ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

Homogeneous and Heterogeneous Multicore Systems

Srinath Bheemaraju
Chief Manager- SW
Architecture, ADAS, Continental Autonomous
Mobility, Bangalore, India

Teja Sri Venkat Saidhar Movva
SW Architect, ADAS,
Continental Autonomous Mobility,
Bangalore, India

Priyadharshini S
Engineer, ADAS Continental Autonomous Mobility,
Bangalore, India

Gayathri Nair, Engineer, ADAS
Continental Autonomous
Mobility, Bangalore, India

Chandravardhana Kuluru
SW Architect, ADAS,
Continental Autonomous Mobility, Neu-Ulm, Germany

Abstract:- Multicore systems have gained significant importance in the Automotive industry due to data-intensive applications, such as image processing, high-speed process, and GPS application. It enables manufacturers to build smaller chips, simplifying board architecture and routing, reducing power consumption and cost and increasing programmability. Multicore platforms are segregated into two categories namely Homogeneous (symmetric) and Heterogeneous (asymmetric) multicore systems.

In Homogeneous systems, all cores are the same, including frequencies, cache sizes and functions. In Heterogeneous systems, different cores operate with different frequencies, cache sizes and functions.

In this paper, we are going to discuss various aspects of homogeneous and heterogeneous multicore architectures like number and level of caches, interconnection of the cores, Physical and Temporal isolation, Energy Efficiency, Concurrency, Performance, Reliability and Robustness along with the evaluation of these two architectures for applications based on the above-mentioned aspects.

Keywords:- *Multicore System, ADAS, Automotive, Homogeneous, Heterogeneous, System on Chip, Communication, Interconnect bus, CoreConnect, AMBA, Wishbone, OS scheduling, Energy efficiency, Performance, Isolation, Temporal, Physical, Concurrency.*

I. INTRODUCTION

Advanced Driver Assistance Systems (ADAS) applications are characterized by a growing interest in the real-time and low-power implementation of digital signal processing (DSP) techniques for image processing and RADAR signal processing to improve system performance in terms of processing efficiency. The State of the Art, complex algorithms are proposed to bring highly efficient image processing and higher dynamic real-time control systems realization. To achieve the higher computational power required by ADAS applications, it is required to use multicore systems.

In high-performance computational systems, core architecture has higher importance. A generic core contains a processor, FPU, General purpose registers, Timers, Watchdog, Core specific debug port, Core specific I-Cache and D-Cache, optional L2 RAM and a supporting bus system. These elements' combination defines core performance.

- CPU - Central Processing Unit
- DSP - Digital Signal Processor
- GPU - Graphics Processing Unit

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

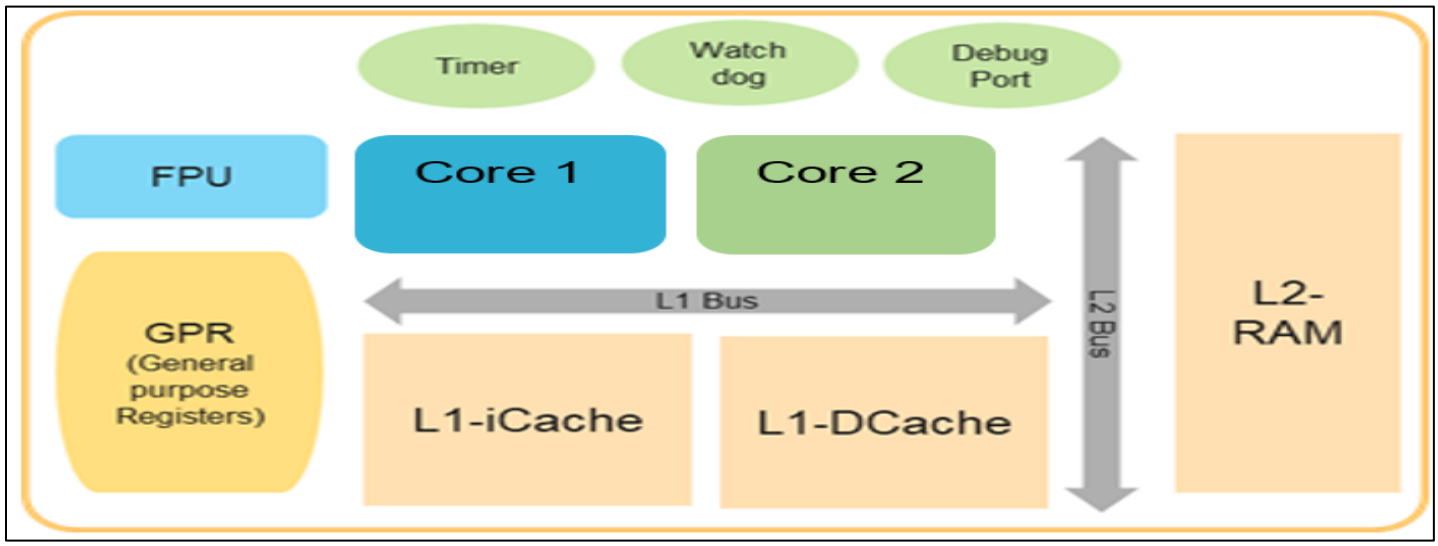

Fig 1: Multicore System

Multicore systems can be categorized as Homogeneous multicore systems and Heterogeneous multicore systems [1][3]

