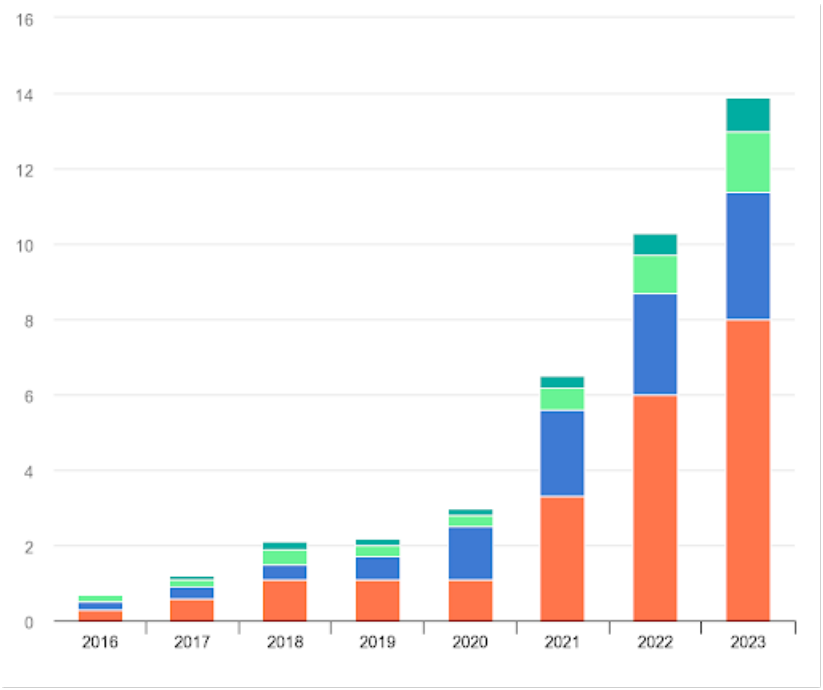
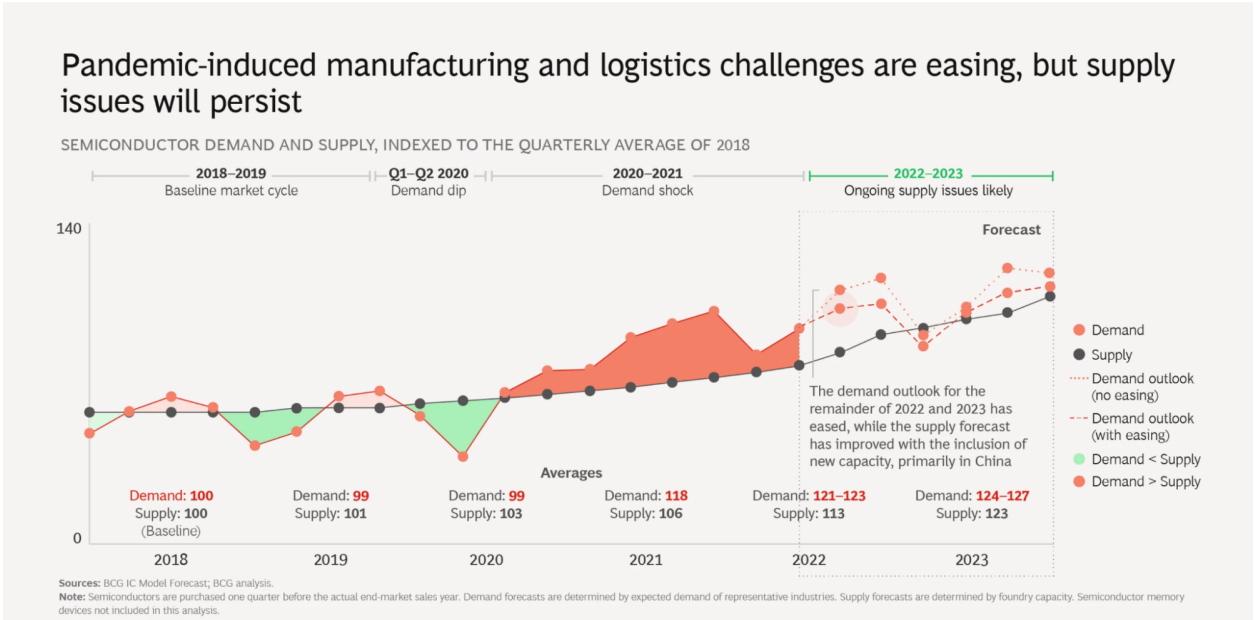


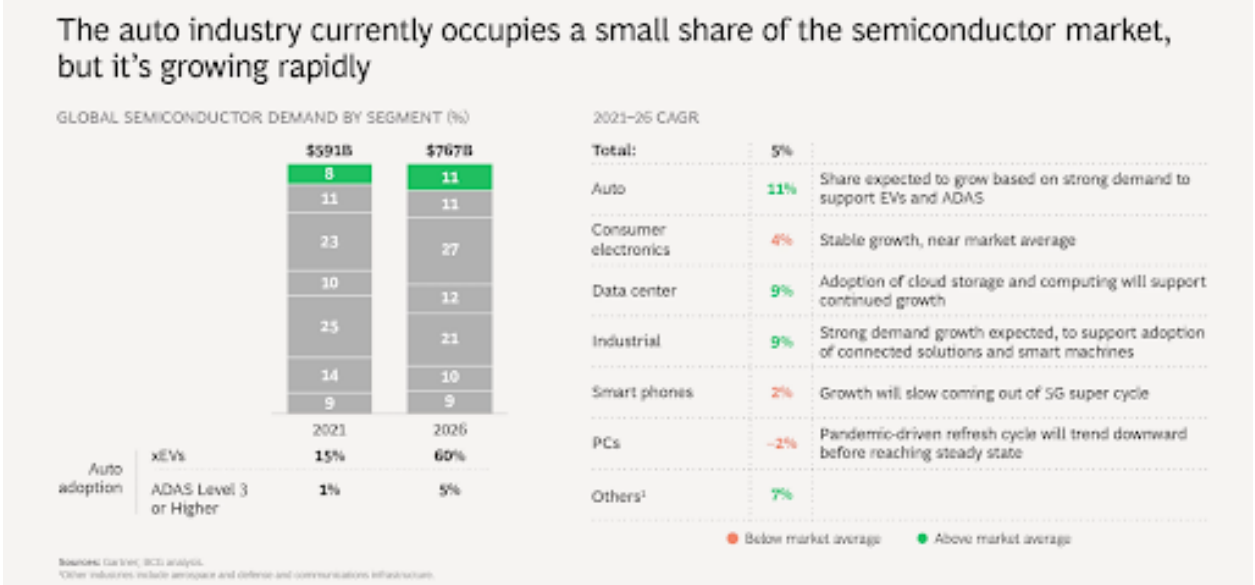
Figure 5: Acceleration in adoption of EVs

1.2 The Problem

- **Global Semiconductor Shortage**-The global semiconductor chip shortage results from the strong demand and lack of supply of semiconductors catalyzed by the COVID-19 pandemic. The movement of the automotive industry towards electric, connected, and autonomous intelligent vehicles is causing a further strain on an already stretched semiconductor market.

  Though the semiconductor production rates have improved significantly from the 2020-21 period, it is expected that these shortages will continue to plague the automotive industry in some form as late as 2026. However, this assumption is being made considering that the automotive industry chip requirement would increase at the current rate. Unfortunately, this might not be the case, as with the increased penetration of semiconductor-intense automotive applications, such as higher levels of autonomous driving systems and electrification, the gap between demand and supply will only widen.





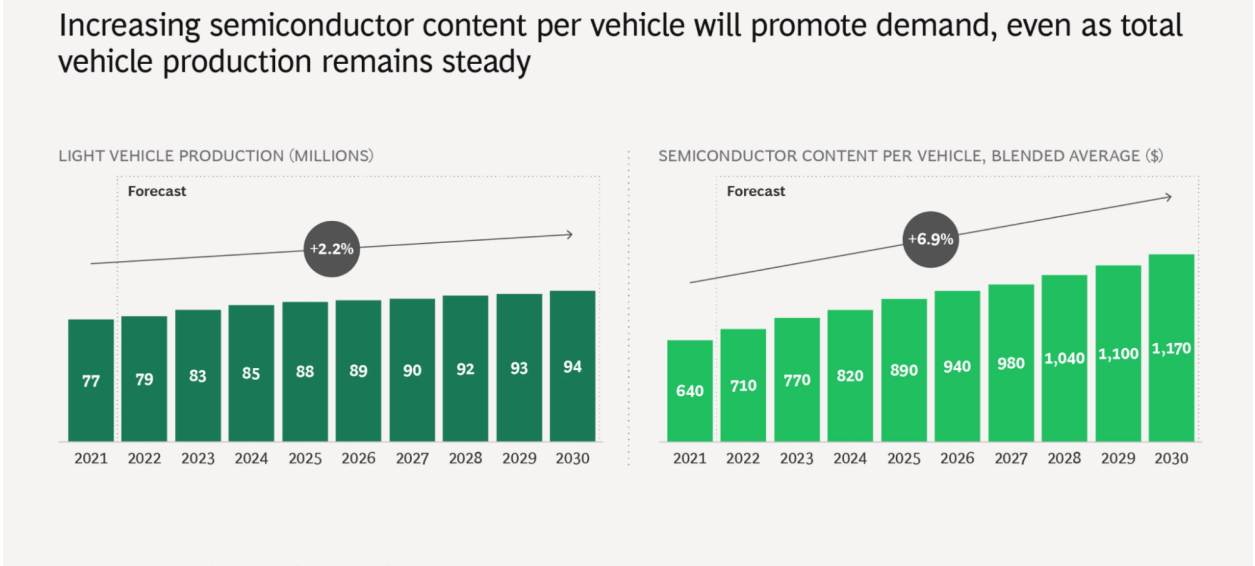


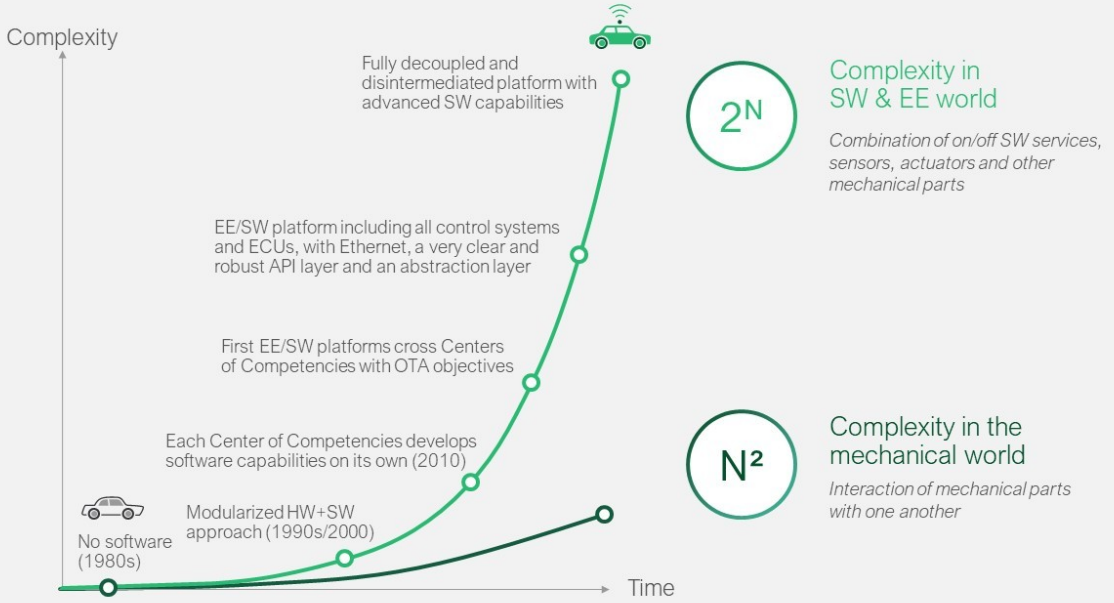
Figure 6: Semiconductor shortage

- **Increasingly Complex Systems**-In recent history, automobiles have undergone a remarkable transformation, with a notable trend being the incorporation of increasingly complex systems. Today's vehicles are equipped with a multitude of advanced technologies that enhance safety, comfort, and performance. These systems encompass everything from intricate computerized infotainment systems to intelligent connectivity features. Complex sensor arrays and artificial intelligence algorithms enable modern cars to autonomously navigate, making self-driving vehicles a reality. Advanced driver-assistance systems, such as adaptive cruise control and lane-keeping assistance, have become commonplace, significantly improving safety on the road. Moreover, electric and hybrid powertrains, which are highly intricate, have gained popularity, reducing emissions and fuel consumption.

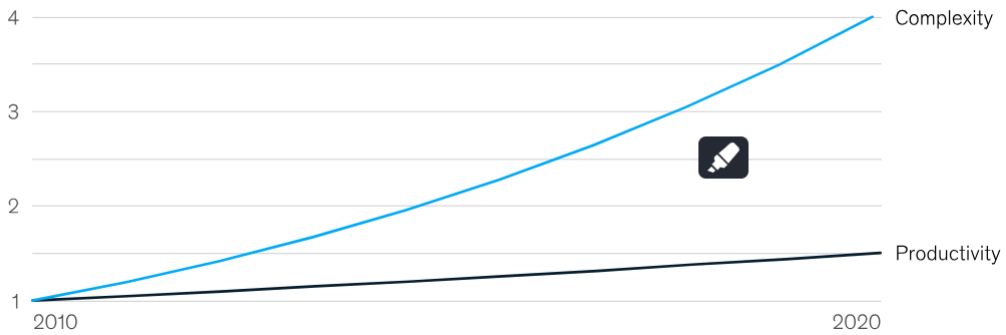
  However, to make all this possible, highly sophisticated systems are required. Modern vehicles implement over 150 electronic control units (ECUs) and nearly 100 million lines of code in order to achieve this. This not only increases the overall cost of the vehicle but also puts a large strain on the hardware and communication systems.

  The average complexity of individual software projects in the automotive industry has grown by 300 per cent over the past decade.

EXHIBIT 1 | Facing exponential complexity in modern car engineering

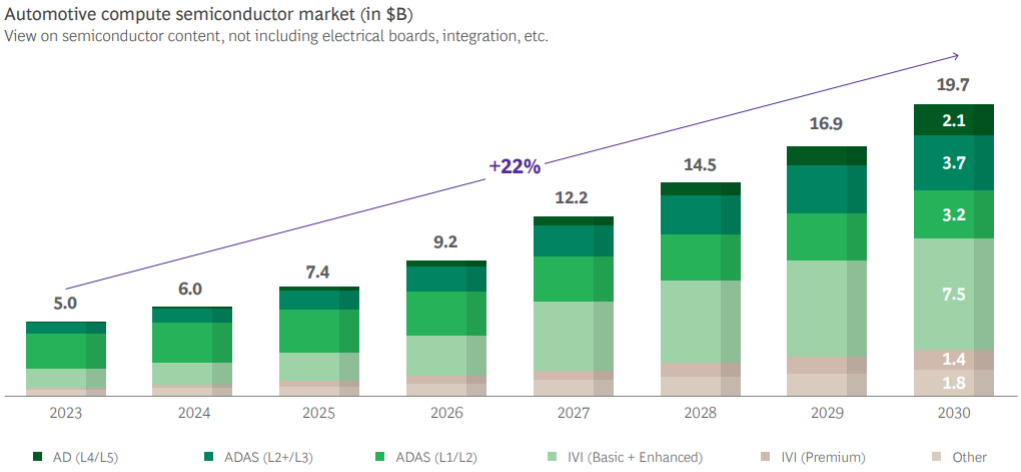


Growth of software complexity and productivity in automotive systems, relative and indexed[1]



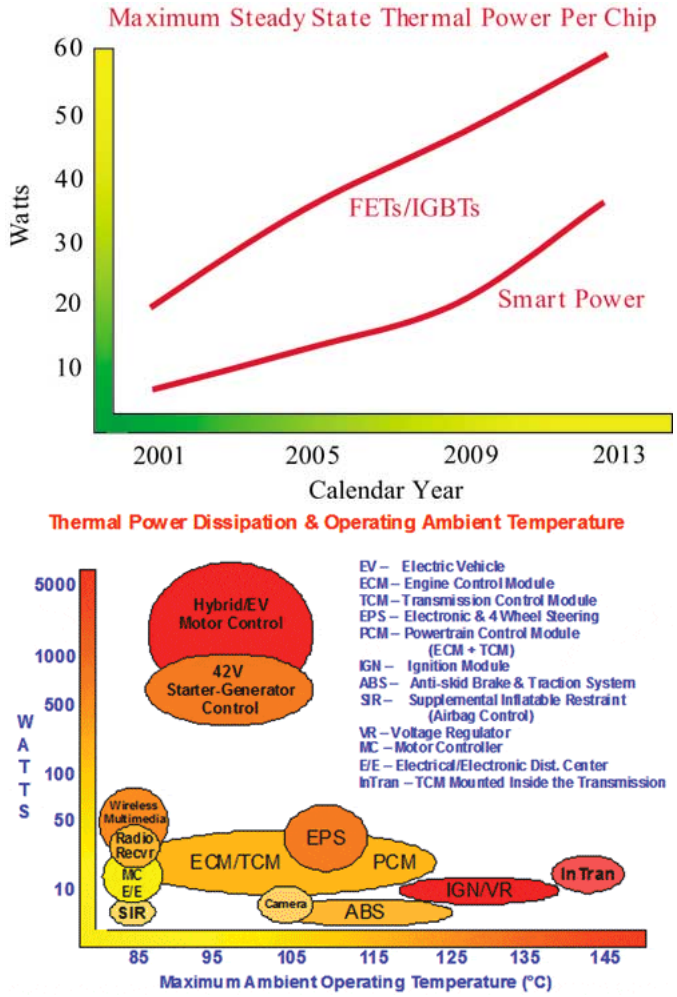
- **Enormous Computational Workloads**-Standard compute power inside the electronic control unit (ECU) will not be able to process the enormous workloads that come with the ADAS, communication, and entertainment functions of tomorrow's vehicles. These systems must process a vast amount of data from various sensors, cameras, and radars, all in real-time, to enable advanced driver-assistance

features and autonomous driving capabilities. Moreover, the integration of complex infotainment and connectivity systems demands significant computing power. As automotive technology continues to evolve, the demand for even more robust computational capabilities in vehicles is undeniable. Anticipated developments suggest that the proportion of light vehicle sales incorporating Advanced Driver Assistance Systems (ADAS) functionalities at Level 2 or above is poised to double from the 2022 levels and potentially reach approximately 50% by the year 2030. Furthermore, the ongoing progress in automotive technology is predominantly directed towards enhancing the in-car experience, introducing more virtual and digital applications, including augmented reality. The driving force behind this trend is undeniably software, which will remain a central element in the automotive industry. It not only fuels innovation but also serves as a key differentiator for Original Equipment Manufacturers (OEMs), significantly influencing consumers' purchasing choices. Facilitating the implementation of these intricate software functionalities demands increasingly powerful chips, consequently driving substantial growth. It is projected that the automotive compute semiconductor market will experience robust expansion, with an anticipated Compound Annual Growth Rate (CAGR) of 22% from 2023 to 2030.



Source: IHS Light Vehicle Forecast, BCG analysis
Note: Includes SoC, CPU, GPU, Chiplet-integrable memory, MCUs, etc.

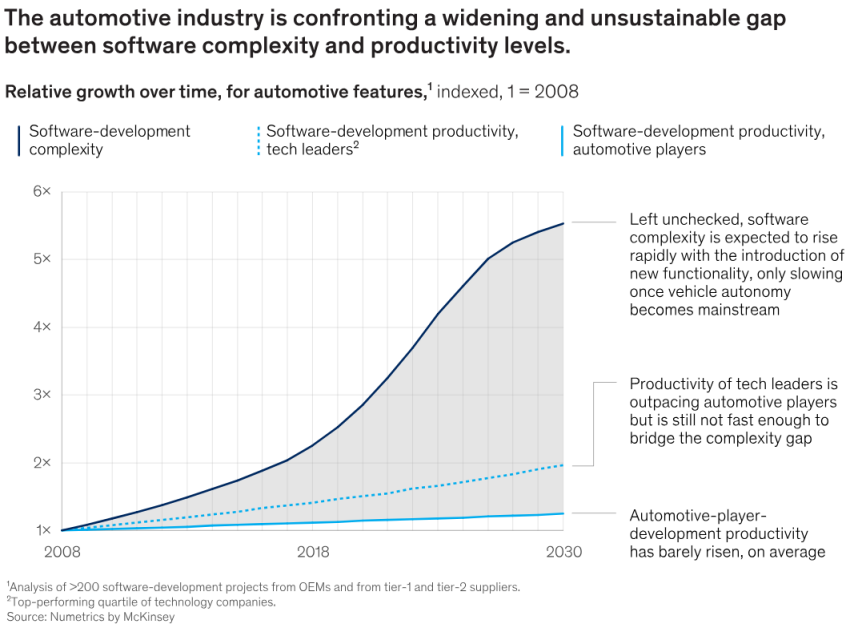
- **Heat Management Challenges**-In semiconductor devices, the component gates have been shrunk down to nanometer sizes, and a single die can now contain millions of gates formed from billions of transistors. Moore's Law has marked this progressive miniaturization and predicts it will continue into the near future until a shift change in the underlying technology. While each new generation of smaller, faster devices offers the designer more features to play with, they generate more heat in the same component footprint. Furthermore, the increased computational performance requirements of the hardware results in much more heat generation, eventually leading to several problems. The properties of the semiconducting material itself change with temperature due to electromigration effects. Outside of the device's temperature limits, the device's performance may not adhere to its specification and produce unexpected behaviors. The challenge is to extract the heat energy from the semiconducting material and dump it into the ambient environment as quickly and efficiently as possible to maintain the device's reliability.



- **Issues with Scalability**-Scalability is a fundamental concern when it comes to the design and development of automobiles. With all the fast-paced updates and technological upgrades coming up in the industry, it is becoming increasingly important to ensure that the hardware technology is scalable to accommodate future advancements and that vehicles can be upgraded without significant hardware replacements.

  A modern vehicle is a cyber-physical system that combines hardware and software components with varying innovation life-cycles, resulting in the challenge of controlling complexity and innovations throughout an electrics/electronics platform's life-cycle. The reliable implementation of over-the-air updates is increasingly impacting customer satisfaction. Thus, electrics/electronics platforms face the challenge of technically and structurally enabling software updates over-the-air, while also considering variability.

  As the volume of software integrated into various aspects of the vehicle continues to grow, so does the complexity of ensuring that different systems function cohesively. Automotive industry participants face significant difficulties in keeping pace with this trend. For example, contemporary infotainment systems now demand a development period of more than three years, with the contributions of numerous software engineers in each iteration. Implementing changes to any individual software module often necessitates extensive reworking. These systems do not consistently support backward compatibility, requiring extensive redevelopment every few years to remain current with new features and performance standards.



- **Energy Efficiency and Power Consumption**-Day by day, manufacturers strive to increase the range and performance of their vehicles. This leads to larger batteries and motors, which are controlled by increasingly complex control circuits. Additionally, energy-draining systems such as larger infotainment systems, and power hungry autonomous driving units further increase the already large amount of energy being consumed. On the other hand, as most are not optimized properly, these extremely power-intensive systems also tend to see a decline in efficiency as performance is pushed to its limits.

  Maximizing energy efficiency is essential, especially in electric vehicles, where hardware components like electric motors and power electronics directly affect range and battery life. Additionally, with the increasing processing requirements and advanced algorithms, significant amounts of energy are consumed just to perform these tasks, raising the need for computationally efficient hardware solutions.

  Data centers, which house the physical infrastructure for running applications, are notorious for their substantial carbon emissions, responsible for approximately 0.3% of global greenhouse gas emissions, equivalent to Argentina's annual carbon production. Recognizing the need to consider the environmental impact of autonomous vehicles, MIT researchers constructed a statistical model. Their findings indicate that a billion autonomous vehicles, each operating for an hour daily with a computer using 840 watts, would generate emissions on par with current data centers.

  The study also reveals that in more than 90% of modeled scenarios, autonomous vehicles must limit their power consumption to under 1.2 kilowatts per vehicle to prevent emissions surpassing those of data centers. Achieving this would require more efficient hardware. In a specific scenario where 95% of the global vehicle fleet is autonomous by 2050, computational workloads double every three years, and current decarbonization rates continue, hardware efficiency would need to double even faster, every 1.1 years, to keep emissions in check. The study underscores the need to proactively design energy-efficient autonomous vehicles to mitigate their carbon footprint.

  Another article from 2018 highlighted that modern production cars equipped with cameras and radar generate approximately 6 gigabytes of data every 30 seconds, a figure that escalates even further for self-driving vehicles featuring additional sensors like lidar. This immense volume of data necessitates complex processing to create a comprehensible depiction of the world for autonomous driving, along with instructions on how to navigate through it. Such processing requires substantial computing power, resulting in high electricity demands. Prototypes typically consume around 2,500 watts, which is enough to power 40 incandescent light bulbs.