➢ *Homogeneous (Symmetric) Multicore*

Homogeneous cores display homogeneity in their attributes, as they possess uniformity in their ability to execute the same functions and possess an equivalent set of capabilities.

Personal computer processors have homogenous cores, which means that no power is consumed when a task is carried out on any of the processor's cores. A task is expected to be completed at the same time irrespective of which core it is scheduled on.

They can be operated in single OS scheduling per core or common OS scheduling for all cores in the system[5]. A popular example of a homogeneous system is the ARM QUAD-CORE Cortex-A53 system. [17]

➢ *Heterogeneous (Asymmetric) Multicore Systems*

Heterogeneous cores are not identical. They can differ in capabilities, and speed, may lack certain features, or otherwise perform a task differently.

Modern high-end mobile phones tend to have heterogeneous cores. Each core has its own specified functionality, different performance and power dissipation. Heterogenous multicore systems can be operated in single OS scheduling per core. Renesas R-Car-M3Ne uses heterogeneous architecture.

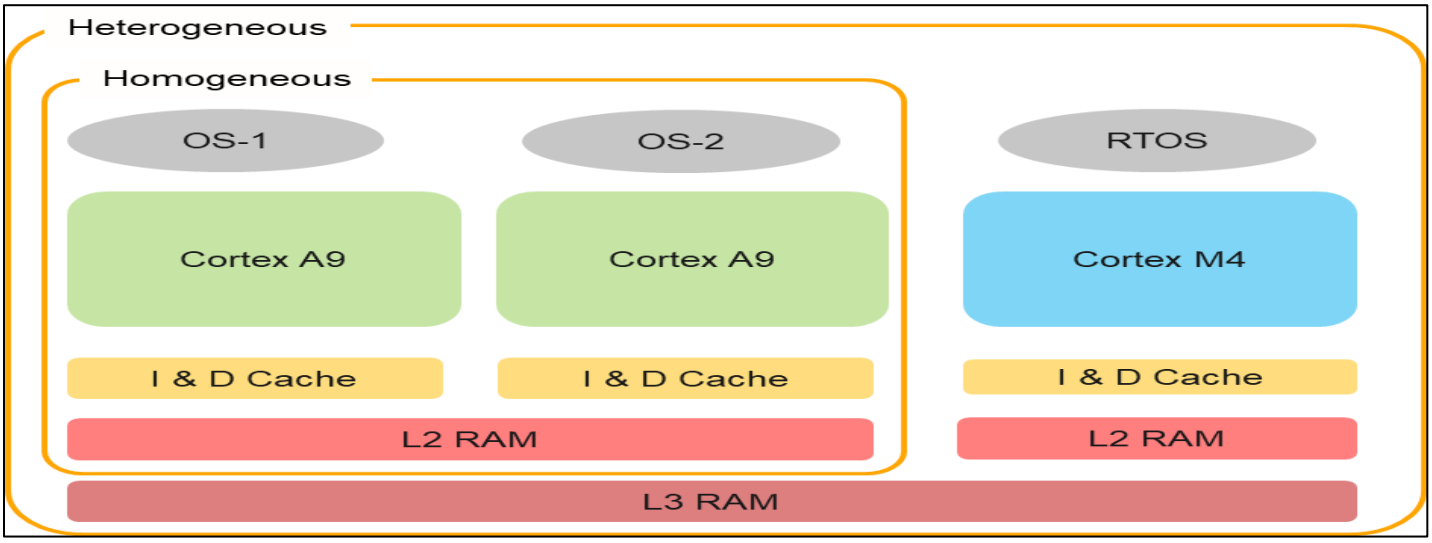

Fig 2: Homogeneous and Heterogeneous Multicore Systems

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

In modern embedded applications, both categories of multi core processors are widely used based on application[16].

II. MULTICORE SYSTEM ARCHITECTURE CONSIDERATIONS

➢ *Various Design and Architecture Aspects for Multicore Systems are Considered in the Subsequent Sections*

- Communication between the cores
- Evaluation of multicore systems
- Physical Isolation
- Temporal Isolation
- Energy efficiency
- Concurrency

III. COMMUNICATION BETWEEN THE CORES

Multicore interconnect plays a significant role in multicore architecture systems. Some of the publicly available bus architectures from leading manufacturers are AMBA, Core Connect and Wishbone [6][18]

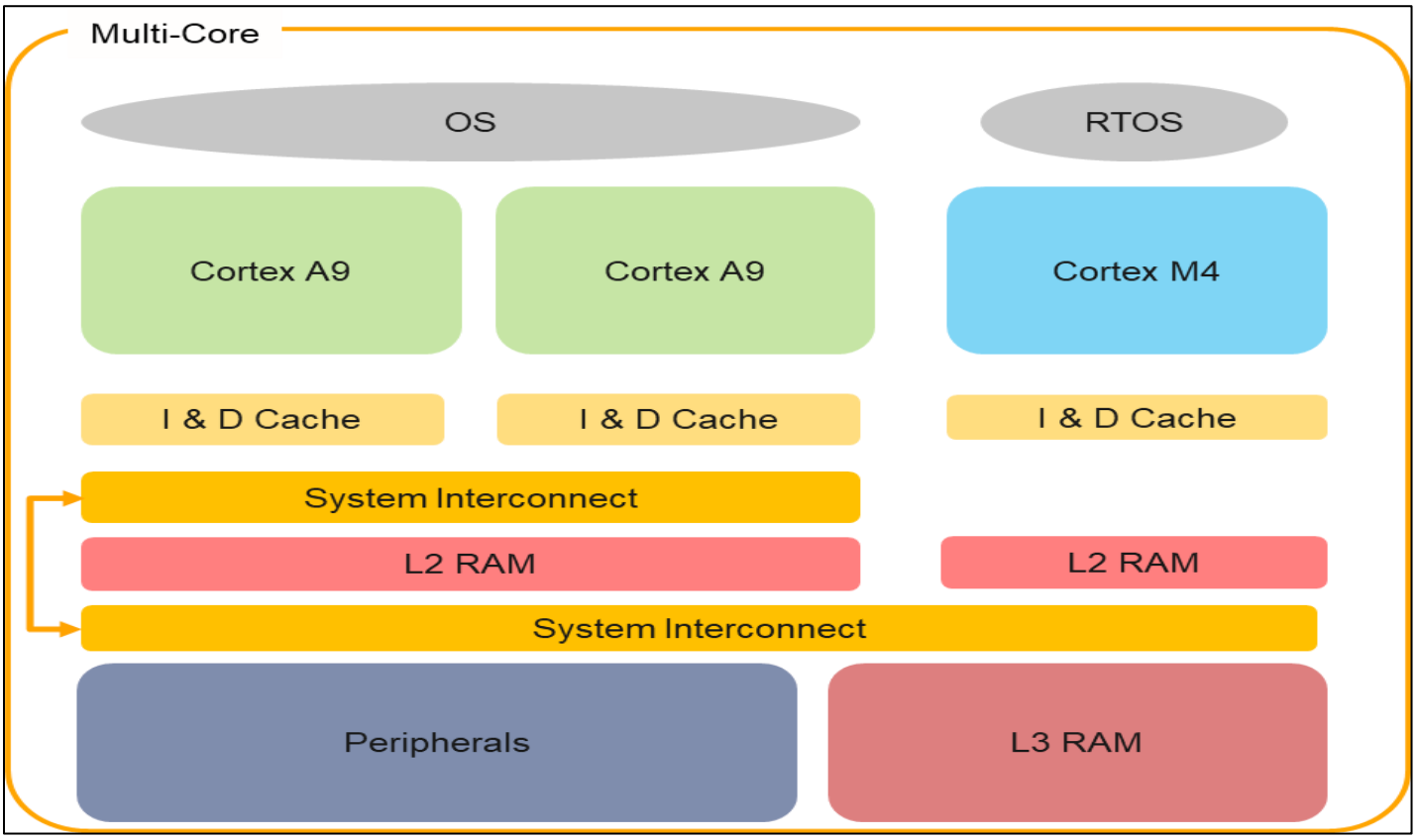

Fig 3: Interconnect in a Multicore System

A. *AMBA[20][6][12]*

Advanced Microcontroller Bus Architecture(AMBA) is a bus standard devised by ARM with the aim to support efficient on-chip communications among ARM processor cores.[20] AMBA is currently one of the top on-chip bus systems used in the design of high-performance multicore computers. This bus architecture helps in reducing the silicon. AMBA is hierarchically divided into the system-bus and peripheral-bus segments, which are linked by a bridge that buffers data and operations between them.[12] The AMBA specifications specify common bus protocols, regardless of processor type, for interconnecting on-chip components in various multicore system structures. Arbitration procedures are not defined by AMBA. Instead, it enables the arbiter to meet the requirements of the application, for interconnecting on-chip components in various multicore system structures, regardless of processor type [6][12] . AMBA is currently in its 5th version.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