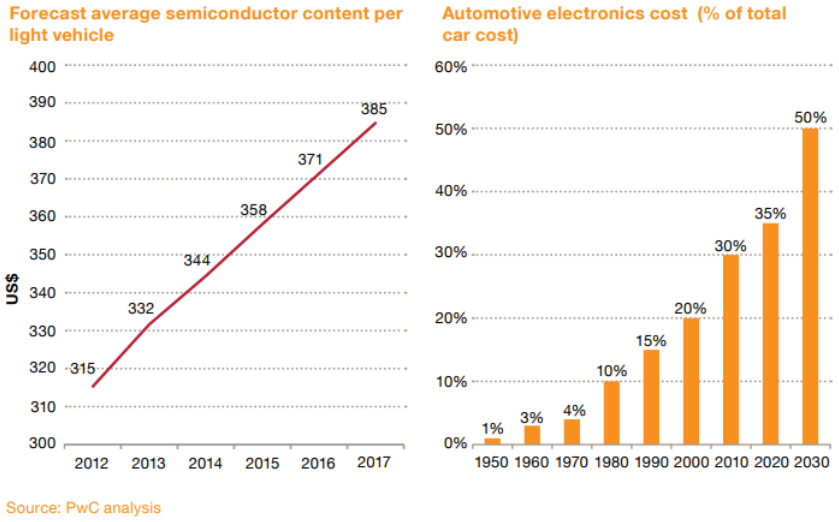
  Embedding such a system into a traditional combustion-engine vehicle is impractical due to the substantial increase in fuel consumption it would entail. This is because the energy required for computing would significantly reduce the vehicle's fuel efficiency. In the context of electric cars, the electricity

drawn by these computing systems results in reduced driving range, as a significant portion of the available battery power is allocated to the computers instead of driving the vehicle.

- **Cost Pressures on Hardware**-Meeting the demands of advanced hardware technology can be expensive, and automakers face the challenge of balancing cost-effectiveness while delivering high-quality hardware in their vehicles. An analysis by PwC reports that the percentage of the cost of electronics out of the total car cost is expected to reach about 50% by the year 2030.

  With the technology becoming more advanced, and with the growing performance requirements, the cost of producing such complex hardware systems is also increasing proportionally. Moreover, the capability of producing such sophisticated electronics is only with a few specialized companies, resulting in a more dangerous situation where these specialists are free to set the price of these systems. The absence of a competitive market can cause the cost of hardware to shoot up uncontrollably.

  Thus, if the automotive OEMs are not able to keep all these issues under check, the public might not buy anymore, resulting in the decline of the whole automotive industry.



1.3 Root Cause Analysis

1.3.1 Global Semiconductor Shortage

The automotive industry is facing a major hurdle – a significant shortage of semiconductors. This scarcity is impacting production, leading to delays and limited availability of new vehicles for consumers. Several interconnected factors contribute to this complex issue, demanding a multifaceted approach to solutions.

One key factor is the lower yield of semiconductors specifically designed for automotive applications. Modern SoCs are intricate and require complex manufacturing processes, which can result in a higher percentage of defective or suboptimal chips. This wastage exacerbates the shortage, particularly for large die sizes needed for advanced features. Additionally, many automotive SoCs are initially designed for consumer electronics, which have shorter lifespans and less stringent reliability requirements. Adapting these chips to the demanding 15-year lifespan and high reliability standards of automotive systems proves challenging, potentially leading to further wastage.

Another contributing factor is the limited manufacturing capacity for these advanced chips. Only a few manufacturers possess the technology to produce the low nm nodes required for the most sophisticated SoCs. This technological gap creates a bottleneck in the supply chain, slowing down production and hindering car manufacturing. Consequently, automotive OEMs struggle to source the necessary chips to meet their production targets.

Further complicating this issue is the decreasing product life cycle of cars. With rapid advancements in hardware and software technology, the average lifespan of a car has significantly decreased, from an average of 15 years to just 5 years. This shorter lifespan leads to faster hardware and software obsolescence. Many existing chips cannot adapt to new software updates or changing hardware requirements, rendering them

unusable before the end of the car's life. This contributes to the growing e-waste problem, as these discarded chips cannot be easily reused due to the complex architecture of modern vehicles.

1.3.2 Increasingly Complex Systems

The automotive industry is facing a significant challenge: the ever-increasing complexity of electrical and electronic (E/E) architectures. As vehicles become more sophisticated and packed with advanced features, the traditional decentralized approach, with numerous independent Electronic Control Units (ECUs) for each function, is reaching its limits. This raises substantial challenges in terms of scalability, communication efficiency, and overall system complexity.

Firstly, the current decentralized architecture, often exceeding 100 ECUs managing 150 million lines of code, creates significant scalability issues. Each function demands its own ECU, leading to overcrowded wiring harnesses and cumbersome software management. This complexity makes it difficult to introduce new features and maintain system reliability.

Secondly, communication efficiency becomes a bottleneck in the decentralized architecture. With numerous ECUs communicating with each other, the network becomes overburdened, impacting real-time performance. This hinders the effectiveness of advanced features like autonomous driving and connected car technologies that rely on seamless data exchange.

Furthermore, the proliferation of sophisticated features necessitates dedicated hardware for each function. This includes high-performance processors for complex sensor fusion and safety systems, powerful AI chips for autonomous driving, advanced networking hardware for V2X connectivity and over-the-air updates, and high-resolution displays requiring powerful GPUs for infotainment and instrument clusters. Integrating all these specialized systems onto a single System-on-Chip (SoC) proves impractical in most cases, requiring multiple interconnected SoCs. This further increases communication complexity and adds to the overall system burden.

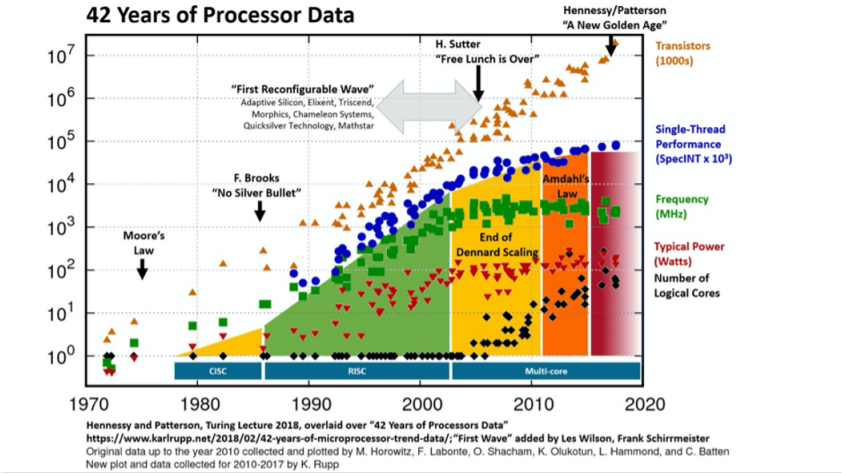
Integration with legacy systems presents another hurdle. As new features are introduced, they need to be compatible with existing vehicle architectures and legacy systems. This often requires additional hardware layers for interconnectivity and adaptation, introducing potential compatibility issues and reliability concerns.

1.3.3 Enormous Computational Workloads

The ever-evolving automotive landscape brings with it a significant challenge: managing enormous computational workloads. As vehicles become increasingly sophisticated, packed with advanced features and functionalities, the traditional decentralized E/E architectures struggle to keep up. This creates a complex scenario with significant implications for both performance and efficiency.

The decentralized approach, with numerous ECUs handling specific functions, leads to redundant processing. Each ECU independently analyzes similar data, resulting in wasted resources and unnecessary computational effort. Additionally, constant communication between these ECUs adds another layer of complexity, further increasing the overall workload and hindering system efficiency.

The proliferation of advanced features, such as autonomous driving, connected car technologies, and enhanced infotainment systems, demands dedicated hardware for optimal performance. AI accelerators, GPUs, NPUs, and other high-end processors contribute to the computational load, adding to the overall system complexity and resource constraints.



The traditional exponential growth in computing power predicted by Moore's Law is slowing down, presenting a new challenge. Single-chip processing power is not increasing as rapidly as before, making it difficult to accommodate the ever-growing computational demands of new features with existing hardware. Additionally, the "dark silicon" phenomenon, where some transistors remain inactive on a chip to manage heat dissipation, further limits the effective computational power available.

SoCs are designed for specific functions and integrate various components optimized for their intended purpose. This tightly integrated design makes them efficient for their specific applications. However, when computational requirements increase due to advanced features, the scalability of existing SoCs becomes a challenge. Scaling them up to meet higher demands might require re-designing the entire SoC, adding to the complexity and cost.

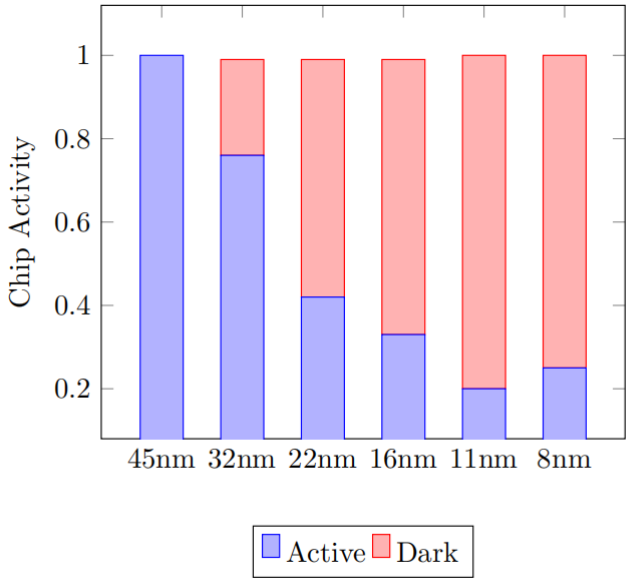


Fig. 1.1: Projections of Dark Silicon [7] [5]

1.3.4 Heat Management Challenges

As automotive E/E architectures become increasingly complex, so does the heat generation within each subsystem. This poses significant challenges for maintaining optimal performance, reliability, and long-term functionality.

Each functional element, from GPUs to AI accelerators, generates heat as it processes data. Under idle conditions, temperatures might range from 30°C to 45°C. However, under the heavy workloads characteristic of autonomous driving and connected car technologies, temperatures can reach 65°C to 85°C. In some cases, even 110°C can be observed for high-performance GPUs. These figures, already significant for consumer

electronics, are likely to be even higher in automotive applications due to the constant processing of large data volumes.

The slowing down of Moore's Law presents another hurdle. Traditionally, computing power was boosted by increasing clock frequencies and single-core performance. However, this approach has reached its limits due to limitations in parallelism and energy penalties.

The shift to multi-core architectures and transistor scaling, while initially effective, also faces challenges. Scaling down transistors reduces power consumption but increases power density, leading to higher heat generation. Reaching the physical limits of scaling technology exacerbates this issue, as voltage cannot be scaled down as effectively as transistor size. This results in higher chip temperatures, demanding more complex heat management solutions.

Automotive electronics often rely on passive cooling methods like heat sinks and chassis conduction due to space and noise constraints. Active cooling systems like fans or liquid cooling are less common due to reliability concerns.

However, the compact packaging and layout of electronic components can trap heat and impede airflow. Additionally, the enclosed spaces within vehicles further limit airflow and heat dissipation. These factors make effective thermal management a significant challenge.

1.3.5 Issues with Scalability

The automotive industry faces a challenge with the increasing inflexibility of its hardware architecture. This inflexibility creates significant barriers to innovation, future-proofing, and the integration of emerging technologies.

Inflexible designs often lack modularity, making it difficult to replace or upgrade individual components without significant effort and cost. This restricts the ability to incorporate new features or technologies that require hardware changes. Fixed and predetermined hardware configurations limit the system's capacity to adapt to changing requirements.

Proprietary or non-standard connectors and communication protocols create hurdles for seamless integration of new hardware components. This reliance on specific interfaces restricts the use of third-party or off-the-shelf components, limiting the flexibility and scalability of the system.

Inflexible hardware often comes with limited computational capabilities that are difficult to upgrade. This poses challenges for running demanding software applications and scaling the system for future advancements.

Inflexible designs are typically not built with future advancements in mind, leading to rapid obsolescence and difficulty meeting evolving consumer expectations. This can result in vehicles becoming outdated and unable to accommodate the latest technologies and features.

Each individual hardware component in an inflexible architecture is designed for a specific task and optimized for its intended use. This specialization limits the component's ability to adapt to different functions or work with other systems, hindering scalability. Rapid software advancements quickly outpace the capabilities of hardware, leading to shorter component lifecycles and reduced functionality.

Replacing existing hardware with new versions can be complicated due to incompatible communication systems and interoperability. Integrating new hardware often requires significant redesign and changes within the coordinating electronics, which can be time-consuming and expensive. The use of SoCs, which integrate various functionalities onto a single chip, further complicates upgrades and replacements, requiring the entire chip to be swapped even if only a single component needs updating.

The trend towards smaller process nodes in SoCs offers improved performance and functionality but sacrifices upgradability. SoCs are highly specific to automotive applications and designed by a limited number of manufacturers, making it difficult to customize or upgrade individual components. This specificity, while beneficial for performance optimization, restricts feature differentiation and limits the flexibility of automotive electronics.

1.3.6 Energy Efficiency and Power Consumption

The growing demand for faster and more powerful electronic devices has brought us face-to-face with a critical challenge: energy efficiency. Despite advancements in technology, several factors contribute to the increasing power consumption of these devices, hindering their overall efficiency.

The "dark silicon" phenomenon arises from the limitations of Moore's Law, which predicts a doubling of transistor density every two years. While transistor density continues to increase, the rate of improvement in computing power has slowed down. This means that even though we have more transistors on a chip, we are not experiencing the same level of performance gains per watt. To manage power consumption and prevent overheating, a significant portion of these transistors remain inactive, essentially becoming "dark silicon." This limits the overall processing power of the chip and reduces its efficiency.

As electronic components become smaller and more densely packed, the task of effectively cooling them becomes increasingly difficult. Traditional cooling methods, such as air or liquid cooling, may not be sufficient to dissipate the heat generated by tightly packed components. This inadequate heat dissipation can lead to thermal throttling, where the device reduces its performance to prevent overheating. This throttling further reduces energy efficiency.

Semiconductors with a smaller band gap have a higher intrinsic carrier concentration, meaning more electrons are available to participate in conduction. While this might seem beneficial, it also leads to increased leakage current, which translates to higher power consumption and decreased efficiency. Additionally, the smaller band gap allows for more thermal generation of electron-hole pairs, further contributing to heat dissipation and energy loss.

The shorter lifespan of modern electronics, particularly those based on system-on-chip (SoC) hardware, plays another significant role in reducing energy efficiency. As product life cycles decrease, so does the time for manufacturers to invest in efficient power management and thermal design. This leads to devices that consume more power and have lower energy efficiency throughout their shorter lifespan.

1.3.7 Cost Pressures on Hardware

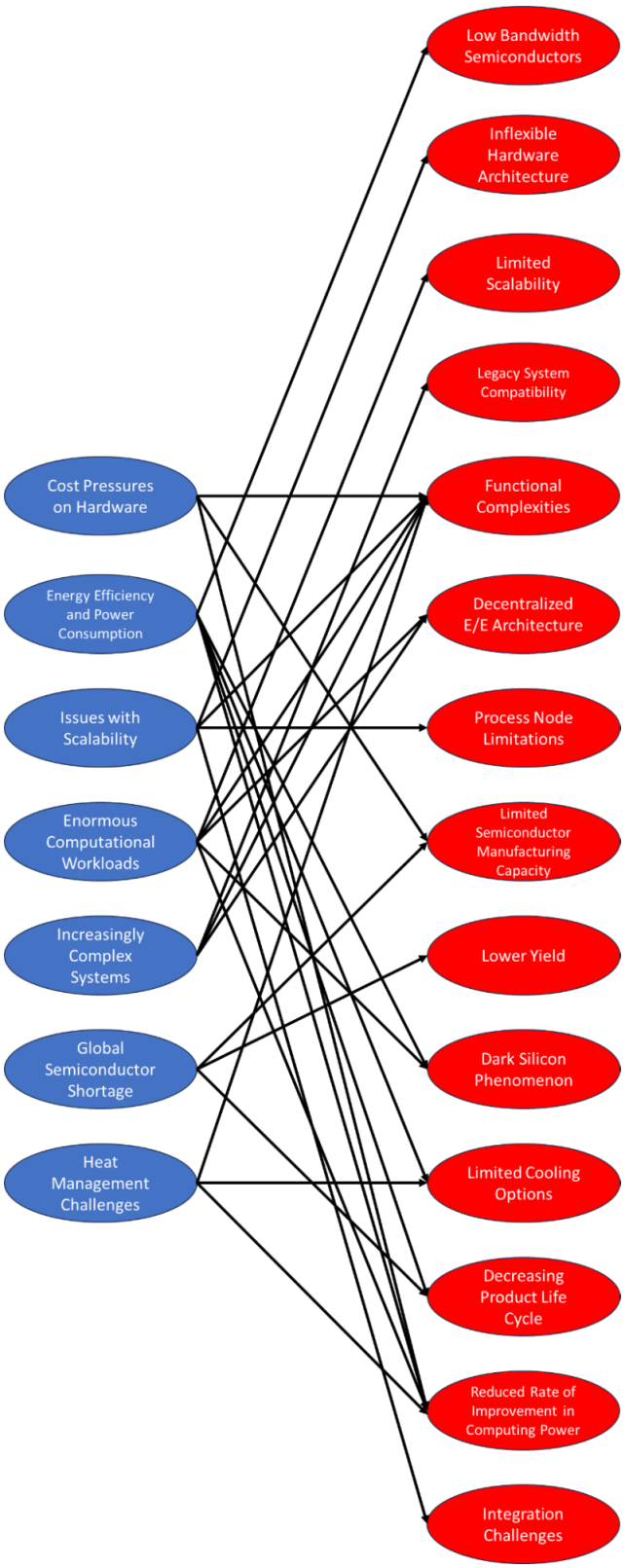
The automotive industry, particularly in the burgeoning electric vehicle (EV) sector, faces significant challenges in controlling hardware costs. These challenges stem from limited manufacturing capacity, functional complexities, and the reduced rate of improvement in computing power.

Only a handful of manufacturers possess the technology necessary to fabricate the specialized hardware required for automotive applications. This lack of competition creates an oligopoly, where these manufacturers have significant control over the pricing of their components. This puts immense cost pressure on the industry.

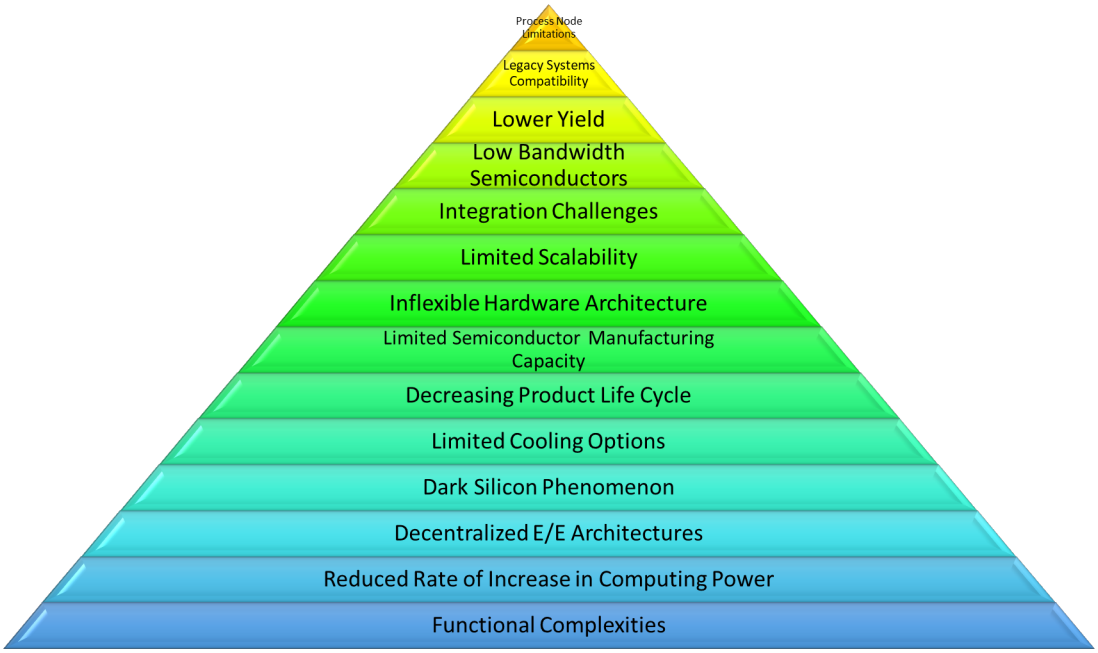
Each subsystem in an automotive system has specific functionality, and often these functionalities are highly specialized. This necessitates the use of dedicated hardware designed to meet these specific requirements. Unfortunately, this specialized hardware cannot be readily purchased from generic vendors, but must be sourced from industry specialists or even manufactured in-house. This significantly increases the cost of the hardware compared to more general-purpose components.

The automotive and EV industries require continuous cost reductions in computing solutions to improve affordability and accessibility. This cost encompasses both the hardware itself and the software associated with development, testing, validation, maintenance, and updates. However, the recent slowdown of Moore's Law, which previously predicted a doubling of transistor density every two years, has rendered the traditional approach of cost reduction through increased transistor count per chip less effective. This necessitates exploring alternative strategies such as optimizing design and architecture, utilizing open-source software platforms, and adopting industry standards.

1.4 Fault Tree Analysis



Based on the fault tree, we can identify the root causes and easily see which causes occur more commonly and prioritize to tackle those specific problems. The pyramid below depicts the root causes and the number of large-scale issues that they cause. Problems near the base of the pyramid are more important to solve as they cause multiple issues, as compared to problems near the peak of the pyramid which cause fewer problems.



1.5 Identifying Alternatives

Now that the underlying root causes have been identified, we explore different alternative semiconductor technologies that can provide suitable solutions to the discussed problems. For our analysis, we considered four major solutions which include the current state-of-the-art technologies used as well as the prospective solutions. They are as follows:

- **System-on-Chip (SoCs)**-A "System on a Chip" (SoC) is a revolutionary electronic component found in modern devices like smartphones and tablets. It stands out for its ability to consolidate numerous essential hardware elements into a single, compact chip. Typically, an SoC includes a Central Processing Unit (CPU) for general computation and a Graphics Processing Unit (GPU) for efficient graphics rendering, enhancing the performance of applications and multimedia tasks. SoCs also encompass memory components (RAM and storage) for swift data access, reducing the reliance on external memory modules. Furthermore, SoCs can house additional components like audio processors, hardware accelerators, and connectivity modules. These integrated elements deliver comprehensive and energy-efficient solutions for the complex functions needed in modern electronics. SoCs represent a significant leap in semiconductor technology, enabling smaller, more energy-efficient devices with diverse capabilities, all within a single chip.

- **Chiplets**-Chiplets are a novel paradigm in semiconductor design, where complex integrated circuits are disassembled into separate, specialized components. These chiplets are then interconnected, allowing for modular and scalable system design. This specialization optimizes performance and power efficiency, and heterogeneous integration enables custom solutions. Chiplet manufacturing can improve yield and time-to-market by reducing waste, and upgrades become easier. Customization offers tailored solutions. However, this approach presents challenges in packaging, power delivery, thermal management, and software development. In summary, chiplets offer flexibility and the potential for enhanced performance and efficiency in diverse electronic systems.

- **Field-Programmable Gate Arrays (FPGAs)**-Field-Programmable Gate Arrays (FPGAs) are user-configurable integrated circuits with hardware-level performance and software-like adaptability. They feature configurable logic blocks and programmable interconnects, allowing custom digital logic creation using hardware description languages. FPGAs excel in rapid prototyping, custom logic deployment, and digital signal processing, serving industries from aerospace to telecommunications. They're prized for specialized hardware acceleration, real-time processing, and extended device lifespans. Valued in education and research, FPGAs aid digital design education and experimentation. However, mastering digital design and FPGA programming languages is essential, making them a top choice for professionals and engineers seeking tailored, high-performance digital solutions.

- **Application-Specific Integrated Circuit (ASIC)**-An ASIC, or Application-Specific Integrated Circuit, is a custom-designed integrated circuit optimized for a specific task. It offers efficiency and high

performance tailored to its intended application, be it in consumer electronics, telecommunications, or industrial settings. ASICs are non-programmable, unlike FPGAs, and their design complexity demands specialized expertise and tools. While development can be costly, ASICs become cost-effective in high-volume production due to their task-specific efficiency. These chips are invaluable when dedicated, high-performance hardware is required to meet precise application requirements, surpassing the capabilities of general-purpose processors or FPGAs.