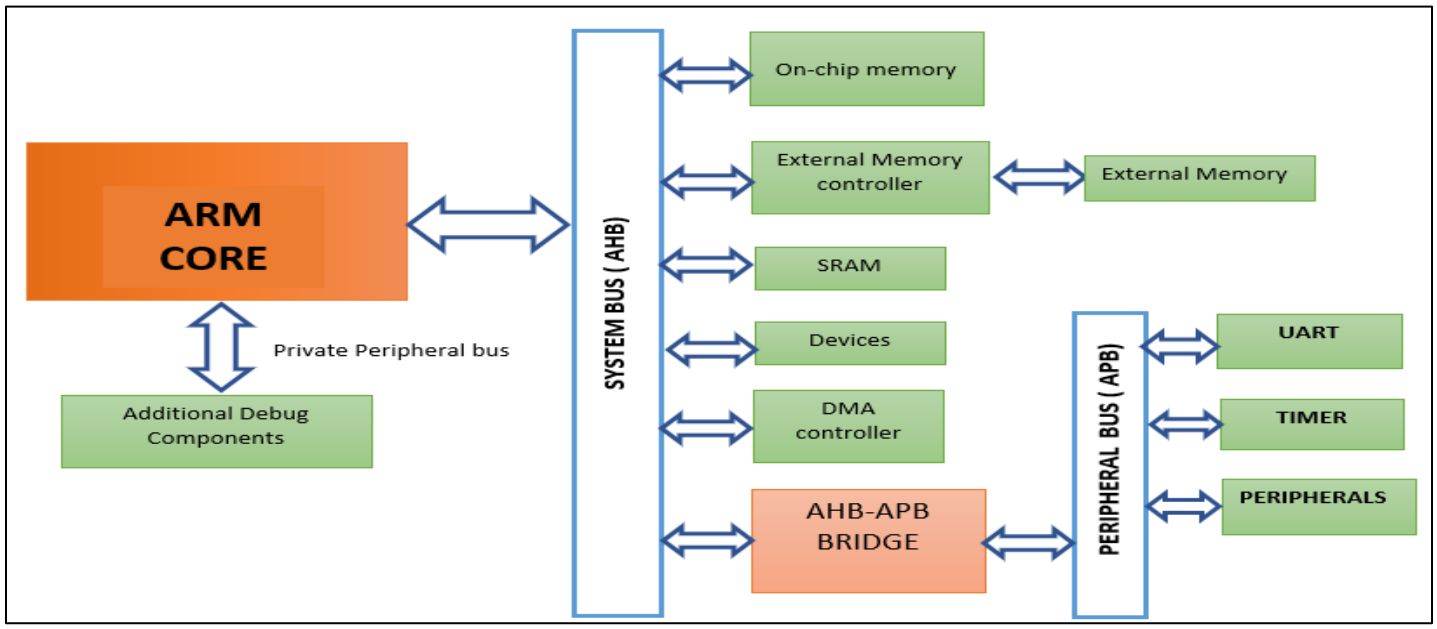

Fig 4: AMBA System Architecture

The private peripheral bus is an APB(Advanced Peripheral Bus) that is used to connect additional on-chip peripherals like debug components. Fig.4[21] shows the AMBA bus architecture devised for ARM processors.

B. Core Connect

Core Connect is an on-chip silicon bus that was developed by IBM for FPGA or ASIC designs[12].

CoreConnect technology is from IBM[12]CoreConnect was developed for FPGA or ASIC designs. It is an on-chip silicon bus and comprises three levels namely the Processor Local Bus (PLB), the On-chip Peripheral Bus (OPB) and the Device Control Register (DCR) bus as shown in Fig. 5.

PLB is the main on-chip bus system and is used to communicate with high-speed peripherals. It is used to connect processors with on-chip memory, memory controllers and other high-speed peripherals, including DMA controllers. It is a synchronous, high-performance, arbitrated bus. It supports concurrent read and writes operations for the same master. A central arbitration mechanism is used to grant access.