1.6 Assessing the Alternate Technology

In order to identify the regions in which these alternatives can be put to use, we perform a SWOT analysis to ascertain areas in which these alternatives can provide fruitful solutions while simultaneously steering away from the applications where their use would be disadvantageous. Through the SWOT analysis we can get a qualitative idea of the relative strengths and weaknesses of the solutions along with their possible applications.

Further, we have prepared a quality factor determination matrix in order to quantitatively analyze each alternative. If multiple alternatives could be considered for a single application, the quality factor determination matrix helps us identify which alternative would be the better choice by quantifying the benefits. In order to do this, the alternative solutions are listed against the root causes identified, and they are rated accordingly. A weightage is assigned to each root cause in proportion to the number of large-scale issues it leads up to.

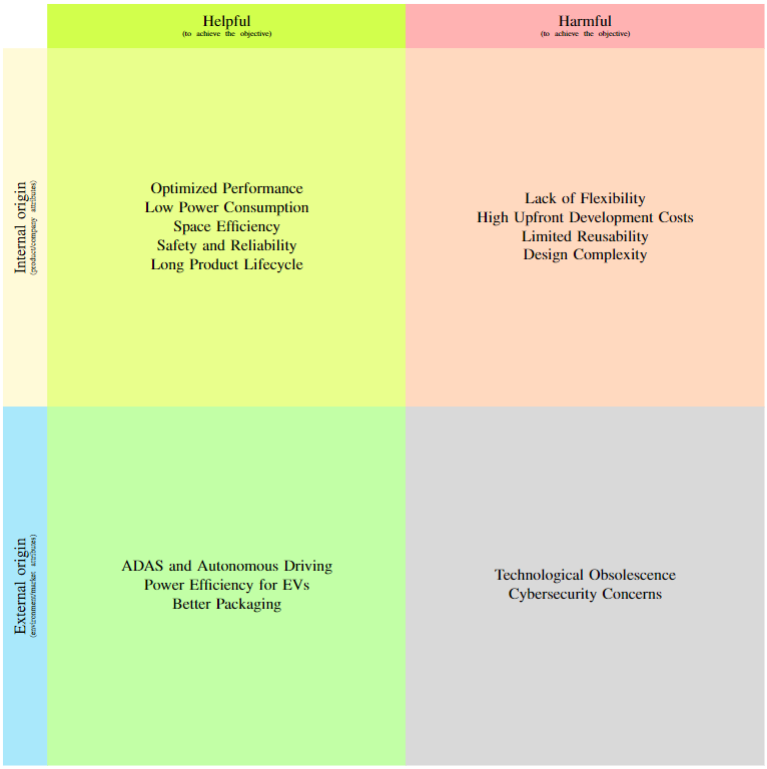
Through both these exercises, we can ultimately come to an informed and elaborate conclusion as to where and why each alternative should be implemented. This would help us identify potential areas of application of chiplets.

- **SoCs SWOT Analysis:**

- **Chiplets SWOT Analysis:**

- **FPGA SWOT Analysis:**

- **ASIC SWOT Analysis:**



Quality Factor Determination (QFD) Matrix:

Technology	Weightage	SoCs	Chiplets	FPGA	ASIC
Computing Power	4	9	7	6	5
Ease of Cooling	3	5	5	4	7
Product Life Cycle	4	5	9	7	6
Hardware Flexibility	2	4	8	9	6
Scalability	3	5	9	6	7
Yield	4	5	8	4	6
Ease of Integration	2	6	8	7	9
Power Efficiency	5	8	8	6	9
Cost	5	8	6	5	9
Total Score		206	240	185	230

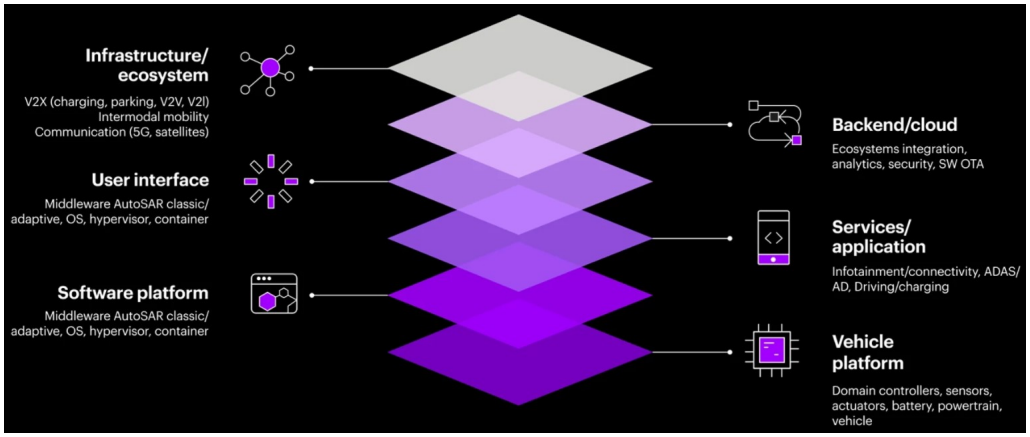
As it is evident from the matrix, chiplets seem to be the best option out of the lot, followed by ASICs, and SoCs, with FPGAs at the bottom of the table. Not only this, through this table we can also get an understanding of how these compare for individual issues, which will help us choose applications accordingly.

The Traceability Matrix below maps the electronics requirements to mitigate the root causes of the current industry trends. This helps us identify the areas in which chiplets can find specific solutions.

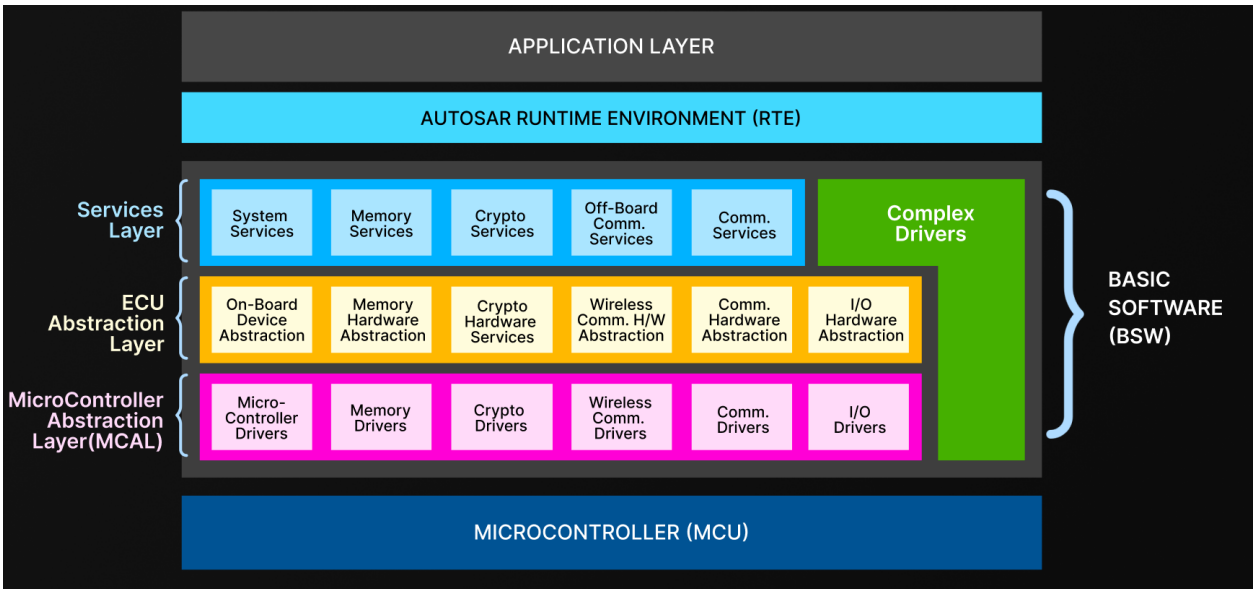
	Software Defined Vehicles	User Feature Differentiation	Intelligent Connected Vehicles	Autonomous Vehicles	Electric Vehicles
Computing Power	SoC	Chiplets	FPGA	SoC	ASIC
Cooling Options	ASIC	ASIC	ASIC	ASIC	ASIC
Product Life Cycle	Chiplets	Chiplets	FPGA	Chiplets	Chiplets
Hardware Flexibility	FPGA	FPGA	FPGA	FPGA	Chiplets
Scalability	Chiplets	Chiplets	Chiplets	Chiplets	ASIC
Yield	Chiplets	Chiplets	Chiplets	Chiplets	Chiplets
Ease of Integration	ASIC	Chiplets	ASIC	ASIC	ASIC
Power Efficiency	Chiplets	Chiplets	SoC	ASIC	ASIC
Cost	ASIC	ASIC	SoC	ASIC	ASIC

2 Applications of Chiplets in the Modern Automotive Industry

The automotive industry is undergoing a transformative shift as trends like connected vehicles, electric mobility, and autonomous driving drive increased digitization. While vehicles traditionally focused on hardware, there's a significant transition towards software-based features. This evolution, turning vehicles into software-centered cyber-physical systems, enables advanced functionalities like over-the-air updates and self-driving capabilities, necessitating seamless interactions between software, hardware, and the vehicle's environment. Scholars propose an electrics/electronics (E/E) platform as a solution, integrating hardware and software principles to establish a comprehensive vehicle architecture. This platform acts as a connection layer, facilitating the close integration of hardware and software within a cyber-physical system, aiming to enhance scalability, cost savings, and the overall effectiveness of vehicle engineering.



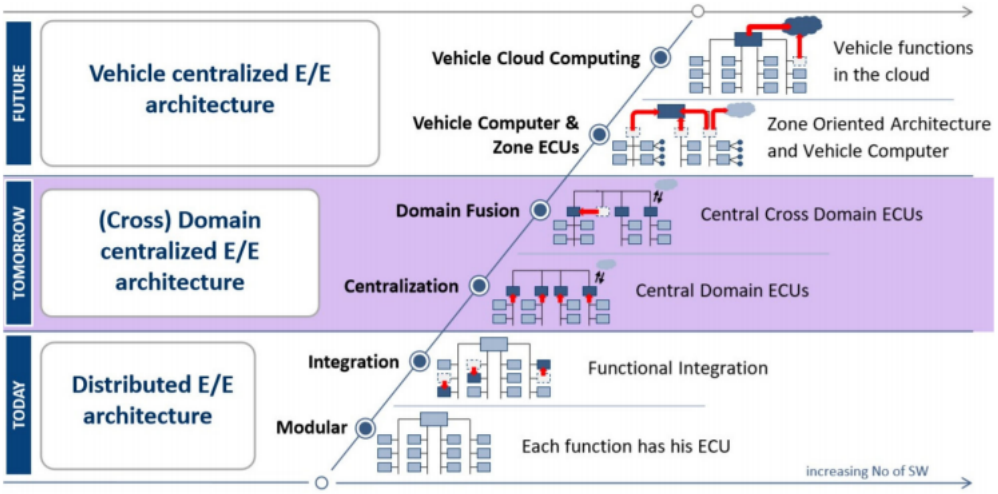
The E/E platform functions as a unified framework for efficient component reuse, introducing a new level of abstraction in structuring products. It integrates all software artifacts and their physical implementations within electric control units, forming an interconnected embedded system. This platform does not alter the existing platform strategy but introduces a cross-vehicle basis that increases the reuse and standardization of software-intensive components.



Moreover, the current landscape of vehicle requirements necessitates over 100 Electronic Control Units (ECUs) with a one-to-one mapping to individual functions, resulting in a decentralized architecture. This decentralized approach significantly elevates the communication load on the vehicle network, leading to increased overall costs. The traditional use of networking technologies like Controller Area Network (CAN), FlexRay, LIN (Local Interconnect Network), and MOST (Media Oriented System Transport) has been adequate for traditional communication. However, with the growing implementation of software functionalities in modern cars, the "one function per ECU" model is proving impractical. The challenges include the expected surge in data volume and the escalating number of interconnections between different ECUs. Building

vehicle variants for diverse market segments requires numerous software releases and may entail constructing multiple architectures, thereby limiting efficiency.

To address the drawbacks of decentralized architectures, the automotive industry is witnessing a shift towards centralized alternatives. Recent developments include domain-centralized (or domain-oriented), cross-domain centralized (or cross-domain-oriented), and vehicle-centralized (or zone-oriented) architectures. This evolution aims to overcome the limitations of current structures, reduce the number of required software releases, and enhance support for a variety of automotive software product lines.



Jaguar Land Rover has been successful in implementing platform-based hardware with domain-centralized control units. The figure below depicts this implementation in the electric Jaguar I-Pace model. They have integrated everything from the battery pack modules to the electronic control units in this skateboard platform. Jaguar Land Rover adopted an Ethernet backbone for their Electrical/Electronic (E/E) architecture in 2017. They have centralized the infotainment domain by designating a Domain Control Unit (DCU) to oversee all infotainment functionalities. Moreover, Jaguar Land Rover, has expressed intentions to implement dynamic network configuration and employ a service-oriented architecture (SOA) which aims to enhance flexibility in incorporating new features, effectively manage various vehicle variants, promote hardware and software reuse, and accommodate the growing need for increased bandwidth.

Domain-oriented architectures resolve numerous challenges inherent in decentralized architectures. However, they encounter two primary obstacles. Firstly, demanding functionalities like Advanced Driver Assistance Systems (ADAS) and autonomous driving systems necessitate extensive communication between Domain Control Units (DCUs), blurring the delineation of DCU boundaries. Secondly, the growing interactivity

among DCUs raises concerns about the adequacy of existing communication bandwidths for future Electrical/Electronic (E/E) architectures, especially as more sophisticated high-end functions emerge.

These shortcomings can be overcome by using a vehicle-centralized architecture. In this configuration, a unified control unit is implemented, which integrates all the functionalities of the previously distributed ECUs. This significantly reduces latency, and streamlines the system communication. Furthermore, the integrated control unit mitigates a lot of redundancies in the system and improves resource management. Centralized control could facilitate a more straightforward integration of high-end functions, such as ADAS, by avoiding complex interactions between separate domain control units. There may also be better optimization of communication bandwidth, potentially addressing concerns about bandwidth sufficiency for future architectures.

However, with traditional electronic technologies, we come across several challenges. Firstly, functional safety is put at risk, since the failure of a completely centralized ECU has more drastic effects than multiple interconnected ECUs. Secondly, harmonization of the multiple functionalities on a single ECU requires major modifications in the current standards and software available.

This is exactly where chiplets can open up a world of opportunities. Chiplets tackle the innate challenges of implementing a vehicle-centralized architecture. Instead of integrating all functions onto a single chip, chiplets are individual semiconductor components that serve specific functions. These chiplets can be designed independently and then assembled into a cohesive system. Since different functions are distributed across multiple chiplets, a failure in one chiplet does not necessarily impact the functionality of the entire system. Furthermore, chiplet architectures provide flexibility to adapt to changing standards by allowing for the replacement or addition of chiplets that comply with the latest specifications. This adaptability ensures that the system can keep pace with technological advancements.

The sections below entail a detailed description of each of our applications and how they have been implemented.

3 Integrated Master Control Unit

Our first application of chiplet-based technology in automobiles implements a vehicle centralized master vehicle control unit, which controls all the functionalities of the various ECUs used in a decentralized architecture.

In a traditional setup, the vehicle control unit acts as the brain of the entire vehicle, coordinating the operation of nearly all major control systems present in it. It ensures the safety and optimal operation of all the components. It performs various functions such as reference motor torque signal generation, monitoring the operation and charging of the battery pack, implementing regenerative braking, managing the cooling system, ensuring faults and errors are properly handled, and so on. The major functions performed by the VCU include:

- Motor Torque Command Validation
- Monitoring Battery SoC and SoH to ensure safe operation.
- Energy Management
- Implementation of regenerative braking algorithm.
- Charging Control
- Vehicle Diagnostics and Communication
- Safety and Fault Handling

Alongside the VCU, there are several other ECUs that work together and ensure the proper functioning of an EV. High-end EVs employ more than 100 ECUs working alongside each other. While the VCU looks after the high level operations of the entire vehicle, these specialised ECUs ensure the functioning of each major subsystem they monitor. They implement the commands of the VCU to ensure that the vehicle operates as expected. The major ECUs kept in place to target these functionalities include:

- Motor Control Module

- Transmission Control Unit
- Battery Management System
- Charging Control System
- Thermal Management System
- Auxiliary Battery Management System
- Electronic Stability Control Unit (ESC)
- Electric Power Steering Unit
- ABS Control Module
- Tire Pressure Monitoring System
- Body Control Module (BCM)
- HVAC Control Module
- Instrument Cluster Control Module
- ADAS and Autonomous Driving System
- In-vehicle communication and V2X connectivity
- Infotainment System ECU
- Telematics Control Unit
- Gateway ECU

As original equipment manufacturers (OEMs) incorporate an increasing number of features into their vehicles, the complexity of the in-car network surges due to a substantial rise in the number of Electronic Control Units (ECUs) and processing units required for implementation. This intricate system of numerous ECUs, confined within the limited space of a car, results in elevated costs and computational demands. The assigned complex tasks to each ECU contribute to significant power consumption, adversely impacting the vehicle's battery life and subsequently diminishing the effective real-time range. Simultaneously, the growing trend toward software-defined features in cars introduces advancements in algorithms and computing paradigms. Regular software updates necessitate compatibility with evolving hardware, presenting a notable challenge in maintaining harmony between the software and hardware components of the vehicle.

3.1 Proposal

In our approach, we intend to integrate all the major control units with the Vehicle Control Unit into a single chiplet based architecture, known as the Integrated Master Control Unit (IMCU).

Innovating automotive architecture, the system incorporates several key features to enhance efficiency and functionality. The centralization of all control units into a unified hub forms a core aspect, streamlining the management of computational tasks. These tasks are efficiently handled by a pooled processor, ensuring optimal utilization for tasks of similar magnitude. The architecture consists of three sub-units, each based on distinct processing clusters:

- Integrated Vehicle Control Unit (VCU)
- Autonomous Driving System (ADS)
- Infotainment and Connectivity Cluster

To facilitate seamless data transfer, a Shared Memory Module is implemented, connecting all three systems. Notably, major sub-components like the motor control module and State of Charge (SoC) estimation for the battery pack are software-defined, providing flexibility and adaptability.

The allocation of processing resources to various subsystems occurs in real time, promoting optimal hardware usage. Leveraging different technology nodes for distinct chiplets introduces versatility to the architecture. The hardware is both simplified and robust, ensuring compatibility with software across several generations.

The architecture boasts far greater power efficiency compared to the prevailing use of multiple Electronic Control Units (ECUs), contributing to an environmentally conscious and energy-efficient design. Lastly, the hardware is customizable according to Original Equipment Manufacturer (OEM) requirements, offering adaptability and flexibility in addressing specific automotive needs.

Advantages of a chiplet-based IMCU:

- Lower hardware requirements as multiple interfaces between various ECUs get eliminated, as all control units are in the same place.
- Most of the modules become software-defined. This implies that the same hardware can be used to support multiple generations of software updates; thus the life of the product increases.
- Processor pooling results in the availability of a powerful processor and flexible allocation of cores to various tasks, ensuring optimal performance and hardware utilisation.
- A single base platform can be used to design several cars just by changing the configuration of the chiplets.
- Usage of various processing nodes ensures that the more expensive smaller nodes are only utilised for systems that benefit from the additional performance (such as the processor). The other chiplets within the cluster may utilize a lesser technology node to ensure a good cost-to-performance ratio (such as IO interfaces).
- Smaller module size along with decrease in number of seperate ECUs required, implies better integration into structures with greater space constraints, such as a skateboard chassis type vehicle, and greater cabin space availability.
- Reduced latency and faster computation due to integration due to decrease in the length of the connections between components, and elimination of delays due to data conversion and transmission occurred in case of having multiple ECUs.
- Lesser power drawn compared to traditional architectures, increasing energy savings and extending the real-life range of the vehicle.

Overall Architecture of IMCU

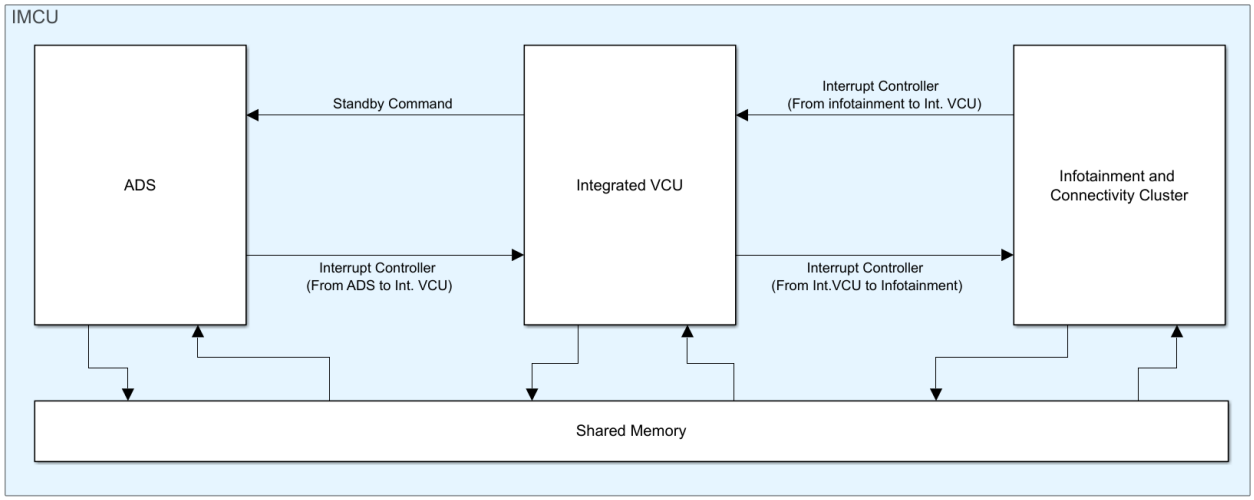


Figure 7: Overall Architecture of IMCU

3.2 Integrated VCU

The "Integrated VCU" subsystem of the IMCU incorporates all the functions present in a traditional VCU and integrates it with all other related ECUs (except for those integrated into the ADS and Infotainment and Instruments Cluster).

The following are the ECUs integrated in this subsystem:

- Motor Control Module
- Transmission Control Unit
- Battery Management System
- Charging Control System
- Thermal Management System
- Auxiliary Battery Management System
- Electronic Stability Control Unit (ESC)
- Electric Power Steering Unit
- ABS Control Module
- Tire Pressure Monitoring System
- Body Control Module (BCM)
- HVAC Control Module

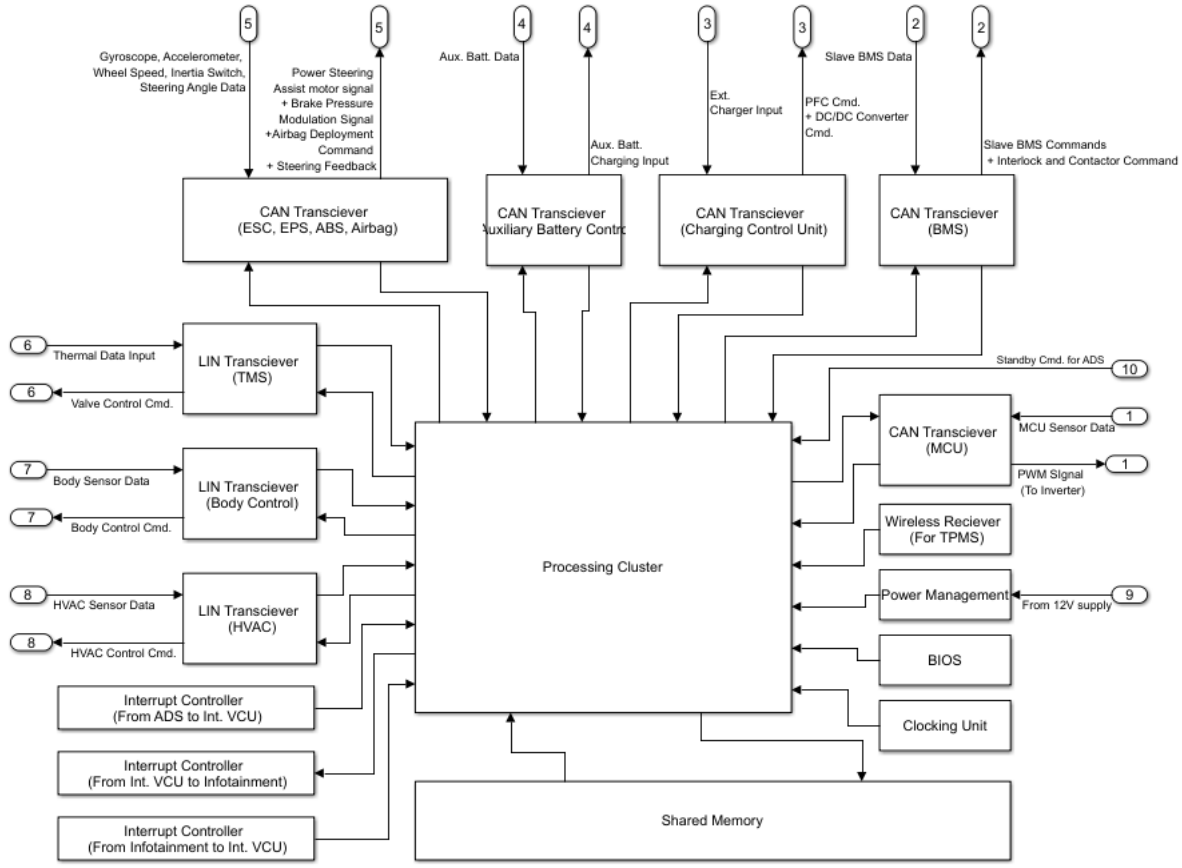


Figure 8: Integrated VCU subsystem of IMCU

In our implementation, we have a central processing cluster for the Integrated VCU subsystem, which is connected to various CAN and LIN interfaces, memory elements, and interrupt controllers. The processing cluster is powerful enough to handle the computation of all the ECUs that are integrated into this. By integrating the ECUs, the need for having multiple interfacing units to communicate between them is eliminated, and so is the latency.