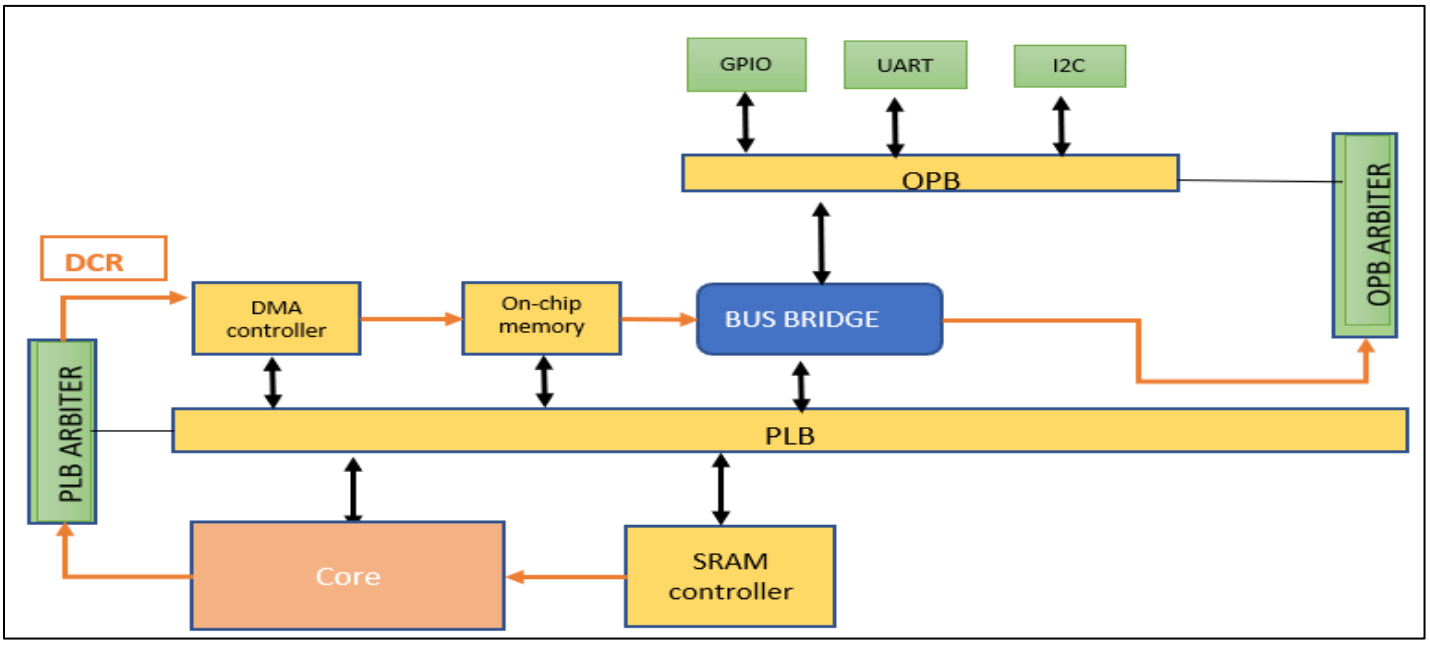

Fig 5: Core Connect System Architecture

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

OPB is designed to support slower peripherals. It is implemented as a straightforward multimaster, arbitrated bus. It is a fully synchronous, 32-bit address bus and 32-bit data bus. Data transactions between master and slave take place in a single clock and burst operations are supported. The masters compete for the bus through arbitration. PLB transactions are converted into OPB transactions by the PLB to OPB Bridge. DCR is used for configuring on-chip device. It offers an alternative path to the system for setting the individual device control registers. A bus arbiter decides which one of the multi-masters will be allowed to control the bus for each bus cycle.

C. Wishbone

It is a portable interface for semiconductor IP cores[13][12]. Design reusability is made possible by establishing a shared, logical interface between IP cores. Therefore, the system is more reliable, portable and the end user experiences a shorter time to market. Wishbone is a standard for building IP cores, not an IP core in and of itself. The handshaking protocol implemented in Wishbone allows each IP core to regulate speed.[14]

➢ *Wishbone Supports Various IP Core Interconnections Including*

- Point-to-point
- Data flow interconnection
- Shared bus interconnection
- Crossbar switch
- Off-chip

Fig 6 shows possible wishbone topologies and how masters and slaves can be connected. Point-to-point is a simple way of connecting master and slave[19]. Data flow interconnection is a connection in which the data flows through a set of IP cores in a sequential order that is prearranged. In shared bus interconnection only one master can use the interconnection at a time, as it initiates the addressable bus cycles for a slave. Crossbar switch architecture helps to operate several channels in parallel. It increases the data transfer rate. In Fig 6 dotted lines denote a possible connection. Similar connections can be made to establish communication channels. Off-chip is used when we have an interface that extends off-chip.[14]