The computations done by the individual ECUs are now offloaded to the chiplet-based processing cluster shown in the diagram. Hence, these ECUs are now essentially software defined, rather than a hardware unit. The outputs generated from these computations from each (now software-defined) system is stored in the shared memory unit. Any other system that needs the outputs of the computation or the data collected by the sensors, can access it from the shared memory.

In order to streamline data collection and transmission, we perform data-level sensor fusion for sensors that are physically close together, combining their raw data. This is then sent to the IMCU via a CAN bus. The IMCU receives the data, and the appropriate sensor data gets stored in the shared memory, from where all systems that require the data can access it. A more detailed explanation of the sensor fusion technique has been given later under the ADS subsystem section.

For sensors that transmit non-critical data, LIN has been utilized instead of CAN. The memory module shown is shared by all the subsystems of the IMCU, ensuring any data collected by one is available for utilisation by the other. For the transmission of data, the same CAN buses that were used by the sensors are used to transmit the required control signals to the appropriate actuators.

The functions performed by the Integrated VCU subsystem include:

- Determination of PWM signals for the motor inverter, based on user input from throttle pedal.
- Estimation of Battery SoC. SoH and temperature from the data provided by the slave BMS.
- Control of charging of the battery pack.
- Control of thermal management system pumps, ensuring proper cooling wherever required.
- Ensures stability of vehicle in dynamic driving conditions, and alerts the driver to any potential hazards.
- Provides inputs to the required servomotors for power steering assistance.
- Performs ABS control computations.
- Monitors the data obtained from the TPMS to ensure the safety of the vehicle.
- Controls the HVAC system of the car.
- Controls body functions such as automatic windshield wipers, etc.

The processing cluster architecture (using chiplets) is:

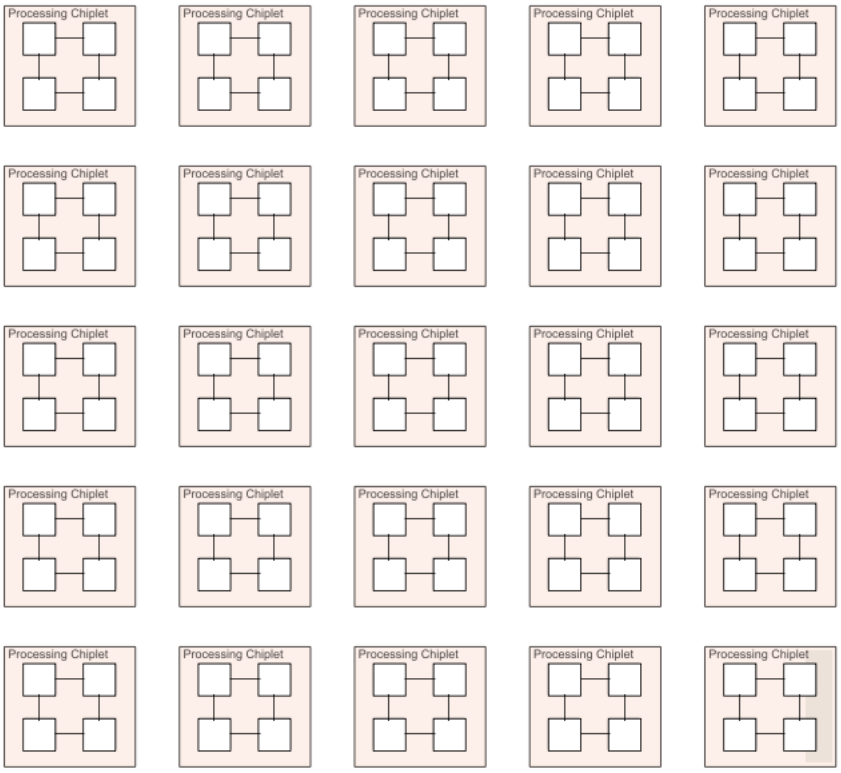


Figure 9: Integrated VCU processing cluster

3.3 Autonomous Driving System (ADS)

We aim to implement an Autonomous Driving System (ADS) into our system. Its purpose is to facilitate the autonomous driving of the car, with minimal human intervention. The ADS integrates the data from various sensors such as the LiDAR and RADAR, processes it, prepares a course of action, and executes it in collaboration with the Integrated VCU subsystem.

The systems integrated into the ADS subsystem include:

- **Perception Subsystem:** Cameras, LiDAR, RADAR, etc.
- **Localization Subsystem:** GPS, Inertial Measurement Unit, etc.
- **Planning Subsystem:** Route Planning, Trajectory Planning, Maneuver Planning
- **Control Subsystem:** Steering Control, Acceleration/Deceleration Control, Braking Control

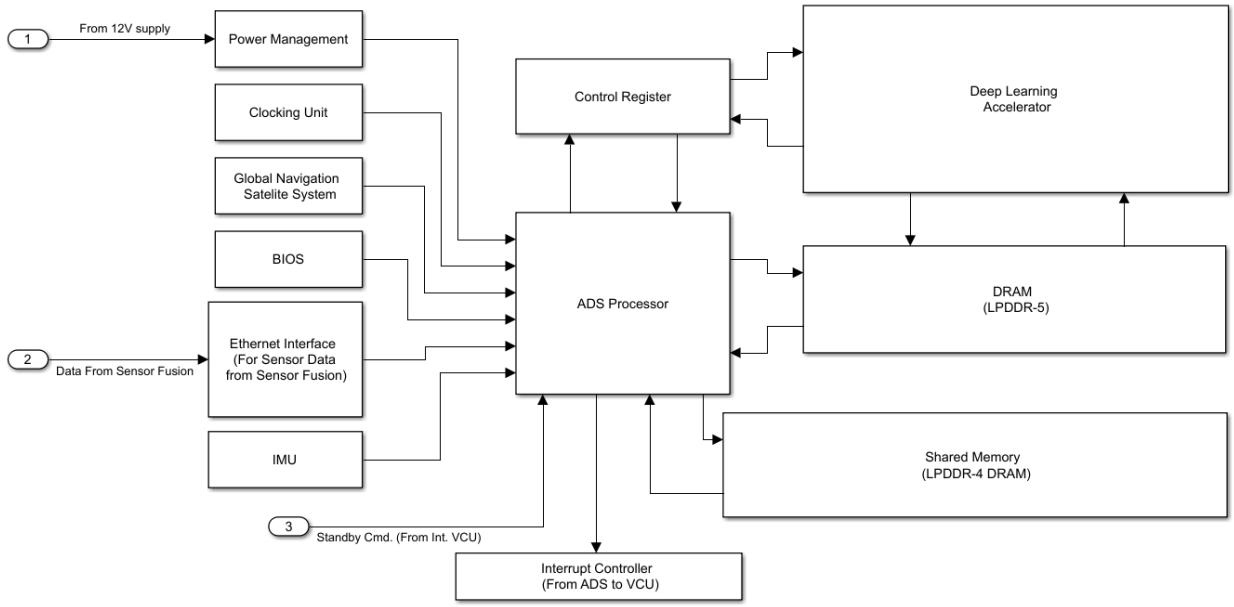


Figure 10: ADS subsystem of IMCU

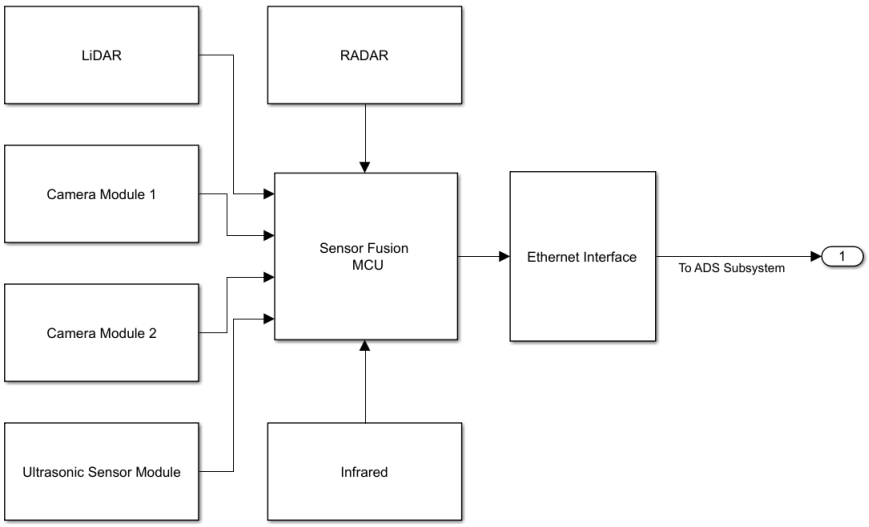


Figure 11: Sensor Fusion Diagram

In our implementation, the data from various peripheral sensors such as camera modules, LiDAR and Radar are combined by data level sensor fusion. The combined data is transmitted to the ADS subsystem of the IMCU by using Ethernet. The ethernet interface receives the data and passes it onto the ADS processor. The processor also receives data from various other sensors and systems such as the Inertial Measurement Unit (IMU) and the Global Navigation Satellite System that are directly connected to it. It can also access data collected by the Integrated VCU and the Infotainment and Connectivity Cluster subsystems via the Shared Memory module (LPDDR-4 DRAM). The processing involving Deep Learning Agorithms is done by the Deep Learning Accelerator of the system. Any other processing is done by the ADS processor itself. The processor is also connected to an interrupt controller that sends interrupts to the Integrated VCU processing cluster to perform whatever task is necessary. There is also a standby command line coming from the Integrated VCU to the ADS processor, in case the ADS system has to be put on standby. The ADS processor is made of a cluster of chiplets-based processor cores, making the processor scalable according to the complexity of the algorithms and level of computation we require it to perform.

We propose a heterogeneous dataflow accelerator (HDA) as an alternative to traditional computing for DL applications in the ADS subsystem. This is a composite system consisting of subsystems that focus on optimizing the flow of data between the distributed components. These are accelerators based on dataflow

principles and offer adaptability by the use of several sub accelerators each implementing distinct dataflows, thus specialising in computation of certain layer types.

We also propose the implementation of a neuromorphic processor for each DL/ML application. This is a processor divided into coresets and cores to manage fan-in and fan-out for each core. The cores contain neural networks according to the SNN model that we need to implement. These processors are extremely power efficient and provide low latency since they are the direct implementation of the neural network itself in silicon instead of a processor computing the inference.

The exact implementation of this subsystem is described in detail in a further section.

3.4 Infotainment and Connectivity Cluster

This subsystem integrates the infotainment ECU and all other ECUs involved with data logging and telemetry. Its main purpose is to perform all Infotainment functions along with all V2X functions required by the car. It interacts with the user interface and conveys all the information that needs to be passed on to the driver and other passengers. It integrates various connectivity functions, such as WiFi, Bluetooth, etc, for entertainment systems and also supports smart connection technologies such as IoT integration and V2X communication.

The following ECUs of a traditional architecture are integrated into this subsystem:

- Instrument Cluster Control Module
- Infotainment System ECU
- Telematics and Connectivity Control Unit

Infotainment Instrument Interfaces
Several wired interfaces such as HDMI, USB, and Ethernet connect with various infotainment instruments, including touchscreen displays. These interfaces convey critical data to the driver and enable features like radio playback and multimedia content from the internet or USB media devices.

V2X Connectivity in Intelligent Transportation Systems
Vehicle-to-Everything (V2X) connectivity is crucial for Intelligent Transportation Systems (ITS), facilitating communication between vehicles, infrastructure, and elements within the transportation ecosystem. V2X includes Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), Vehicle-to-Cloud (V2C), and Vehicle-to-Infrastructure/Internet of Things (V2IoT) communications. These leverage wireless technologies like DSRC/VANETs, cellular networks (LTE and 5G), Zigbee, UWB, Wi-Fi, WiMAX, RFID, and Bluetooth.

To ensure secure data transfer between the car and the cloud, a Hardware Firewall is implemented. It monitors network traffic, allowing only legitimate connections and employing techniques such as Network Address Translation (NAT) and Virtual Private Networks (VPN) to enhance security.

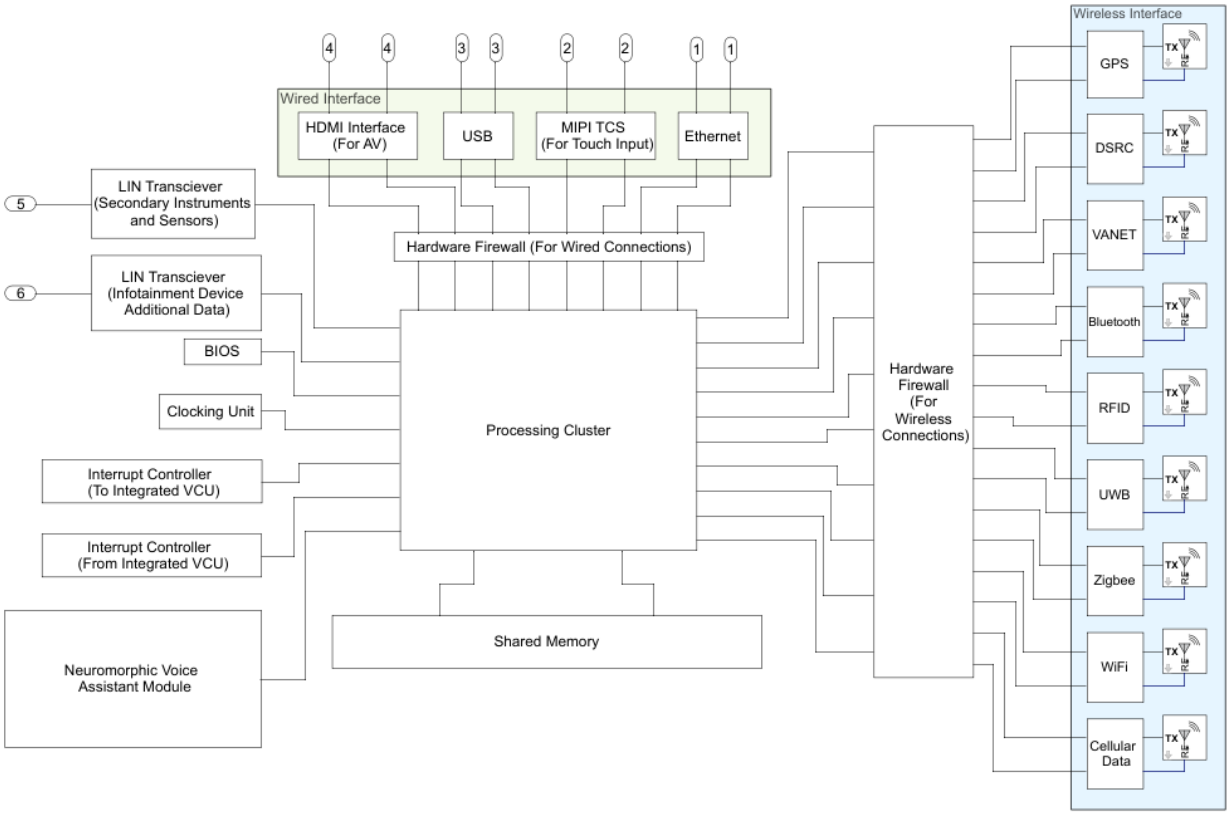


Figure 12: Microarchitecture of Infotainment and Instruments Cluster of IMCU

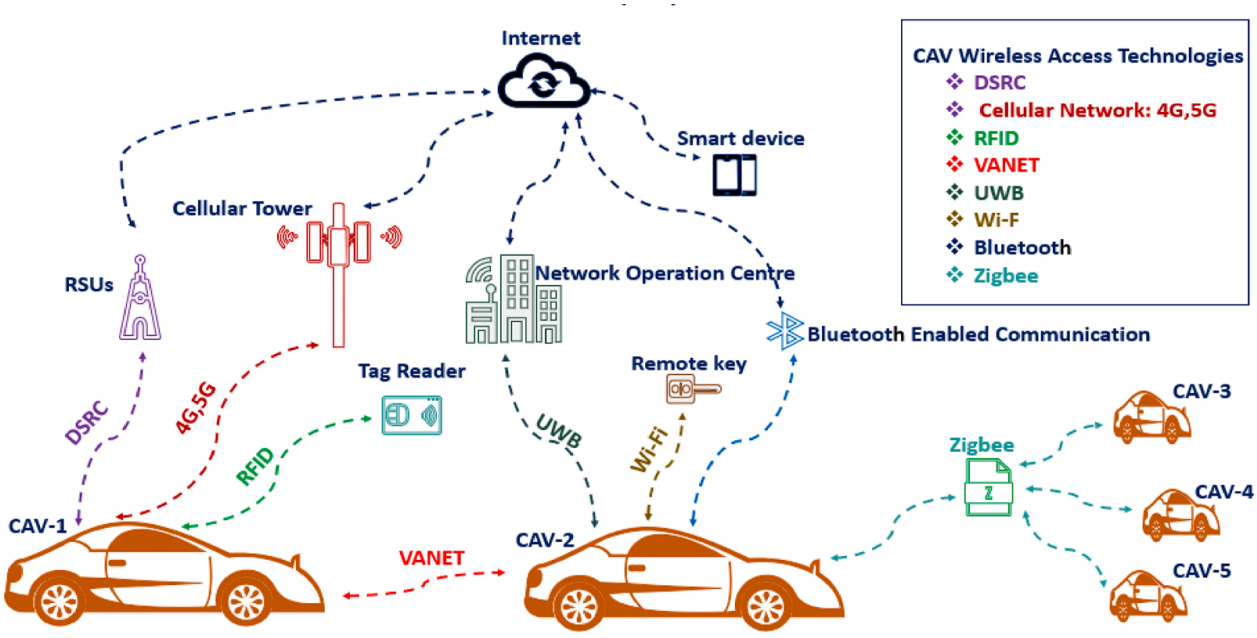


Figure 13: V2X Communication

Communication within Vehicle Subsystems

The Infotainment and Connectivity Cluster shares data and instructions with the Integrated Vehicle Control Unit (VCU) through a shared memory module accessible to all three subsystems. Two interrupt controllers facilitate the exchange of instructions between the Infotainment and Connectivity Cluster and the Integrated VCU subsystem.

Speech Recognition in Human-Machine Interface (HMI)

Speech recognition plays a pivotal role in the HMI of modern cars, enhancing user experience and safety.

It enables hands-free interaction, allowing drivers to control various functions through voice commands, such as making calls, sending messages, adjusting navigation settings, and managing multimedia playback. Advancements driven by natural language processing and machine learning algorithms contribute to better understanding diverse accents, languages, and commands.

Neuromorphic circuits can be implemented for speech recognition, offering varying degrees of functionality. One approach involves a wake-up module that triggers the main processing system upon detecting specific keywords, improving power efficiency. Another approach implements a complete neuromorphic system using a Spiking Neural Network (SNN) architecture for speech recognition, achieving energy efficiency compared to traditional processors.

In the implementation of a keyword spotting model, a Loihi neuromorphic processor running speech2spikes demonstrated significantly lower energy usage compared to GPU and CPU alternatives. This efficiency extends beyond keyword spotting, with orders-of-magnitude improvements observed across various tasks.

The Processing cluster architecture for the Infotainment and Connectivity cluster (using chiplets) is:

Infotainment and Connectivity Cluster Processor

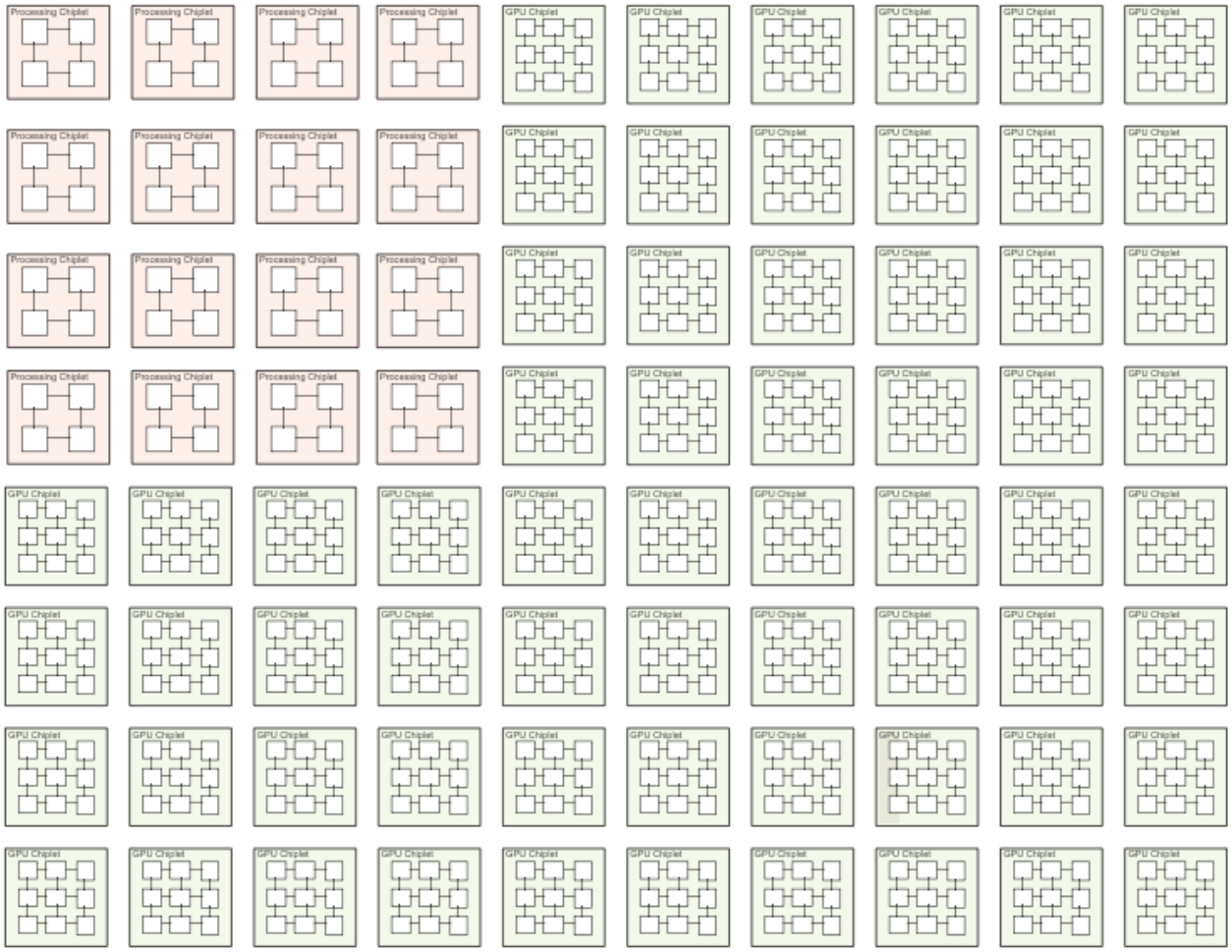
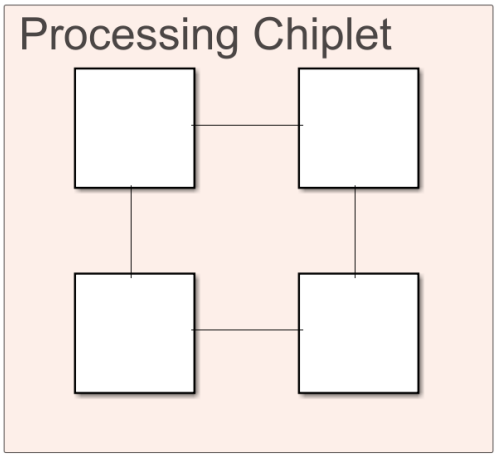
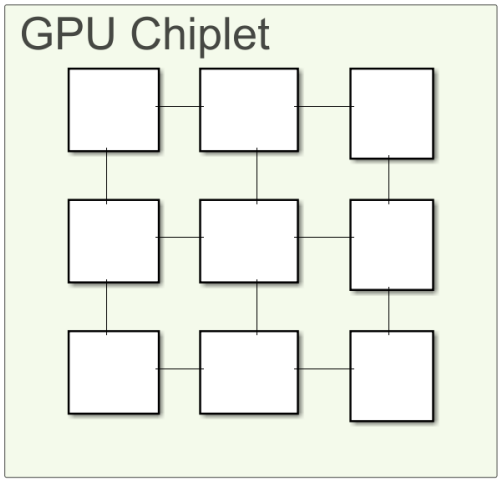


Figure 14: Infotainment and Connectivity Cluster processor



(a) CPU Chiplet



(b) GPU Chiplet

4 Autonomous Driving System

The second application builds upon the autonomous driving subsystem by synergizing the benefits of chiplets and novel application-specific hardware. Our proposed solution leverages these advantages to achieve significantly improved results, surpassing what could be attained with conventional hardware.

Autonomous driving makes use of advanced machine learning and deep learning to navigate and maneuver through roads effectively. The use of general-purpose computing for machine learning and deep learning applications does not make optimal use of the hardware resources that are allocated for the same. This is due to the fact that general-purpose architectures are better suited for a wider range of applications rather than being application-specific. This results in wasted clock cycles and unnecessary data transfers, which adds to the overall latency, and inefficient power usage.

By making use of dedicated hardware, one can optimize the data flow, thereby achieving lower latency and minimal power consumption. For this reason, we propose the use of a dedicated hardware architecture for machine learning and deep learning applications.

Chiplets introduce a transformative paradigm in semiconductor design, facilitating the realization of heterogeneous data flow architectures and neuromorphic computing. One key advantage of chiplets lies in their ability to accommodate specialized functionality. Scalability is another crucial aspect enabled by chiplets. Designers can mix and match chiplets based on application requirements, providing scalability that is particularly beneficial for addressing the varied computational needs of neuromorphic computing. This flexibility allows for the creation of highly adaptable systems that can be tailored to specific performance demands.

4.1 Proposal

The proposal is divided into two approaches towards the problem in order to optimise the architecture for DL inference. The first is the design of a processor to which all the DL workloads are mapped by the central VCU controller. This processor has a heterogeneous architecture which allows it to map multiple layers onto hardware partitions in a way that optimises the overall performance.

The second approach is the use of newer in-memory processing technology and neuromorphic system design. The system architecture is inspired by the human brain and tries to mimic the same behaviour. The basic compute units are the neurons and the synapse which comprises memristive elements that model the distance between two neurons. Thus this involves the actual implementation of the neural network rather than having a processor process the model.

We also propose the use of Design Space Exploration (DSE) for the optimization and mapping of hardware resources for both architectures. The design space is defined by the parameters that can be changed in the design and maps to the cost that we define on the performance of the design. Since design space parameters are associated with the IP that we use in the design, DSE also entails the database generation containing the mapping of the IPs to the design space parameters. The cost metric calculation is usually

performed by a simulator that works in conjunction with the search heuristic.

The following sections expand more on the various components of the proposal.

4.2 Heterogeneous Data Flow Accelerators

We propose a heterogeneous dataflow accelerator (HDA) as an alternative to traditional computing for DL applications in the ADS subsystem. This is a composite system consisting of subsystems that focus on optimizing the flow of data between the distributed components. Dataflow architectures can be classified as follows:

- **Fixed Dataflow Accelerators**: These are accelerators implementing only a single dataflow. These kinds of accelerators perform well for certain types of layers but are suitable for others.
- **Reconfigurable Dataflow Accelerators**: These ASIC based accelerators follow a coarse-grained reconfigurable architecture. This flexibility allows the accelerator to change the type of dataflow between the processing elements in such a way that is optimised for the layer which is being computed. The downside to this flexibility is the additional hardware components such as switches and wires for re-routing data. Thus these are not very efficient when it comes to inference on edge devices where space and hardware resources are limited.
- **Heterogenous Dataflow Architecture**: These are accelerators based on dataflow principles and offer adaptability by the use of several sub-accelerators each implementing distinct dataflows, thus specialising in computation of certain layer types.

4.2.1 Design Considerations for HDA

While designing an HDA, we need to take into consideration some parameters depending on the workload that is to be offloaded to the accelerator. These are described as follows:

- **Dataflow Selection for Sub-Accelerators**: The kind of layers that are dominant in the workload determines the kind of sub-accelerators that need to be included in the design. This greatly influences the efficiency of the accelerator. In general, these sub-accelerators have considerable diversity among them to serve different layer operations.
- **Hardware Resource Partitioning**: The resource available has to be divided among the different sub-accelerators. Since these accelerators are repeating in nature, it is easy to scale them. The efficiency of the accelerator depends on the resource allocation for the architectures according to the distribution of layer operations in the whole workload.
- **Layer Scheduling**: Since we have multiple sub-accelerators that can work independently, we need a scheduler that, depending on the workload, schedules layer computations to different sub-accelerators optimally. This influences the latency and the energy consumption of the accelerator.

To tackle the challenge that is posed by these design considerations, we propose the use of a Design Space Exploration algorithm to get a smaller design space set with similar costs and satisfying the constraints. The DSE algorithm is explained in the following section.

4.2.2 DSE Implementation

For the design space exploration for optimization of the hardware resource allocation and scheduling of the workload, we propose the use of HERALD. The framework takes in total amount of hardware resources, dataflows of sub-accelerators and target DNNs to run as inputs and gives out the hardware resource partitioning for each sub-accelerator, layer execution schedule and an estimate of the latency and energy consumption. After deployment too, for a given architecture, HERALD can still generate an optimised scheduling.

In recent dataflow accelerators, we see that the types of heterogeneity in a DNN model is based on the layer shape and layer operation. Classification networks have a dwindling size of layers as we go deeper into the network whereas segmentation networks need to retain the size of image at each layer and thus maintain the same layer size throughout. In terms of operation performed by a layer, we classify them as simple and complex layers. Complex layers are made up of simple layer operations. The operation themselves differ between the various layers but there can also be differences in the way complex layers are composed by the use of simple layers.

HERALD provides a framework based on which the designer has to set the parameters for design space exploration. We specify the cost functions based on the use case such as area and power concerns in edge computing devices. The DSE assumes the following general steps involved in the inference:

- Fetch filter values from DRAM and store them in a global buffer.
- Distribute filter values to sub-accelerators based on the layer execution schedule.
- Fetch activation from DRAM and store them in the global buffer.
- Stream activation values to their corresponding sub-accelerators based on layer execution schedule.
- Store streamed-out output activation from each sub-accelerator to the global buffer.
- while sub-accelerators compute output activation, fetch next filter values from DRAM and send the filter values to the next accelerator.
- When a sub-accelerator finishes executing a layer, stream output activation stored in the global buffer as input activation of the next layer.

HERALD allows the designer to specify the search algorithm as exhaustive search, binary sampling based search or random search. For layer scheduling HERALD implements a greedy search for latency, energy and EDP with global load-balancing. Once the layers are assigned to each sub-accelerator, The order of the layers is determined. This is done using either a breadth first or a depth first search. Depth first search results in large idle times because of the layer dependence within the model but also has lesser number of context switches as compared to scheduling using a breadth first search.

4.2.3 Sub-Accelerators

Our proposal includes the use of three sub-accelerator for handling the major neural network computation workloads, although more can be added and used as the need arises. These are as follows:

- ShiDianNao
- Eyeriss
- EdgeDRNN

We expand on each of these under the following headings.

4.2.4 ShiDianNao

ShiDianNao is a neural network accelerator the is based on output stationary data flow architecture which is specifically optimized for convolutional neural networks (CNN).A ShiDianNao accelerator constructed using a TSMC 65nm technology has a peak performance of 194 GOP/s ,power consumption of 32.0mW and an area consumption of $4.86mm^2$.

Architecture

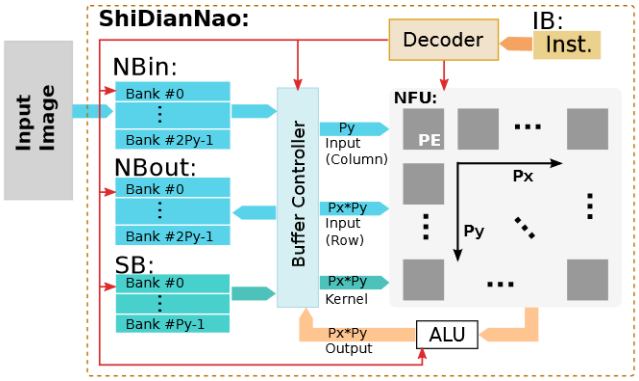


Figure 16: Architecture of ShiDianNao

Neural Function Unit (NFU)

The NFU structure comprises Px × Py Processing Elements (PEs), aligning naturally with the 2D structure of feature maps. Intra processing element communication is implemented to reduce the band width , thus reducing the complexity in the wiring. Intra processing element communication helps to reduce the required bandwidth by reusing the input given to the processing element on the right and bottom of all the processing elements in the 2D array of processing elements.

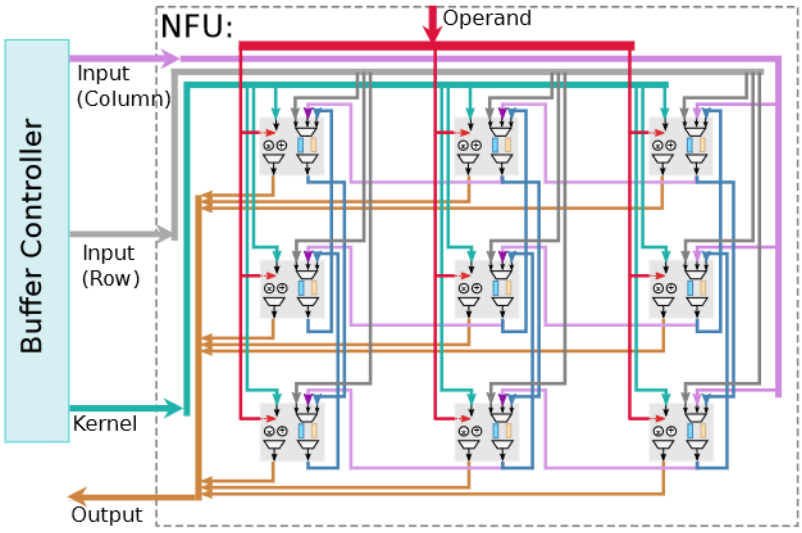


Figure 17: Architecture of Neural Functional Unit (NFU)

- **Processing elements** A processing unit can execute various operations such as multiplication and addition for convolutional, classifier, or normalization layers, solely an addition for an average pooling layer, or a comparison for a max pooling layer, among others, within a single cycle. It possesses three inputs: one for receiving control signals, another for accessing synapses (like kernel values in convolutional layers) from SB, and a third for retrieving neurons from input/output buffers or adjacent processing units. Furthermore, a processing unit features two outputs: one for storing computation outcomes in input/output buffers and another for transmitting locally-stored neurons to neighboring processing units.

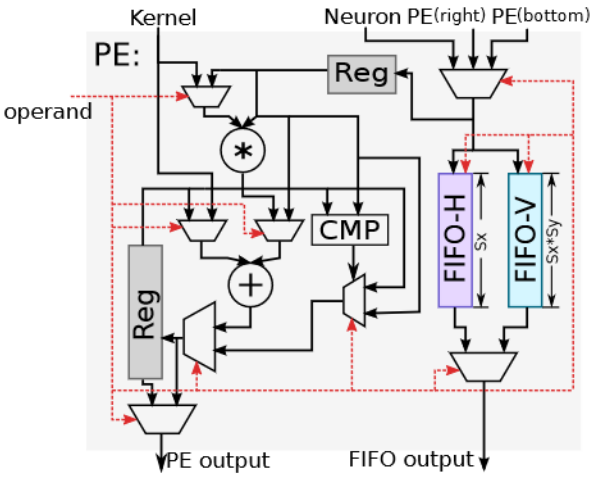


Figure 18: Architecture of the Processing Element

- **Inter-PE data propagation** In CNNs, the output neurons in convolutional, pooling, and normalization layers rely on data from neighboring input neurons within a rectangular window. Typically, these windows significantly overlap for adjacent neurons. However, fetching these overlapping windows for

processing repeatedly from the buffer to various processing elements demands substantial bandwidth, potentially causing wiring complications.

To optimize data reuse and alleviate bandwidth issues, we introduce a method for inter-PE data transmission within the PE mesh. This approach enables each processing element (PE) to share locally stored input neurons with its adjacent left and lower neighbors. Implementation involves incorporating two First In First Out registers (FIFOs), namely FIFO-H and FIFO-V, within each processing element. FIFO-H accumulates and transmits data from the input/output buffer (NBin/NBout) and from the right neighbor PE, distributing this information to the left neighbor for reuse. Meanwhile, FIFO-V stores and shares data from NBin/NBout and the upper neighbor PE, disseminating this data to the lower neighbor PE for reuse.

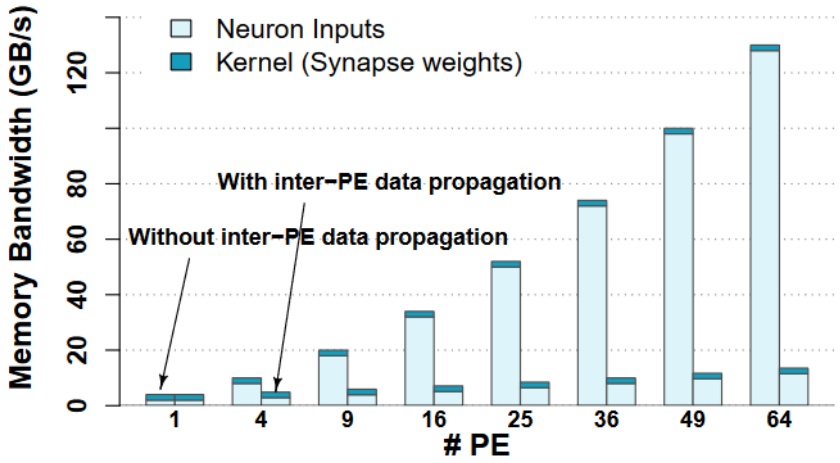


Figure 19: Bandwidth required vs number of processing elements used

Through this inter-PE data propagation mechanism, the internal bandwidth requirements can be significantly diminished, thereby mitigating the demand for high bandwidth within the system

- **Arithmetic Logic Unit (ALU)** A lightweight Arithmetic Logic Unit (ALU) is employed to supplement the processing elements. Within this ALU, various 16-bit fixed-point arithmetic operators are integrated, encompassing division (utilized for average pooling and normalization layers) along with non-linear activation functions such as tanh() and sigmoid() (applied in convolutional and pooling layers). The computation of activation function values is accomplished using piecewise linear interpolation (represented by f(x) = a_ix + b_i, when x exists within the range [x_i, x_{i+1}], where i ranges from 0 to 15). This method is recognized for causing only minimal accuracy loss in Convolutional Neural Networks (CNNs). To facilitate efficient computation, segment coefficients ai and bi are pre-stored in registers, allowing for the approximation to be swiftly computed using a multiplier and an adder.

Storage

An on-chip SRAM is used as the storage for this neural network accelerator which is further divided into NBin to store input neurons , NBout to store output neurons and SB to store intermediate results. The NBin and NBout must be sufficiently large to store the neurons of a whole layer.The NBin and NBout has 2*P_y banks of width 2*P_x bytes each where P_y and P_x are the number of processing elements in a column and a row of the Neural Functional Unit respectively.

Controls

Buffer controllers

On-chip buffer controllers facilitate effective recycling of data and computation within the NFU (Neural Function Unit). For instance, the NB controller, responsible for managing read and write operations in NBin and NBout, efficiently facilitates six read modes and one write mode. These six read modes are employed in various combinations to handle computations required for different layers in a convolutional neural network, such as:

- Read multiple banks from the beginning.
- Read multiple banks from between.
- Read one bank.
- Read a single neuron.
- Read neurons with a given step size.
- Read a single neuron per bank.

Control instructions

Control instructions encompass a set of procedures essential for executing various operations like convolutions, pooling, etc. These instructions are stored in a two-tiered Hierarchical Finite State Machine (HFSM) to outline the execution sequence of the accelerator. Within the HFSM, the first-tier states delineate abstract tasks performed by the accelerator, such as different layer types or ALU tasks. Each first-tier state is associated with multiple second-tier states that define the corresponding detailed low-level execution events. For instance, the second-tier states linked with the first-tier state Conv (convolutional layer) represent phases necessary for processing an input-output feature map pair.

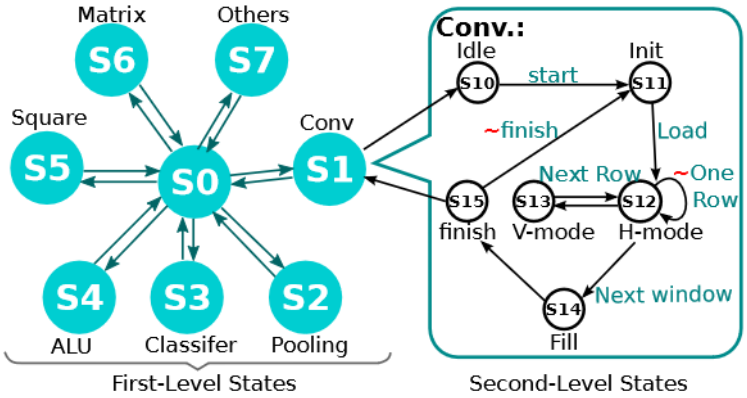


Figure 20: Hierarchical control finite state machine

To represent each HFSM state and its related parameters (e.g., feature map size), a 61-bit instruction is utilized. These instructions can be decoded into precise control signals for a specific number of cycles in the accelerator. This methodology ensures minimal loss of flexibility in practical application. For instance, a CNN requiring 50,000 cycles only demands 1 KB of instruction storage and a lightweight decoder that occupies a mere 0.03 mm^2 in our 65 nm process technology.

Dataflow

Convolutional layer

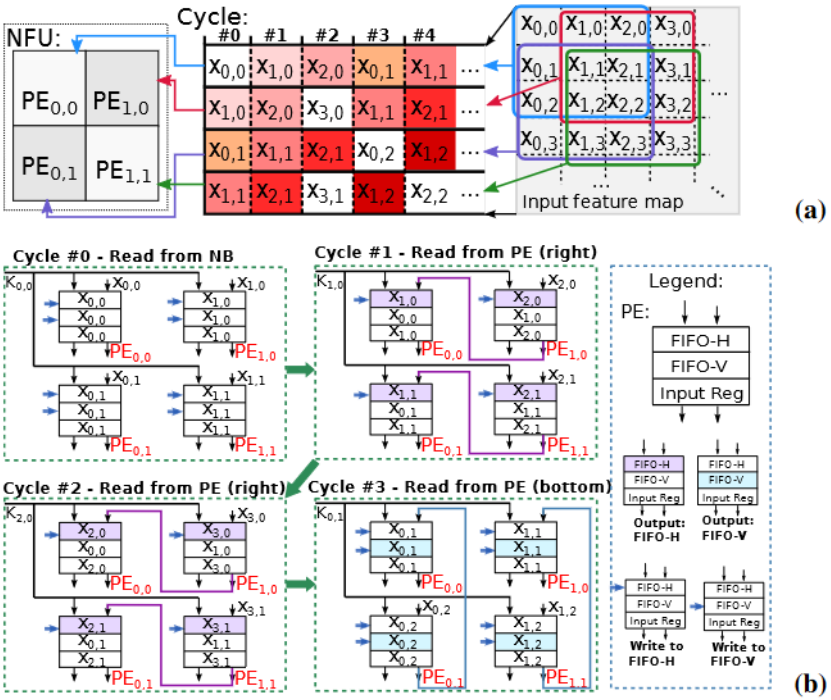


Figure 21: Convolution performed by ShiDianNao

Each individual processing element (PE) within the NFU (Neuron Feature Unit) is dedicated to computing the values for a single output neuron within a convolutional layer. When calculating each output feature map, every PE in the accelerator remains focused on processing a sole output neuron continuously, without switching to another until the current one is computed. The multiplication and accumulation operations for a single output neuron occur sequentially within a series of cycles performed by a single processing element.

Initially, the processing element retrieves data from the Nbin. Subsequently, during the computation of each row of the kernel for the multiply and accumulate operation, the next input neuron is accessed from the FIFO-H (First-In-First-Out buffer for horizontal access) of the processing element to the right. After accessing all the input neurons along a row of the kernel, the subsequent neuron is retrieved from the FIFO-V (First-In-First-Out buffer for vertical access) of the processing element positioned below the currently described processing element. This sequential process continues until all input neurons are obtained and processed.

Pooling layer

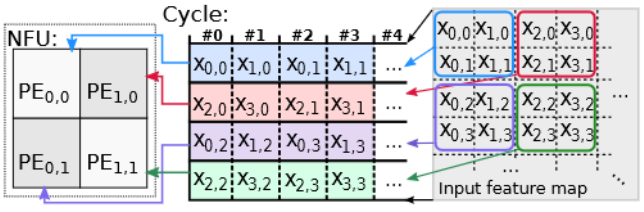


Figure 22: Pooling performed by ShiDianNao

For pooling operation each processing element in the NFU access input neurons only from a single window. As there is no overlap between adjacent windows while pooling in general there is no inter processing element communication. On the cases where there exist overlap between windows inter processing element communication is used.

Classifier layer

In a Convolutional Neural Network (CNN), the convolutional layers enable different pairs of input and output neurons to share synaptic weights using the kernel, while pooling layers do not involve synaptic weights. Conversely, classifier layers typically consist of fully connected neurons without weight sharing among different input-output neuron pairs. Consequently, classifier layers often occupy the most space within the synaptic array, such as 97.28% for LeNet-5. During the execution of a classifier layer, each Processing Element (PE) handles a singular output neuron and remains dedicated to its computation until completion.

In contrast to a convolutional layer, where each cycle involves reading a single synaptic weight and $P_x \times P_y$ different input neurons for all $P_x \times P_y$ PEs, a cycle in a classifier layer requires reading $P_x \times P_y$ distinct synaptic weights and a single input neuron for all $P_x \times P_y$ PEs. Subsequently, each PE performs multiplication between the synaptic weight and input neuron, accumulating the outcome into the partial sum stored in its local register. After several cycles, when the dot product (relating input neurons and synapses) linked with an output neuron is computed, the outcome is forwarded to the Arithmetic Logic Unit (ALU) for the activation function computation.

Normalization Layers

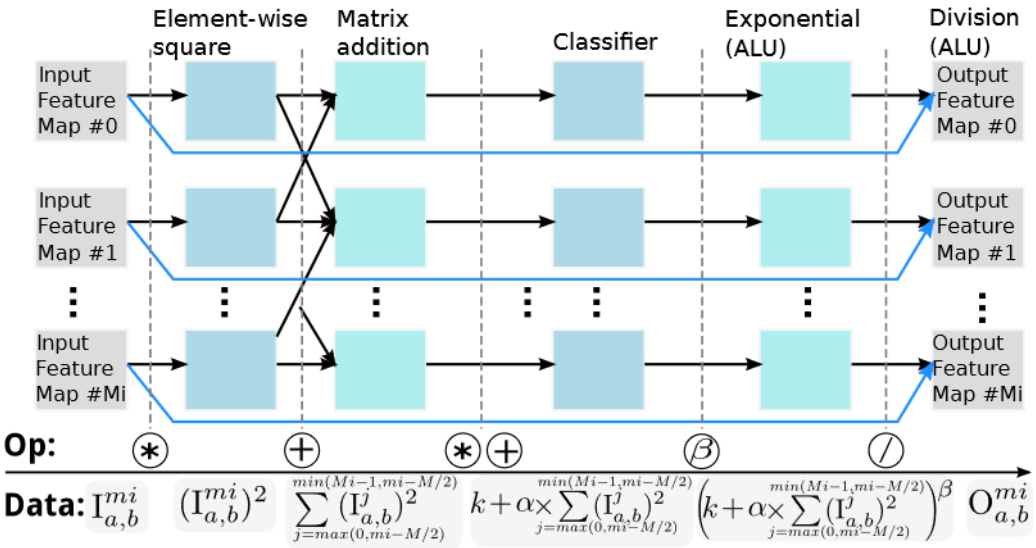


Figure 23: Decomposition of an LRN layer.