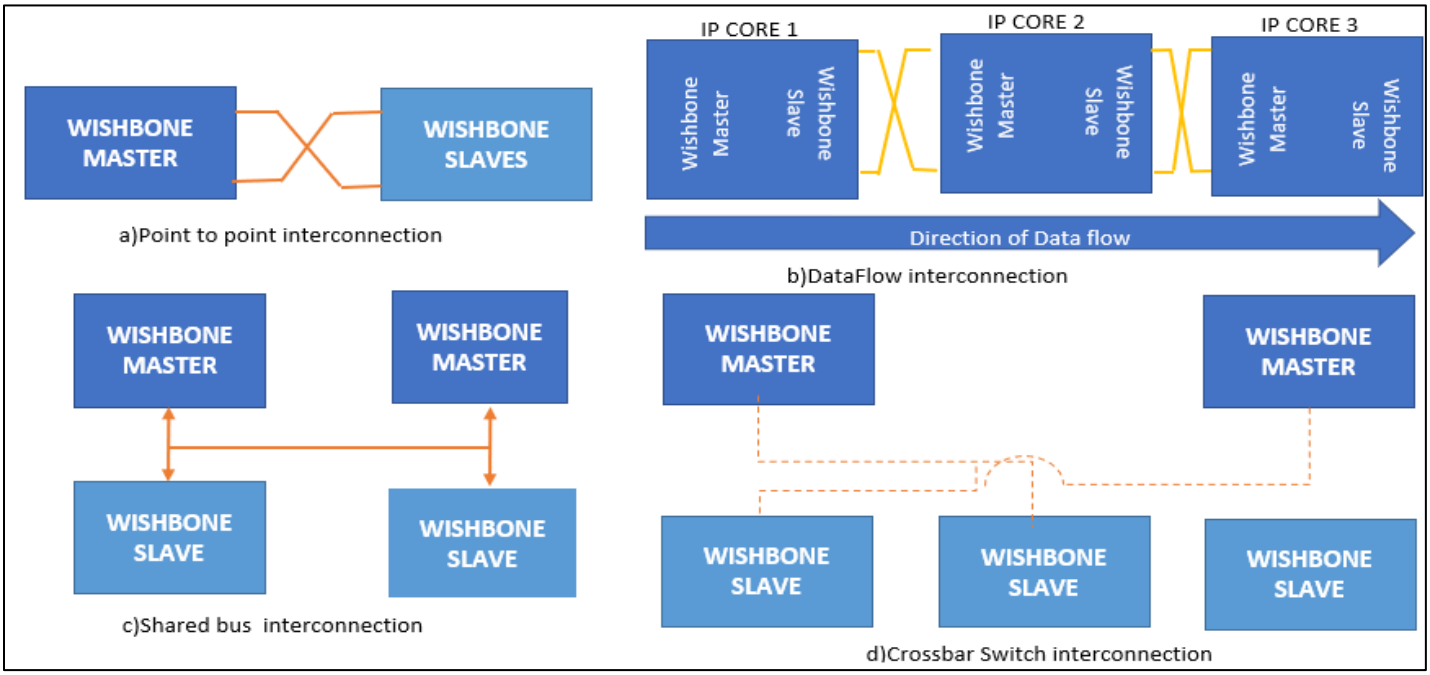

Fig 6: Wishbone Topologies

IV. DESIGN FACTOR FOR MULTICORE SYSTEM

The following sections explain the criteria for evaluating multicore systems in design and performance.

A. Isolation

Isolation physically and logically separates the resources and software programs between the multiple cores. [7][11].

➢ *Physical Isolation*

Physical isolation ensures that various cores in a single chip cannot access the same physical hardware (e.g., memory locations such as caches and RAM).

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

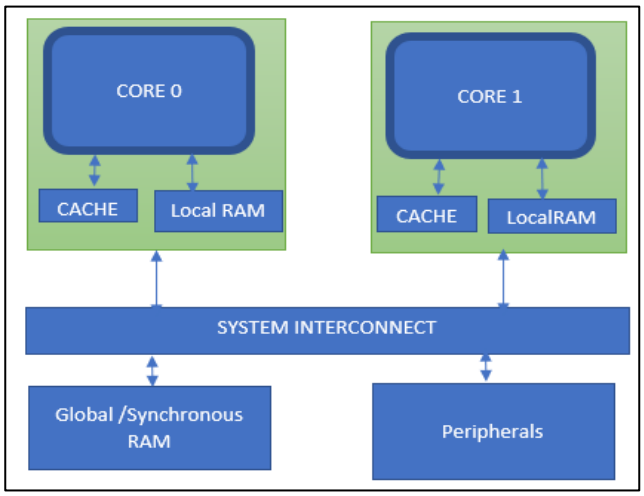

Fig 7: Physical Isolation

Physical isolation involves physically separating the resources, such as memory and I/O devices, between different cores. Each core has its own dedicated resources and there is no sharing between cores. This approach ensures that there is no interference between tasks or threads running on different cores. Fig 7 shows how two cores in a multicore system are physically isolated from each other but may share some resources.

High availability is often valued more than performance. To achieve high system availability, redundant hardware is commonly used to detect errors. Chip multiprocessors with a lot of the same resources, such as cores, memory and interconnection networks, would seem to be the best starting points for developing high-availability solutions on chips. But on the contrary, doing so poses significant challenges with respect to error containment and replacing the faulty component. Increasing silicon and transient fault rates with future technology scaling make the issue worse.