Normalization layers can be broken down into several sub-layers and basic computational operations to enable execution by our accelerator. For instance, an LRN (Local Response Normalization) layer is decomposed into various elements including a classifier sub-layer, an element-wise square operation, a matrix addition, exponential functions, and divisions. Similarly, an LCN (Local Contrast Normalization) layer is broken down into multiple components such as convolutional sub-layers, a pooling sub-layer, a classifier sub-layer, matrix additions, element-wise square operations, and divisions.

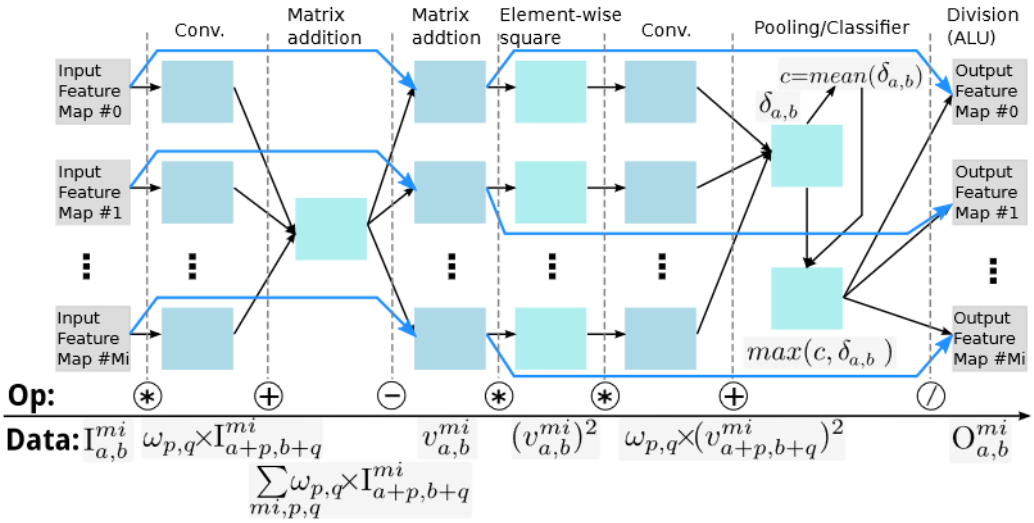


Figure 24: Decomposition of an LCN layer.

Convolutional, pooling, and classifier sub-layers follow rules outlined in earlier sections, while exponential functions and divisions are handled by the Arithmetic Logic Unit (ALU). The remaining computational primitives , like element-wise square operations and matrix additions, are managed by the Neuron Feature Unit (NFU). Supporting these primitives involves each Processing Element (PE) working on a matrix element output using its multiplier or adder in each cycle. Eventually, the results from all Px × Py PEs are written to NBout.

4.2.5 EdgeDRNN

The EdgeDRNN is designed specifically for efficient edge RNN inference at a low latency, particularly with a batch size of 1. It utilizes the delta network algorithm inspired by spiking neural networks to take advantage of the temporal sparsity present in RNNs. This approach involves storing weights in cost-effective DRAM, allowing EdgeDRNN to perform computations for large multi-layer RNNs on budget-friendly FPGA hardware.

Operating at a batch size of 1, EdgeDRNN achieves a mean effective throughput of 20.2 GOp/s and demonstrates a wall plug power efficiency that is more than 4 times higher than that of commercial edge AI platforms.

The primary components of EdgeDRNN include:

- The Delta Unit responsible for encoding delta vectors and generating weight column pointers (pcol).
- The Processing Element (PE) Array designed for matrix-sparse vector multiplications.
- The Control (CTRL) module housing finite state machines (FSMs) and encoding instructions to manage the AXI Datamover.
- Additional modules such as the Configuration (CFG) module composed of configuration registers, the Output Buffer (OBUF) for buffering and redirecting outputs back to the Delta Unit, and the W-FIFO for buffering weights.

Architecture

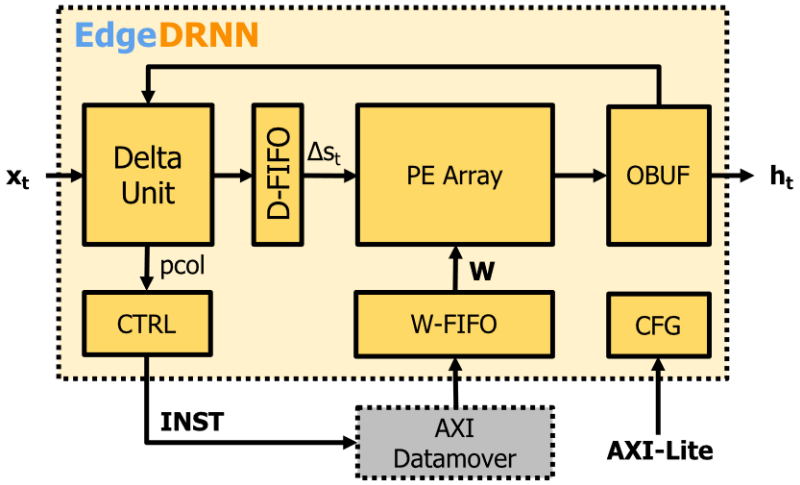


Figure 25: Architecture of EdgeDRNN.

Delta Unit

The Delta unit serves the purpose of computing alterations in the input vector (x_t) and the prior neuron activation (h_{t-1}) concerning their preceding values. It operates based on the state of a finite state machine, selecting either (x_t) or (h_{t-1}) for processing, handling one element of the chosen vector's delta state per cycle. Upon processing an element, the change in state is compared to a threshold value. If this change surpasses the threshold, the new delta state vector elements meeting or exceeding the threshold and their respective physical weight column addresses (pcol) are sent to the D-FIFO and CTRL. Simultaneously, the corresponding state element (st) is updated in the BRAM to modify the state memory. Conversely, if the elements fall below the threshold, they are disregarded and not added to the D-FIFO. To minimize latency, employing multiple Delta units (N units) enables the processing of subsections of the state elements (st) in segments of length N.

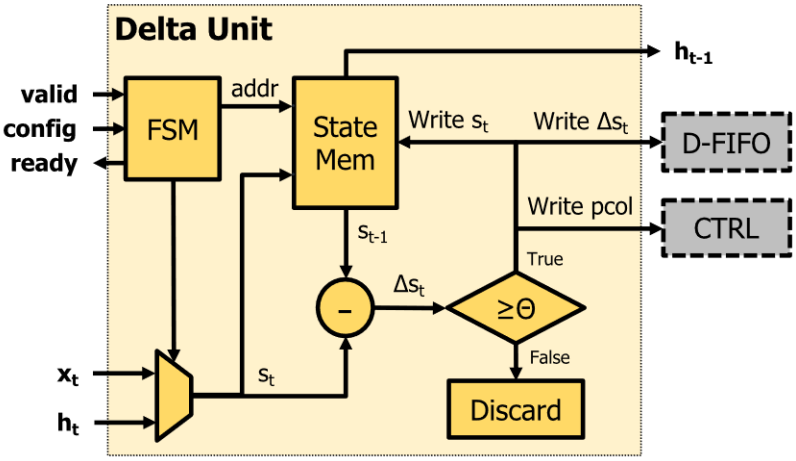


Figure 26: Architecture of delta unit.

CTRL unit

The CTRL module contains FSMs that control the PE array. This module generates 80-bit instructions for controlling the AXI Datamover to fetch RNN parameters. The instruction contains pcol and the burst length calculated from the dimensions of the network stored in configuration registers.

Processing Element Array

The Processing Element (PE) consists of a 16-bit multiplier (MUL) and two adders: a 32-bit ADD0 and a 16-bit ADD1. To enable versatility for both MxV and vector dot products, multiplexers are positioned before the MUL's operands. ADD0's multiplexer allows the selection between '0' and BRAM data, opting for '0' during BRAM initialization. ADD1 manages element-wise vector additions. All these units are parameterized in System Verilog RTL, allowing configuration during compile-time to support any fixed-point precision within their designated bit width. Additionally, the PE facilitates tanh and sigmoid functions through lookup tables (LUTs), with a fixed 16-bit input bit width for the LUTs and an adjustable output bit width ranging from 5 to 9 bits.

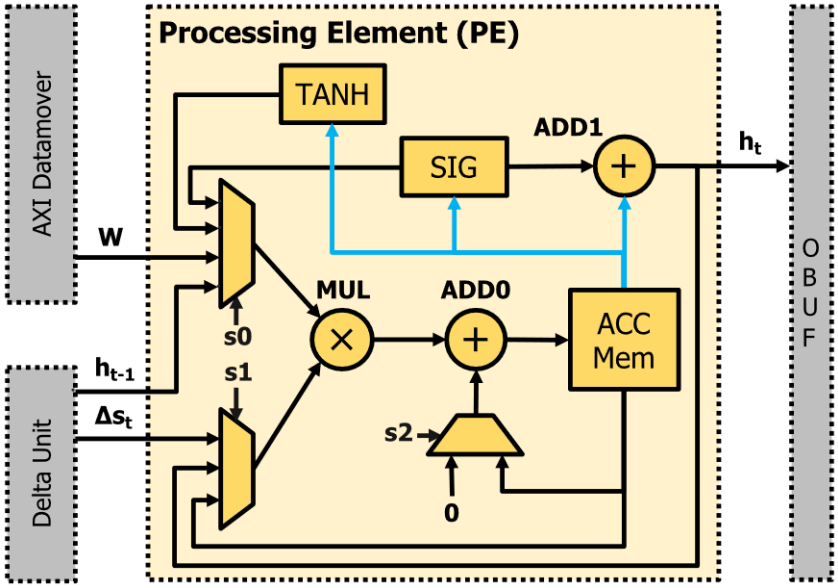


Figure 27: Architecture of processing element

Dataflow

The Delta Unit and the control unit work together to estimate key parameters, including delta vectors and physical weight column pointers. These parameters are then passed on to the D-FIFO and CTRL modules for further processing. The GRU-RNN's weight matrices are combined, and biases are added as the first column. Additionally, an element of 1 is appended to each input state vector as the first element.

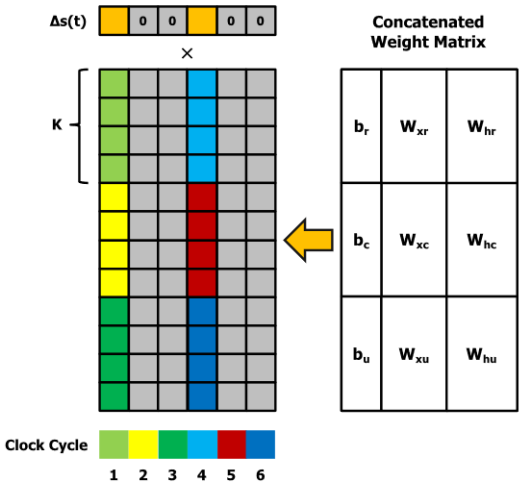
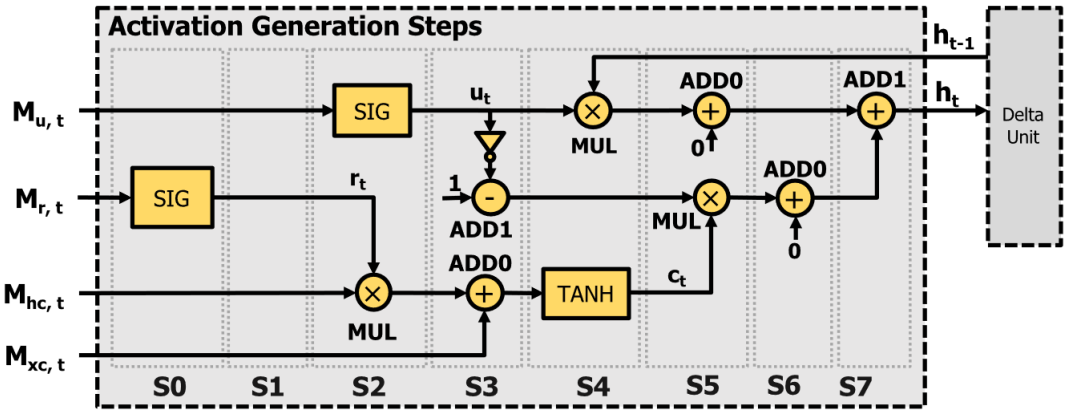


Figure 28: Dataflow of matrix vector multiplication (MxV).

The Processing Array (PE array) plays a crucial role in the efficiency of EdgeDRNN. It performs selective multiplication , only processing non-zero delta state elements with their corresponding valid columns. The

products are accumulated in the Accumulation Memory (ACC Mem) to compute delta memory vectors.

Furthermore, the PE array is responsible for calculating the activation state h_t after the matrix-vector multiplication (MxV). It fetches the delta memory vectors from the ACC Mem and utilizes an 8-stage pipeline to calculate h_t. Flip-flops are employed to buffer paths without any operator in any stage for one clock cycle.



4.3 Neuromorphic Approach

We introduce neuromorphic computing as an alternative to traditional computing for the DL applications in the ADS subsystem. Neuromorphic computing is brain inspired computing paradigm where the neural network is implemented as a physical interconnection of semconductor neurons.

4.3.1 Neuromorphic Computing vs ANN

Neuromorphic computing and traditional AI differ significantly in terms of their approach to processing information. While AI relies on algorithms and rules to process information, neuromorphic computing uses hardware that mimics the structure and function of the brain. Neuromorphic computing is designed to process information more naturally and efficiently, with less power and greater speed. IBM TrueNorth achieves a power consumption as low as 70mW per chip, and consumes 176,000% less energy per event compared to an optimized simulator on CPU.

4.3.2 Current State of Technology

Neuromorphic computing is in it's early stages of development but significant reseach has already been done. There are physical implementations of neuromorphic processing cores such as the IBM TrueNorth and Intel Loihi.

The advantages of a neuromorphic neural network is as follows:

- Development of more efficient and powerful networks.
- Could lead to more accurate outputs from the networks.
- Extremely power efficient a compared to neural processing units.

The shortcomings of neuromorphic neural networks are as follows:

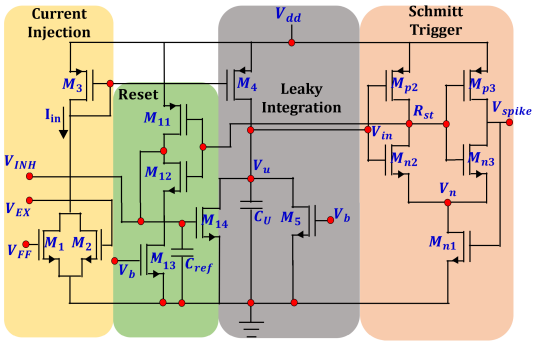
- As machines can learn from biased data, there is a chance that the learning may be improper.
- Ethical and legal implications of a machine that can learn and adapt on it's own.
- The technology is at it's infancy and would require significant research before it can be widely accepted.

4.3.3 Implementation

This section describes the proposed implementation of the neuromorphic compute processor cluster and then moves on to describe the architecture for a single layer of the neural network that can be replicated to implement the while network according to the task requirements.

Neuron

The neuron is one of the basic parts in a neural network that is responsible for the transmission of data between other neurons to carry forward the processed data. We propose the use of a Leaky Integrate and Fire (LIF) neuron model for this purpose. This is used to implement the accumulation of signals and then a non-linear activation function on the sum based on the firing threshold of the neuron.

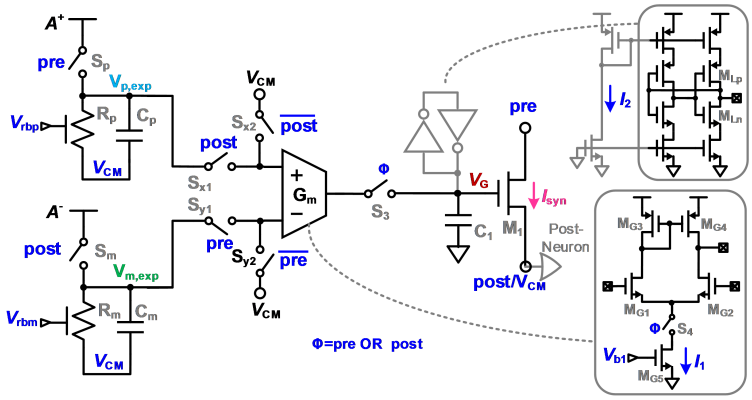


The sub-parts of this model are described as follows along with their working:

- **Current Injection**: This is responsible for the introduction of current into the circuit which is then processed upon by the subsequent subparts. The input is received as spikes from the previous neuron through a synaptic connection (T_ff), excitatory (T_Ex) and inhibitory T_Inh from lateral neurons. The excitatory and inhibitory spikes are required for the realization of a Winner Take All (WTA) configuration.
- **Leaky Integration**: This is responsible for the accumulation of all the input spikes and a leak of the membrane potential to mimic the behaviour of a biological neuron. The accumulation is done in a capacitor that gets charged and is allowed to discharge as per the amount of leak current that is to be modelled. The output goes to the Schmitt trigger circuit.
- **Schmitt Trigger**: This circuit is responsible for issuing the spike when the accumulator output reaches a threshold voltage. The output is exposed as the output of the neuron and is also fed to the reset circuit.
- **Reset**: This circuit is responsible for draining the accumulator capacitor after the issuance of a spike by the Schmitt trigger.

Spike Timing Dependant Plasticity

STDP is a learning algorithm used for training SNN models. It is a Hebbian-like learning rule which updates the weights between two neurons based on the timing difference between their triggering. If the presynaptic neuron fires earlier (later) than the postsynaptic neuron, the weight between the neurons increases (decreases).



The sub-parts and their workings are explained as follows:

- **Exponential Decay**: This is implemented as an RC circuit. It's used to incorporate the time difference between the spikes as the voltage level that is put on the line.
- **Gain**: This is an differential amplifier to increase the spike voltage level after the exponential decay. The amplified signal is passed to the synapse for training. The amplification of the pre-synaptic spike is positive while amplification of the post-synaptic spike is negative to incorporate Long Term Potentiation (LTP) and Long Term Depression (LTD).
- **Switch**: This is used to connect the amplifier to the input of the synapse but only at certain conditions. The switch for the pre-synaptic spike is controlled by the post-synaptic spike and vice-versa. This along with the exponential decay and the differential gains of the pre and post spikes implements the STDP learning rule.

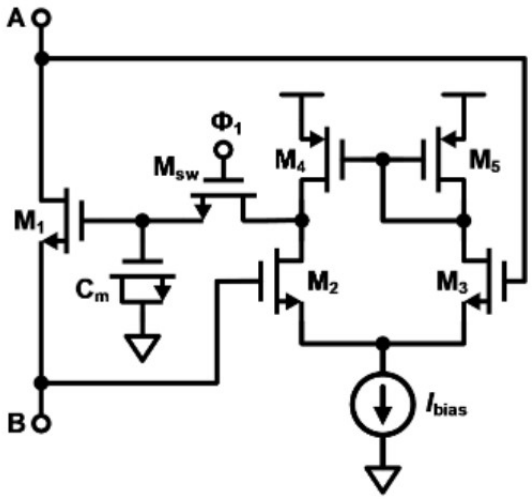
The pre-spike arrives at the STDP module and is passed through to the synapse. The output line corresponding to the particular STDP is then fedback to the STDP as the post-spike. Supposing that this spike comes after a time interval of Δt. At this point the switch for the pre-spike is closed which puts the amplified attenuated pre-spike on the output line to the synapse to train the memristive elements. The same happens when the post-spike arrives before the pre-spike but in that case the amplification is negative. Thus in the case when the pre-spike arrives first the update to the weights is positive while it is negative when the post-spike arrives before the pre-spike. Since the post-spike is the combination of many neuronal inputs to the neuron after the synapse, both the cases are likely depending on the network architecture and the input pattern.

Memristor Emulator

Memristive elements are central to the functioning of a neuromorphic neural network as it acts as the memory unit which also participates in the computation thus giving the processing-in-memory aspect. These are circuit elements with peculiar properties and characteristics that make them retain values that can be changed. A memristor has the ability to change it's conductance based on the voltage that is applied across it's terminals.

Since the manufacturing processes for memristive materials is not very mature and their control is also difficult, we propose the use of a CMOS based emulator circuit to mimic the characteristics. The memristive element used here is a capacitor attached to a latch which gives the output to the gate of an nMOS.

Since the control of digital systems is easier as compared to analog systems, we make use of an array of bistable memristor emulator of size 16. The 16 wide array by itself can model a 4-bit weight since all the inputs are identical so just the combination of the activated memristors affects the product. To improve upon this we make the inputs such that the order of the memristors makes a difference. To do this we add differential gains to the lines. Thus the memristor array can model a 16-bit weight.



The weight is modelled by the resultant conductance of the line. The output is the current through the line corresponding to the voltage that was applied. The individual memristor emulator circuits stabilise to either if their states at random during initialization. The resultant conductance of an array can be calculated as follows:

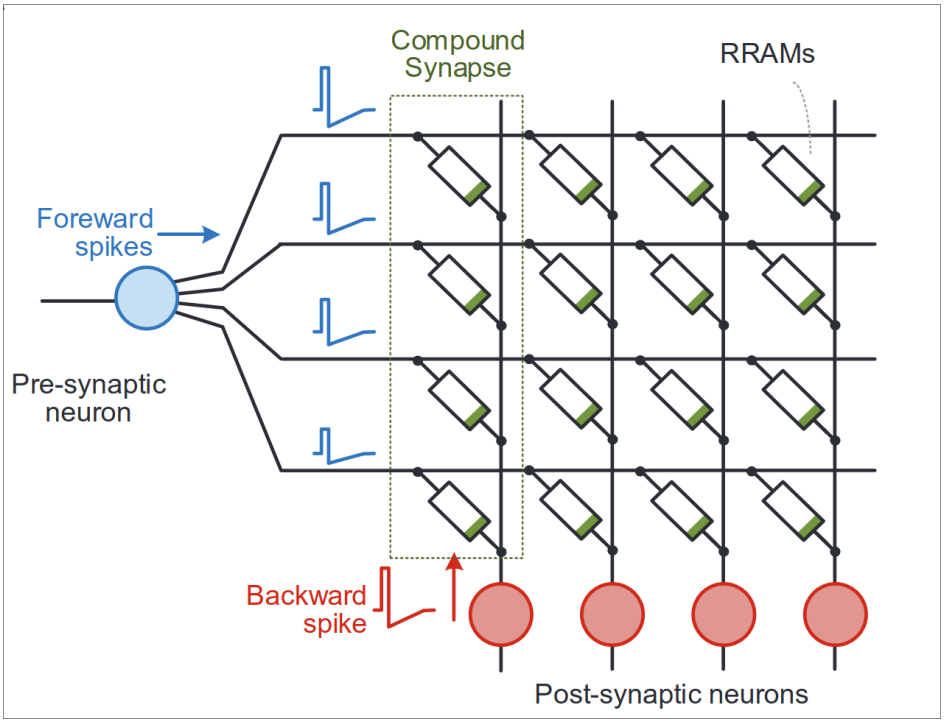
$$W_{net} = \sum_{i=0}^{16} [G_{on} * a(i) + G_{off} * (1 - a(i))]\alpha_i \tag{1}$$

Where a(i) is the status of the memristor emulator depicting if it is active or not as the values 1 and 0 respectively. G_{on} and G_{off} are the on and off conductances of the memristor emulators and α_i are the differential gains of the lines.

Synapse

Synapse is the connection between a set of input neurons and a set of output neurons. It is implemented as a crossbar arrangement of the combination of a STDP unit and a memristor emulator together between each interconnection of the input and output neurons. Thus for an implementation containing 9 inputs and 9 outputs, we need a synapse of size 81.

Each input neuron passes through an STDP unit and a memristor emulator to connect to every single output neuron, the wire connecting the the interconnections to the output neuron implements a reductive or function to sum all the inputs. Thus each neuron input is a Multiply and Accumulate (MAC) operation. Finally the output neuron implements the activation function in the form of the threshold activation.



Layer

A layer of the neural network comprises of the input neurons, the synaptic crossbar arrangement of both STDP and memristor emulator units and the output neurons. A single layer of a neural network performs some operations on the inputs to get the transformed outputs. These outputs are then worked upon by the subsequent layers to get to the required input. We implement programmable switches so that we can have the whole network either in inference mode, where the architecture performs inference and the STDP units are cutoff from the network, or in the training mode, where the networks weights are manipulated by the STDP units.

Coreset

Coreset is an implementation technique proposed for mapping neural networks to neuromorphic processors. This is based on evolutionary algorithms to find the best network to hardware mapping to get optimised performance as well as to take care of the fan-ins and fan-outs of different components. The system is divided hierarchically as follows:

- **Neural Network Layers**: A set of layers are implemented together to accommodate layers smaller than a defined maximum size.
- **Core**: A number of the above mentioned layers are combined to form a core. The layers inside a core can be connected to each other laterally to incorporate layers larger than what a single layer can accommodate.
- **Coreset**: A number of cores are combined together to form coresets which allow for a more coarse grained control over the mapping of the layers.

4.4 Design Space Exploration

Design Space Exploration (DSE) refers to the systematic analysis and pruning of the unwanted design points based on parameters of interest. Given the parameters that can be changed, a design space is generated. This is then mapped to cost function based on the performance of the system that is defined by the respective design point. The design parameters that can be changed are mapped to a specific IP, thus restricting the way these can be changed. To accommodate this, the first step of the DSE run is to generate the available set of IPs and populating a database with parameters of concern for the respective IP and how it behaves in the system. The next step is to generate and evaluate the performance of candidate designs. In the first iteration a very basic design is initialised. The evaluation is based on some cost metric associated to the designs and a genetic algorithm is used to generate design points for next generation. This way the algorithm

moves to a better pool of design points which are better performing with respect to the cost function that was used and satisfies the constraints that were set.

5 Chiplet-Based Power Electronics

Power converters are essential components in electric vehicles (EVs), as they manage the flow of electrical energy between the battery, motor, and other subsystems. They play a crucial role in ensuring efficient and reliable operation of the EV powertrain.

5.1 Current Architecture

There are mainly four types of power converter present in an EV:

- **On-Board Charger (AC-DC Converter)**: This converter converts AC from the grid or charging station to DC for charging the battery.
- **Motor Driver (3-Phase Inverter)**: This converter receives DC power from the battery and converts it to three-phase AC power to drive the electric motor. The inverter controls the speed and torque of the motor, hence enabling smooth and efficient acceleration and braking. Pulse Width Modulation (PWM) techniques are employed to control the inverter and achieve desired motor performance
- **Auxiliary Power Supply (Unidirectional DC-DC Converter)**: This converter steps down the high-voltage DC bus voltage to a lower voltage level required for various auxiliary systems in the EV. It supplies power to components like headlights, infotainment systems, and other electronic devices. Different topologies like buck converters are commonly used for this purpose.
- **Stepping Up Battery Terminal Voltage (Bidirectional DC-DC Converter)**: The primary function of this converter is to boost the battery voltage to the desired HV DC Bus Voltage. Its bidirectional nature aids in Regenerative Braking.

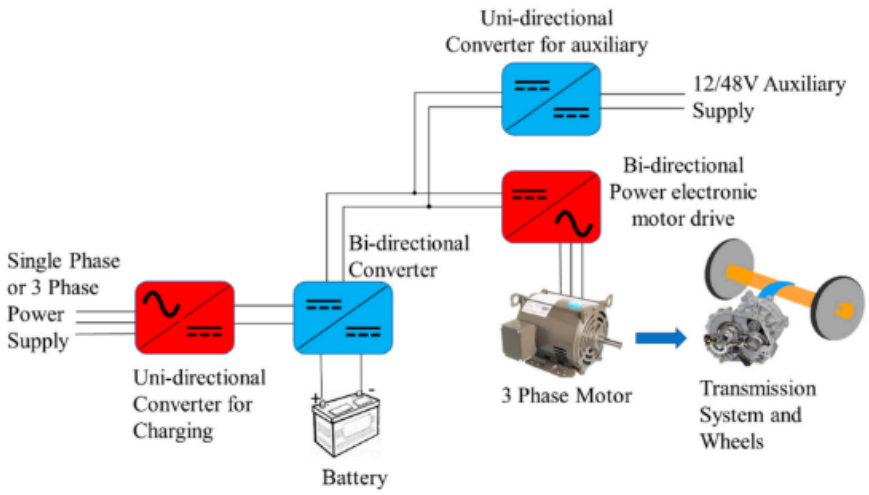


Figure 29: EV Power Converters

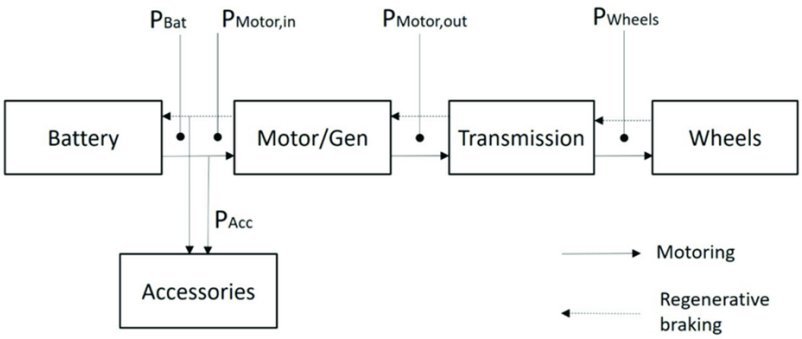


Figure 30: Power Flow Diagram

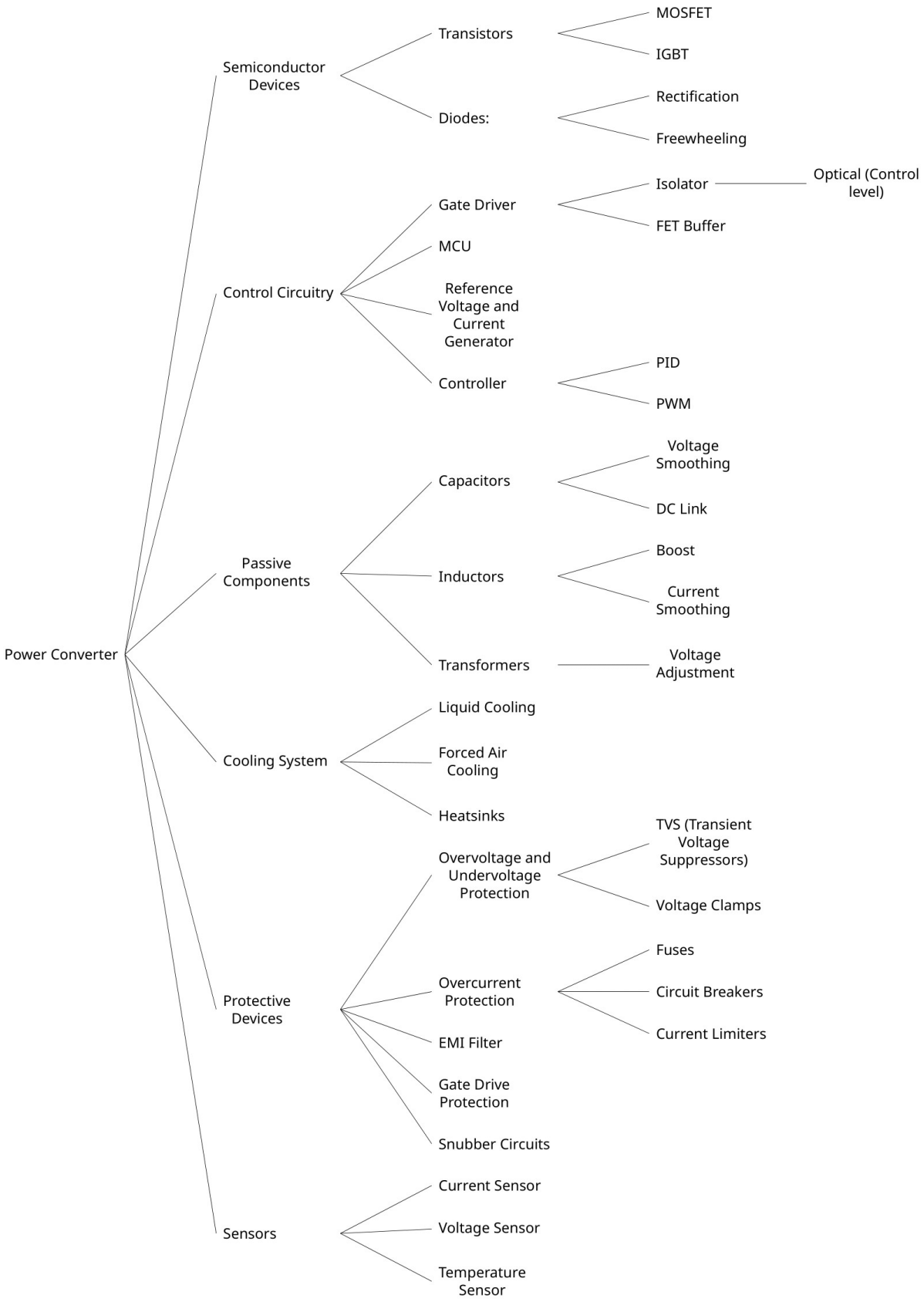


Figure 31: Components of Power Converters

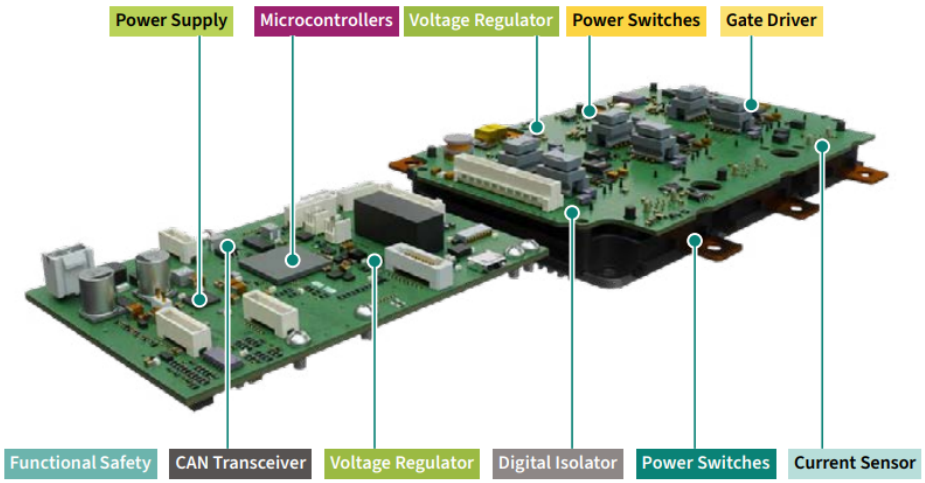


Figure 32: An example : Traction Inverter

The above circuit is the conventional implementation of a Traction Inverter. Circuits like the Micro-controller, Gate Driver, and Power Switches are discrete components on PCB.

Limitations

- **Efficiency**: Silicon-based power devices, such as IGBTs and Si MOSFETs, have relatively high switching losses, especially in high-power applications. These losses can reduce the overall efficiency of power converters in EVs. Enhancing efficiency can lead to longer driving ranges and reduced energy consumption.
- **Thermal Management**: Managing heat generated by power electronics remains a challenge. Silicon devices have high operating temperatures, which may require more advanced and efficient cooling systems, adding complexity and cost to the power electronics. Heat can also reduce the efficiency and lifespan of components. Developing more effective and compact cooling solutions is a priority.
- **Reliability Issue**: EVs rely on the reliability of power electronics for safe and continuous operation. Ensuring that power electronic components and systems are robust and long-lasting is crucial as failures in power electronics can be costly and potentially dangerous. In today's scenario we have limited option to track the health and aging for power converter switches (especially in inverter).
- **Size Factor**: As most modern EV power electronics are implemented on a traditional PCB with individual components rather than on a chip, the overall circuit takes up a significant amount of volume and weight of the vehicle. This can limit power density and increase the weight of power electronic components in an EV, which is a critical concern for electric vehicle designers.
  Moreover, critical connections like gate driver and MOSFET gate terminal should be uniform, failing which may lead to asynchronous switching degrading the performance of the converter.

5.2 Proposal

Chiplets provide significant advantages to power converters by permitting the integration of wide bandgap (WBG) power devices with silicon control logic, memory devices, and with evolving passive devices.

To reduce PE circuit size and path impedance and achieve higher power processing densities, components need to be smaller and integrated closer together. WBG-based semiconductors have higher operating frequencies, which help reduce component size and distance between them, which in turn reduces path impedance and radiated and coupled electrical noise.

The main challenge is to significantly reduce the interconnect resistance and parasitics between separately manufactured components (e.g., power transistors, gate drivers, controllers, capacitors and inductors).

The silicon driver die and WBG FET Die are on a single package, which helps equalise the length between gate drivers and gate of switch and at the same time reduces it. Gate driver losses are reduced significantly. As the technology advances Driver Circuit can also be made from WBG Semiconductor, making the system

more robust. By using WBG, computational speed also increases significantly. Due to well-proven processes, Si Semiconductor still has the highest market in this domain.

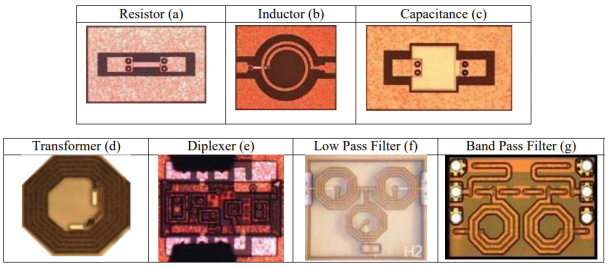


Figure 33: Integration of Passive Components

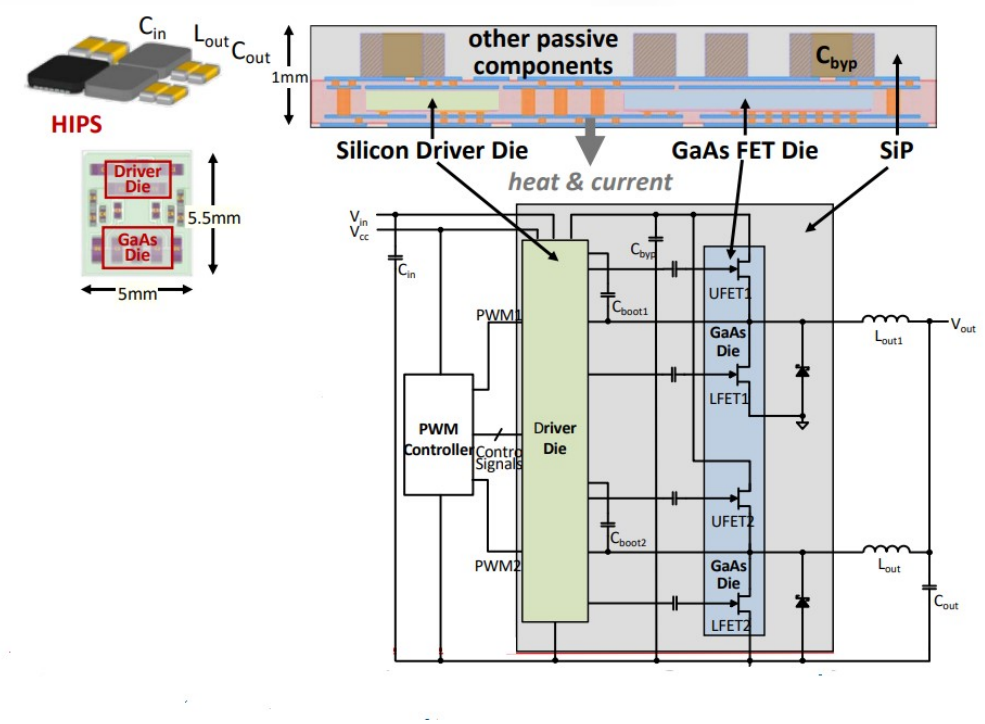


Figure 34: Heterogeneously Integrated Power Stage

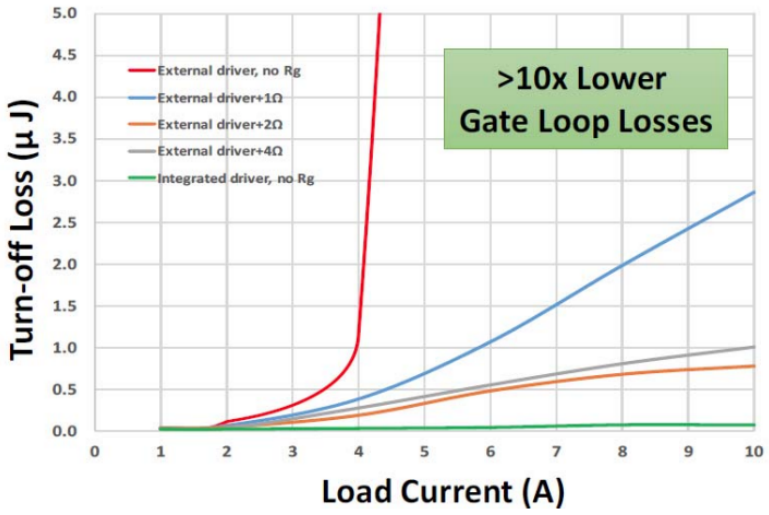
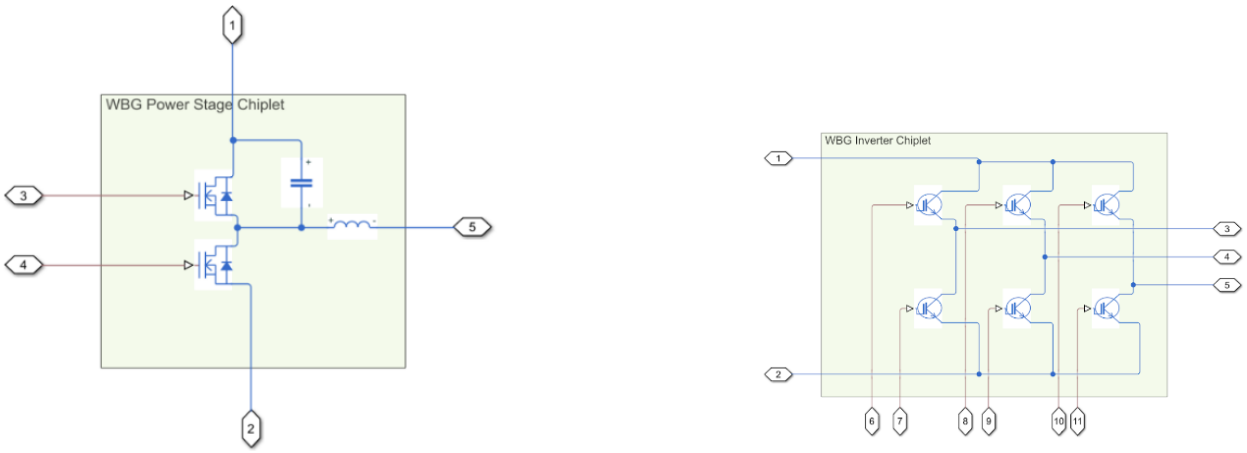


Figure 35: Gate Driver losses at different gate resistance



(a) Buck Converter Power Stage Chiplet

(b) Inverter Power Stage Chiplet

Figure 36: Example

The individual discrete external switches that would constitute the power stage of a conventional power converter are replaced with WBG-based chiplets. Each power stage chiplet would have multiple MOSFETS/switches arranged in the required arrangement according to the type of converter. This can directly be interfaced with the gate driver component of the control circuit.For current ratings higher than what a single WBG based chiplet would be able to provide, multiple chiplets, connected via interleaving, can be used. Depending on the manufacturer, any output filter/inductor may or may not be integrated into the chiplet.

Similarly, others part of the system can be encapsulated into chiplet based architecture as shown below:

5.2.1 Advantages

- **High Efficiency and Power Density**:
  - Due to chiplet-based implementation control as well as power level components come close thereby reducing the path impedance, parasitics and hence the losses. It also leads to **efficient packaging**
  - For decades, the primary material used for semiconductors has been Silicon, which has a bandgap of 1.2eV. Wide band materials like SiC and GaN require a larger energy (approximately 3.2 ev) applied to conduct. This means a wide band gap (WBG) material can be thinner than Silicon to sustain the same voltage applied. Thinner WBG material results in lower conduction and switching losses compared to silicon of equal voltage rating.The reduced switching losses allow for higher switching speeds, resulting in smaller, more efficient power supplies. Reduced losses also mean the WBG material does not heat up as much as silicon. Additionally, **size and weight of the converter are also reduced**

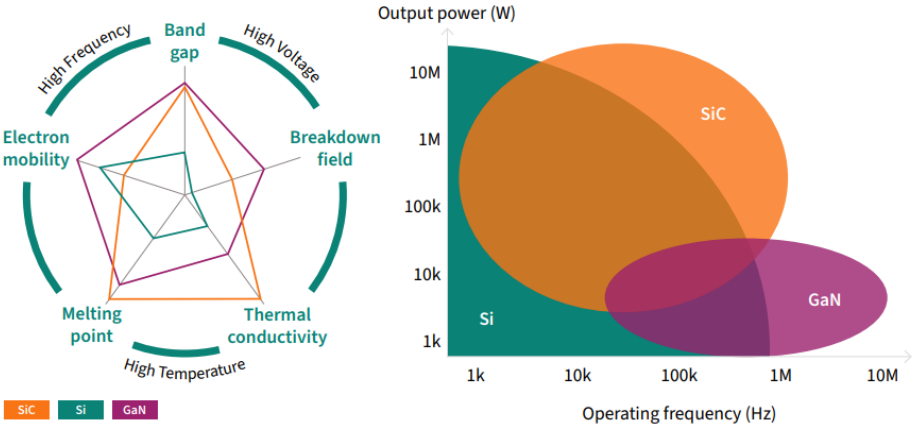


Figure 37: Comparison of Si with other WBGs

- **Fast Time Response**: Wide bandgap materials tend to possess higher electron mobility and electron saturation velocity, allowing for switching frequencies up to 10 times higher than silicon. Chiplet-based architecture adds to this advantage due to inherent minitiarisation.
- **Increase in customisability**: Due to modularity of design OEMs can make their custom power converters.
- **Reduced EMI**: Chiplets Reduce electromagnetic interference (EMI) by minimizing the need for external wiring and providing better shielding. This can improve signal integrity, reduce interference with other electronic devices, and ensure compliance with EMI regulations.

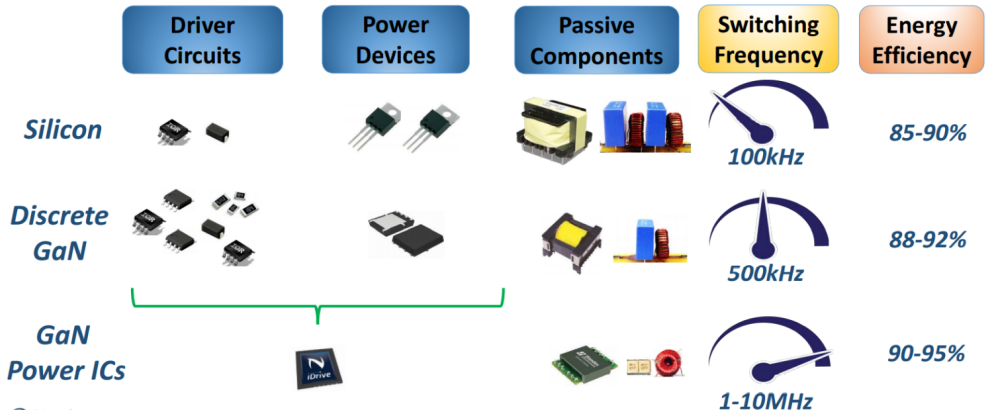


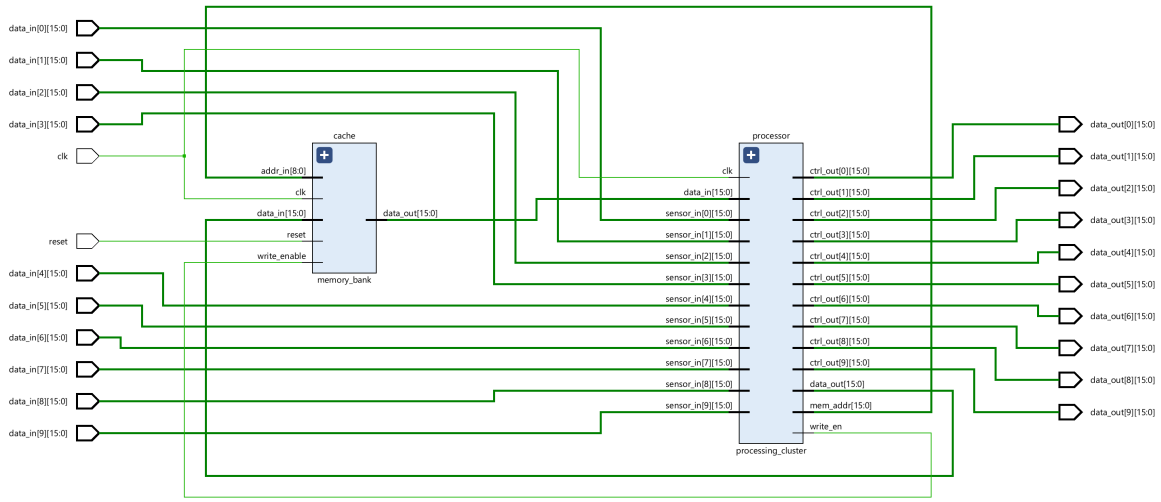
Figure 38: Si vs GaN vs Integrated GaN

6 Micro-Architecture Diagrams

The microarchitectures for each sub-unit was designed using SystemVerilog and was elaborated using Vivado. The units are as follows.