➢ *Temporal Isolation*

It ensures that the execution of software on one core does not impact the temporal behavior of software running on another core.

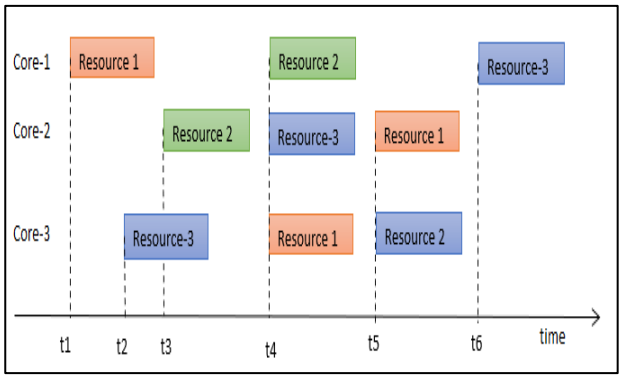

Fig 8: Temporal Isolation

Resources are shared but at a specific time, only one core has access to the specific resource.

Assuring temporal isolation and adequate execution time determinism in a single-core real-time domain is crucial. An extra amount of slack execution time is added to account for nondeterministic disturbances, such as interrupts, with carefully considered temporal effect. Applications are executed in such a way that they seem to be in parallel, but frequent context switches are performed to provide each application a fair share of resources in the allocated time. Effectively, the complete processor is partitioned using time division multiple access (TDMA), with some exceptions. Direct Memory Access (DMA) transfers and the resulting memory bus contention are examples of exceptions that have temporal effects, but they usually have well-known solutions and their implementation is extremely limited.

When moving to multicore, the sequential execution assumption in a single-core system can no longer be made secure. The applications now run simultaneously and have non-exclusive access to resources. Within a multicore environment, applications contend for access to system resources, which is typically mediated through hardware implementation in an implicit manner. The execution has non-deterministic temporal delays as a response. A subset of the effects described in this paper might be present in each processor architecture.

B. Energy Efficiency

Architects can use multicore processors to reduce the number of embedded computers. Multinational systems overcome excessive heat generation due to Moore's law, which reduces the need for cooling. As less energy is released in the form of heat while using multicore processing, battery life is increased while power usage is reduced. [2,9,10]

Various Energy Efficiency techniques in multicore architecture can be implemented based on application.

➢ *They are:*

- Voltage and Frequency scaling[15]
- Cache configuration
- Individual core control
- Low-power and high-power domain configuration

C. Concurrency

Multicore processing improves the intrinsic support for real (as opposed to virtual) parallel processing within individual software applications across multiple applications by allocating applications to various cores as illustrated in fig 8. Concurrency is the capacity for many tasks to be carried out simultaneously and in any sequence without impacting the output.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

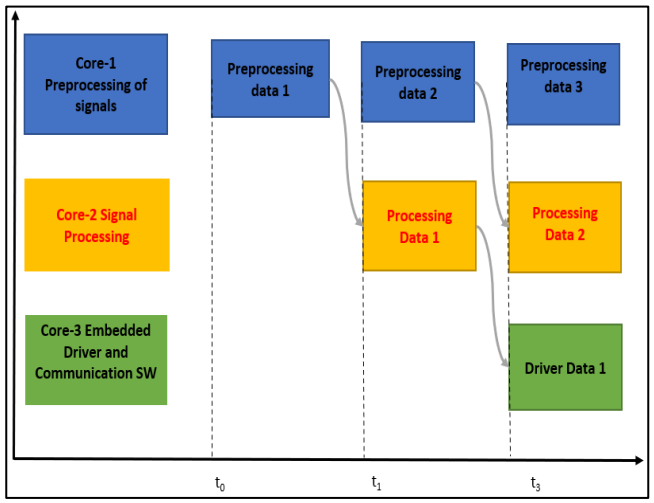

Fig 9: Concurrency

D. Performance

Performance can be improved by multicore processing by running several applications simultaneously. The closer spacing between cores on an integrated chip allows faster cache speeds and lower latency when compared to using separate processors or computers. The number of cores, degree of actual concurrency in the software and use of shared resources are all factors that affect performance[4].

Multi processors(multiple CPUs on different chips attached to the same motherboard) can accomplish high-performance and speed. However, alternative research trends promoted the development of multicore CPUs to simultaneously decrease power consumption and boost computing speed because of their unfavorably high-power consumption. The multicore CPU architecture gave speed and efficiency with less power usage for power-hungry applications.

➢ *Performance Analysis*

To guarantee reliable performance at a specific cost, performance analysis, a criterion that defines the performance of a system, is necessary at every stage of the computer system life cycle. The demand for performance analysis is derived from the following factors:

- The requirements of computer users today are different from those of 20 years ago.
- The popularity of computer technology has increased drastically

These factors have resulted in the development of different computer systems with unique performances. These developments require continuous performance analysis that meets the user's demands and helps to select the best alternative which provides higher performance at a given cost.