6.1 Application 1 - Integrated Master Control Unit

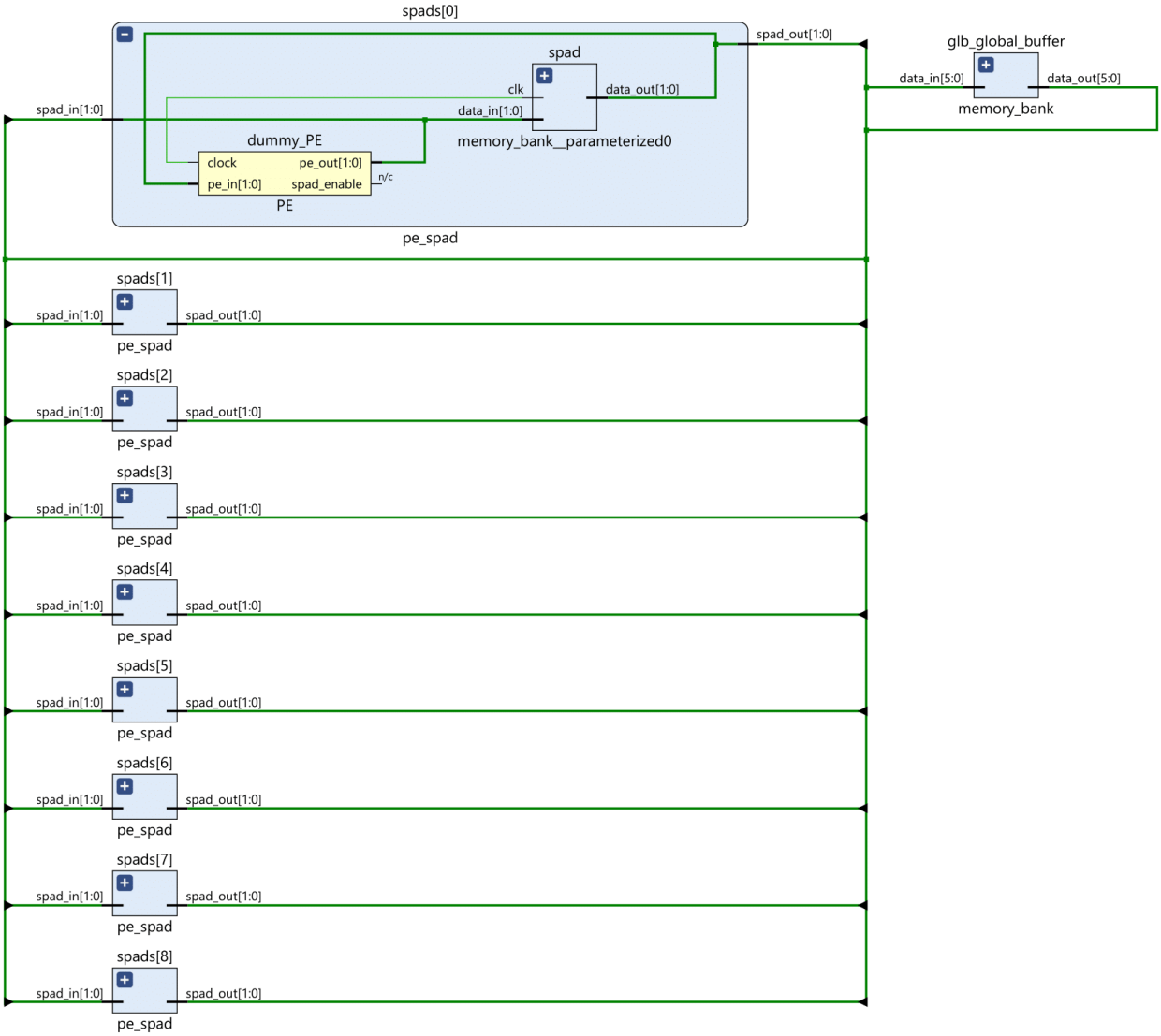
6.1.1 Processing Cluster



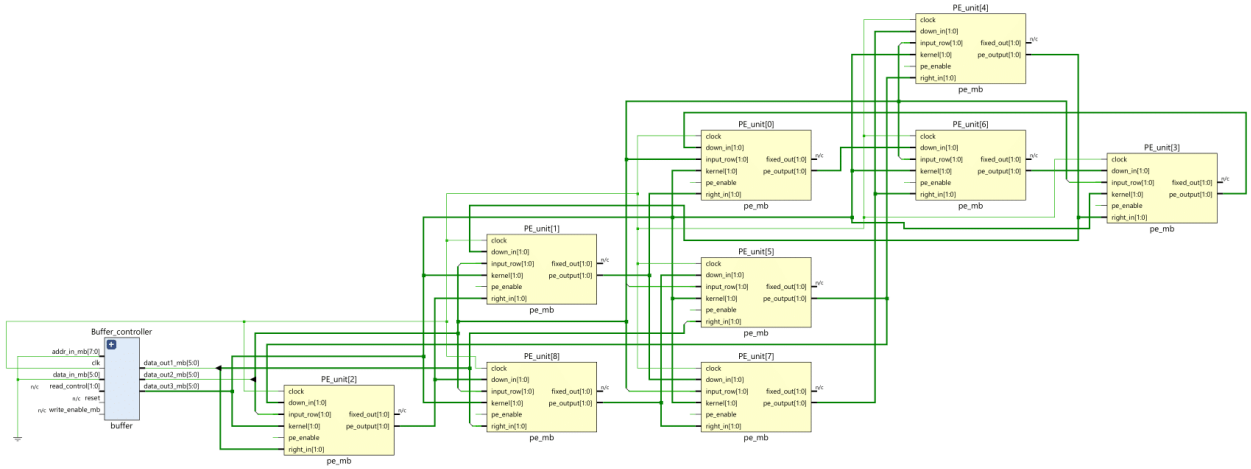
6.2 Application 2 - Autonomous Driving System

6.2.1 Heterogeneous Dataflow Accelerator

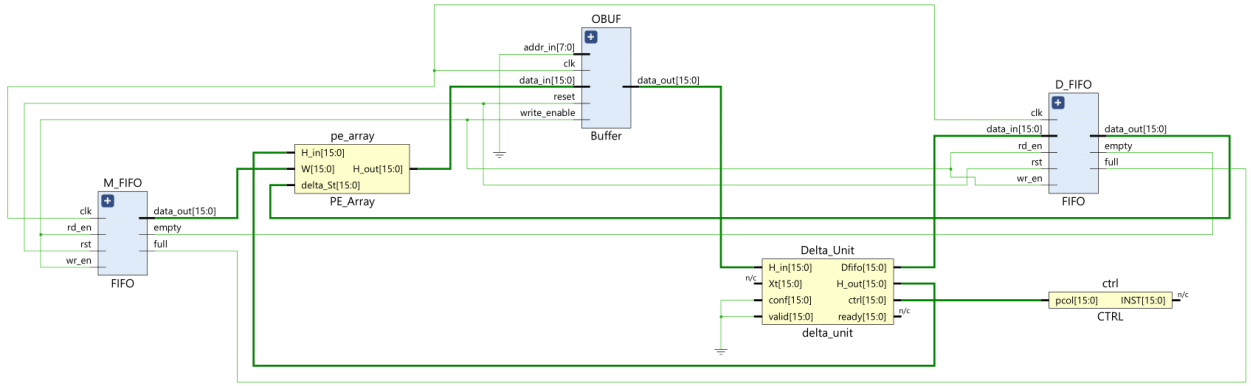
Eyeriss



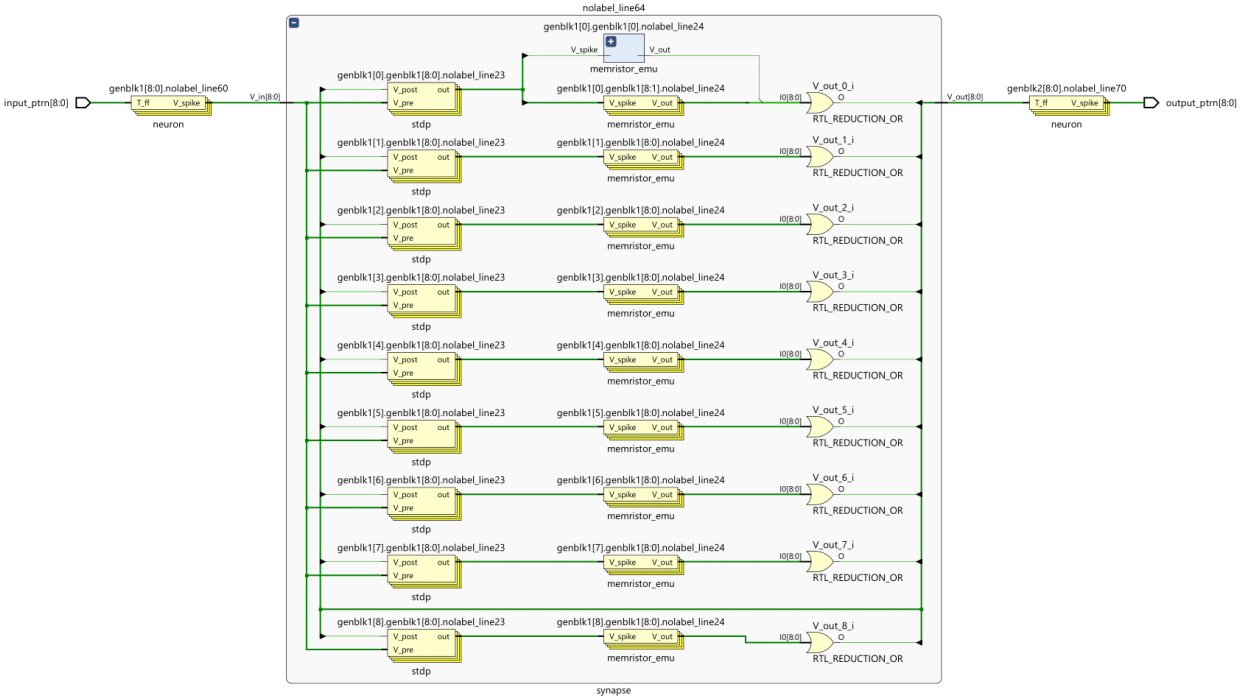
ShiDianNao



EdgeDRNN



6.2.2 Neuromorphic



7 Make Versus Buy Decisions

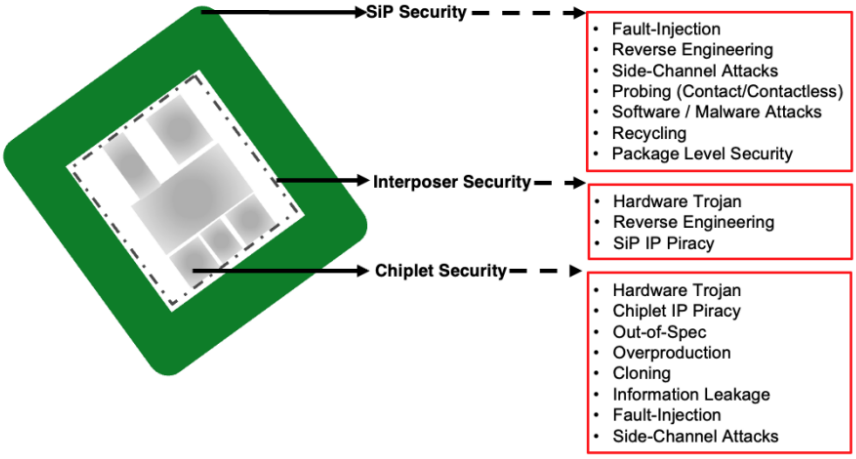
8 Interconnects & Communication Technology

8.1 Performance Metrics

8.2 Security

Integrating multiple chiplets into a heterogeneous package opens the door for security breaches and potential risks associated with malicious modifications or attacks on the individual chiplets during design, assembly, or testing. The second problem is a wider attack surface because the connections between the chiplets are a lot easier to track. This creates a lot of scope for external attacks. The 2.5D and 3D packaging methods provide both promises and challenges for advancing security. The security threats can be broadly classified into two categories:

- **Confidentiality and Integrity of Hardware** - With chiplets originating from diverse sources, there is a risk of unauthorized modifications or the insertion of malicious components. Attackers could exploit design or manufacturing flaws to inject hardware Trojans, which could alter chip functionality or leak sensitive information. This vulnerability is further amplified by the complex supply chain and multiple interfaces in heterogeneous integration. Tracking and verifying the authenticity of each chiplet becomes difficult, and the interconnected nature of these packages increases the attack surface, exposing critical data paths to potential manipulation.
- **Data Security at Runtime** - As chiplets communicate and share sensitive data within the package, protecting it from unauthorized access and manipulation becomes crucial. Side-channel attacks can exploit variations in power consumption or electromagnetic emissions to extract confidential information. Additionally, malicious software running on one chiplet could potentially access or alter data stored or processed on others. The shared interconnects and memory spaces in heterogeneous integration create vulnerabilities, making data leakage and unauthorized access easier. Furthermore, the complex communication protocols and software stacks involved introduce new attack vectors.



8.2.1 Vulnerabilities

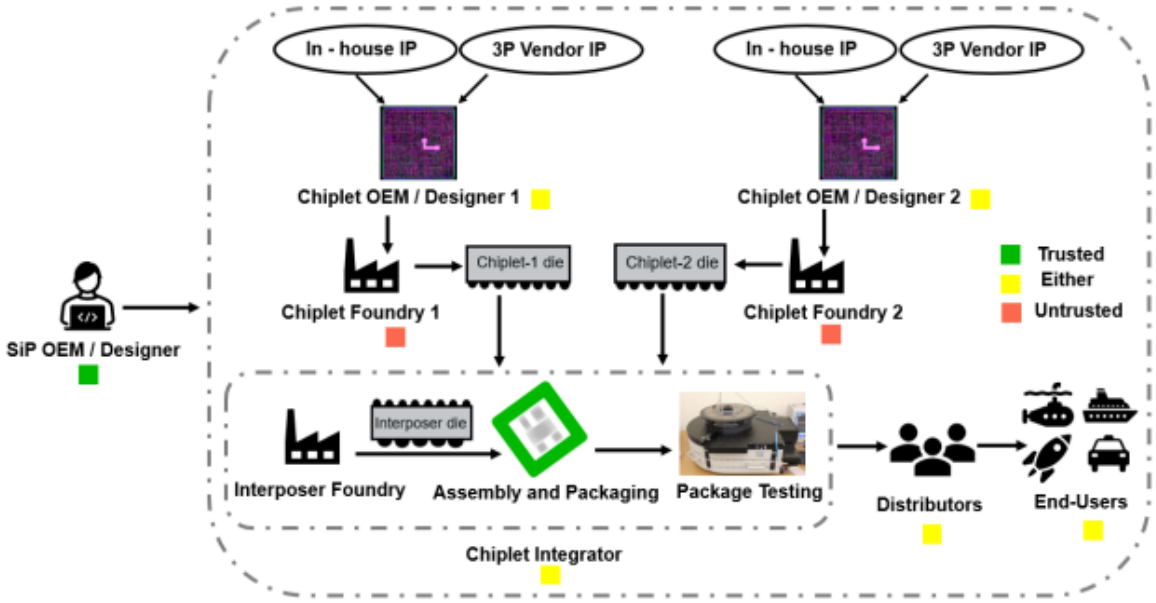
These security vulnerabilities are caused mainly due to the following reasons:

- **Supply Chain Threats** - The fabless design house employs a horizontal business model, outsourcing the fabrication, assembly, and testing of integrated circuits (ICs) to offshore foundries and OSATs to enhance cost efficiency and expedite time-to-market. This approach extends to chiplet design, where fabless design houses are anticipated to adhere to the same model. The threat model highlights the supply entities associated with chiplet security, emphasizing the trusted role of the SiP designer, also known as the original equipment manufacturer (OEM), responsible for SiP security. Chiplet design houses may utilize in-house and third-party intellectual properties (IPs) to craft a chiplet design, subsequently sending it to an offshore foundry for fabrication. The trustworthiness of a chiplet design house depends on its geographical location or market reputation, as it may be deemed either trusted or untrusted. Risks include hardware Trojans, overproduction, IP theft through reverse engineering, and the cloning of chiplets by rival foundries. Recycled chiplets extracted by untrusted distributors or end users pose a potential threat, though the effort required for extraction may mitigate perceived risks. In this threat model, the SiP designer/OEM is regarded as trusted, while all other entities in the supply chain are considered untrusted, with the design tools used in chiplet IP generation being treated as trusted elements.

- **Hardware Attack Vectors** - Hardware Trojans represent malicious alterations in chiplets designed to compromise the security of the System-in-Package (SiP) or the electronic system in which the SiP will operate. The presence of these Trojans raises serious concerns about the confidentiality, integrity, and availability (CIA triad) of the SiP, potentially leading to denial-of-service attacks for reliability issues, unauthorized access through Man-in-the-Middle attacks to sensitive data like encryption keys, and the manipulation of neural network biases.

  Concerns persist regarding reverse engineering (RE) in the context of chiplets. This process involves extracting the Register-Transfer Level (RTL) design by deprocessing and imaging various layers of fabricated integrated circuits. Competitor semiconductor design houses or adversarial foundries may employ RE to gain a competitive and financial advantage. However, this practice can lead to revenue losses for chiplet Original Equipment Manufacturers (OEMs), and the reverse-engineered chiplet may encounter reliability and trust issues.

  A counterfeit chiplet refers to an unauthorized chiplet that does not meet the OEM's specifications and performance standards, is not produced by authorized contractors, is non-conforming, faulty, or a used OEM product misrepresented as "new," and may have inaccurate markings or documentation. Counterfeiting is a lucrative business for adversaries, generating substantial revenue. However, for the OEM, it results in revenue loss and damage to brand reputation.



In order to tackle these security issues. A number of solutions can be employed to specifically maintain the confidentiality and integrity of hardware and also ensure data security at runtime.

8.2.2 Proposed Solutions

The following points expand on the proposed solutions for the above mentioned issues with heterogeneous integration (HI).

Securing Dataflow

For inter-die communication, consideration must be given for data integrity and confidentiality to protect against probing attacks. For this we propose the use of the following:

- Hash-based message authentication code techniques for signal integrity.
- Authenticated encryption protocols for data transmission for ensuring data confidentiality and authenticity.

Protection Against Physical Attacks

In the case of a physical attack wherein the adversarial attempts to damage the package or a side-channel attack, the system should be able to detect and resist such attacks. For this we propose the following:

- Anti-tamper sensors such as a light sensor or a pressure sensor can be used to detect and notify the security control unit about depackaging of a subsystem.
- Use of active or passive shields can help protect against side-channel attacks using EM signature of the package.

Hierarchical Security

Heterogeneous integration of integrated circuits (IC) and Systems-on-Chip (SoC) from untrusted suppliers exposes the System-in-Package (SiP) to Denial-of-Service (DoS) attacks or unauthorized access to valuable data such as encryption keys or metadata derived from sensor readings.

To counteract this, the system can run security monitoring software on a trusted die in the SiP. The security controller is then able to control the component die's access to power rails as a response to detection of an attack in progress and can initiate safety routines.

Metadata computation for each component can be calculated and checked for by the security controller to check against hardware trojans. This can be done by calculating the clock periods it takes to complete

an algorithm of known length and known inputs. This technique assumes that the metadata generated will be different if there is a hardware trojan in the system.

Protection Against Supply Chain Attacks

Supply chain attacks refers to manipulation of the hardware during the various stages in the supply change with malicious intents. Supply chain attacks can be categorised as follows:

- Interconnect modification
- Material manipulation
- Hardware trojans

We propose the use of blockchain in the supply chain to make it more secure and provide the following benefits:

- **Traceability**: As the number of operations on the devices increases due to the increased complexity of the design, the intermediate products need to be shipped from one fabrication unit to another. A blockchain based supply chain helps in keeping a track of the devices that are being shipped from one fab to another and testing of the units at arrival helps keep a check on tampering and also provides the stage at which tampering was done.
- **Transparency**: Transparency ensures that the modifications done on a device are authorised and can be traced back to the particular fab unit.

These features help protecting the IP against counterfeiting and reverse engineering. Since blockchain records can transparently verify certifications and verify the authenticity of the product, counterfeiting is reduced. By keeping a track of all the individual devices, reverse engineering can also be prevented.

Platform Security

Platform integrity is a key concern with regards to the security of a critical system. In this context, integrity means that the system behaves as intended and the platform refers to the computing system on which the application runs. We prpose the use of a Trusted Platform Module (TPM) as a root of trust for the software and the hardware.https://www.overleaf.com/project/656346c1a8cbe343d43564a1 TPM ccontains a number of Platform Configuration Registers (PCR) that allow the secure storage of security-relevant metrics.

TPM ensures the following features in the platform:

- Only authentic hardware parts are included in the system. Counterfeits are more difficult to be introduce into the system. The presence of hardware trojans will make the system not bootable unless the keys in the TPM are tampered with which is very difficult in itself.
- Encryption keys for stored data can be securely stored in the TPM and can only be accessed by authorised devices.
- Only authorised software can run on the platform.
- The TPM firmware can be updated hence ensuring that the security systm can keep up with the developments in the field of cybersecurity.

8.3 Packaging Solutions

9 Thermal Management and Cooling

Semiconductor devices exhibit inherent characteristics that result in the generation of heat during their operation. This phenomenon is primarily attributed to the dynamic processes occurring within the semiconductor materials at the atomic level as they facilitate the flow of electric current. Understanding the causes of heat generation in semiconductors is crucial for devising effective thermal management strategies to ensure optimal device performance and longevity.

9.1 Causes of Heat Generation

1. **Joule Heating**: The most significant contributor to heat generation in semiconductors is Joule heating, also known as resistive heating. As electric current flows through a semiconductor material, it encounters resistance, leading to the conversion of electrical energy into heat. This process is directly proportional to the square of the current and the resistance of the material.

2. **Switching Losses**: In devices such as transistors, diodes, and integrated circuits, rapid switching between on and off states during operation results in switching losses. These quick transitions generate heat due to the inherent capacitance and inductance of the semiconductor components.

3. **Leakage Currents**: Imperfections in semiconductor materials can lead to leakage currents—undesirable currents flowing through the device even when it's in the off state. These leakage currents generate heat and can compromise the efficiency of the device.

4. **Electron-hole pair recombination**: When an electron and a hole recombine, the energy associated with their motion and charge difference is released. This released energy appears as heat. The heat generated during this recombination process adds to the overall thermal load of the semiconductor device.

9.2 Need for Cooling Systems

Efficient heat elimination is critical for ensuring optimal performance and extending the operational lifespan of chiplet technology within automotive electronics. The unique challenges posed by the integration of chiplets in vehicles require careful consideration of thermal management strategies to mitigate the risks associated with elevated temperatures.

In the context of chiplet technology in automotive electronics, the imperative for effective heat elimination becomes particularly evident in preventing thermal-induced failures. Sustained high temperatures, inherent to the compact and interconnected nature of chiplet configurations, can lead to thermal runaway—a self-reinforcing cycle that poses a significant risk of catastrophic failure. Adequate cooling measures are essential to break this cycle, ensuring the integrity of semiconductor components and preventing potential thermal-related risks specific to chiplet architectures in automotive applications.

Heat generation within chiplet-based semiconductor devices signifies not only a loss of energy but also a potential threat to the reliability and performance of automotive electronics. Eliminating excess heat is crucial for enhancing energy efficiency, reducing power consumption, and minimizing energy wastage specific to chiplet technology. This optimization is vital not only for individual chiplets but also for contributing to the overall energy-efficient design of automotive electronics, aligning with sustainability goals in the automotive industry.

Chiplet-based semiconductor devices, encompassing components from microprocessors to integrated circuits, demand efficient heat elimination to maintain optimal performance within specific temperature ranges. Consistent cooling is essential to prevent thermal throttling, safeguarding chiplet technology from performance degradation and ensuring stability in diverse driving conditions.

Given the diverse applications of chiplet technology in automotive electronics, ranging from advanced driver assistance systems to in-vehicle infotainment, effective heat elimination is fundamental for reliability across varied driving conditions. Robust thermal management strategies enable chiplet components to operate efficiently in environments with fluctuating temperatures, contributing to reliable performance and minimizing the risk of malfunctions due to temperature-related stress specific to chiplet architectures in the automotive context.

9.3 Cooling Techniques

Based on heat transfer effectiveness cooling modes can be classified into four general categories which are:

1. Passive Cooling
    (a) Radiation and free convection
2. Active Cooling

(a) Forced air-cooling
(b) Forced liquid cooling
(c) Liquid Evaporation

9.4 Requirements for Cooling of Chiplet-Based Systems

9.5 Emerging Cooling Techniques

References