The selection of proper evaluation criteria is the initial step for performance analysis. Analytical modeling, simulation and measurement are the primary methods. In evaluating multicore CPUs performance, the techniques used depend on different considerations. However, we cannot trust the result of one technique unless we compare that result with other techniques. [4,1]

➢ *Performance Metrics*

Performance Metrics that are used in evaluating multicore CPUs performance for applications.

- Throughput: Throughput can be defined as the average time taken by the cores for executing the processes.
- Response time: It is defined as the interval between when a request is submitted and when the system starts generating the response. Ideal response time of a core must be minimum.
- Execution time: The time taken to complete program execution. The major concern is to achieve minimum execution time without having to compromise energy, thermal issues and other vital factors.
- Energy: It is the power needed to run a program. To reduce power consumption, multicore systems must have separate power management units. The architecture must be such that the thermal energy dissipated by cores is also minimum.
- Memory bandwidth: It is described by the rate of data read from the CPU core to the RAM (Random Access Memory) or vice versa. In another way, it can be defined by how often the data can be accessed to and from the memory. When the memory bandwidth decreases, the core will have difficulty processing data or even loading it. This will result in a delay in processing it.
- Memory latency: The time delay for the memory controller signaled the memory module to access a byte or word from the RAM and until it is retrieved by the memory module. It is also known as CAS (Column Address Strobe) latency.
- Percent of memory contention: Memory contention is a condition where two programs or parts of a program attempt to read data from the same memory block at the same moment. Contention value between 5% and 50% is a "normal overcommit".

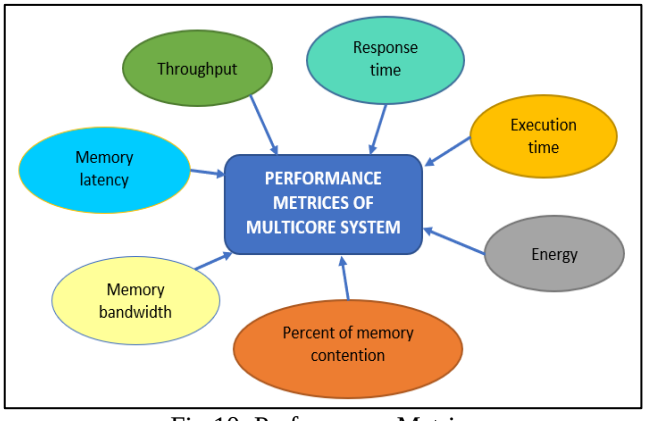

Fig 10: Performance Metrics

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

V. MULTICORE SYSTEM EVALUATION WITH EXAMPLES

➢ *The Following List of Parameters and Factors Need to be Considered for the Evaluation of Multicore Systems*

- Memory: Memory architecture used and memory speed can affect the performance of the multicore CPU.
- Scalability: Affects performance based on the rate of increase of the workloads (i.e. tasks). If the number of tasks is equal to the number of cores, then performance scales linearly but in reality, if the number of tasks exceeds the number of cores then, the performance depends on various factors like cache utilization, etc.
- I/O bandwidth: I/O bandwidth refers to the rate at which a core reads or writes to the network. This can affect the performance as it utilizes the CPU cores which leads to more resources consumption
- Inter-core communication: The interaction between cores in multicore CPU's are vital. This can be implemented by various mechanisms, affecting overall CPU performance due to shared workloads between cores. Cores communicating effectively can increase the performance of the system as a whole
- Operating system (OS): OS is a software that manages the hardware. It assigns tasks to cores based on a scheduling mechanism. It provides an orderly manner of allocation of cores for programs, memory etc.
- CPU clock speed: Clock speed affects processor performance. Slow clock speed reduces throughput. Increasing the clock speed to a certain extent is only permitted. It is not possible to increase clock speed beyond this limit.
- Numbers of cores: Number of cores present and their types can affect the CPU performance as multicore architecture divides the workload between cores.
- Cache coherent: The challenge of maintaining consistency in the data in caches is known as cache coherence. Multicore architectures uses different caching mechanisms as the cache is shared among the cores. Maintaining the consistency of data will affect the performance of the cores.

Considering all these multicore evaluation parameters, a few examples are discussed with respect to ARM architecture. ARM Cortex-A cores are utilized for devices that run powerful operating systems like Linux or Android which require High-performance and computational power. For real-time applications which require deterministic interrupt response to meet hard real time requirements, Cortex-R cores with features like tightly coupled memory are recommended. Cortex-M core are optimized for low-power, cost-sensitive embedded systems. An example of multicore architecture that can be considered for power optimization based and performance and safety or real time based is shown in the figures below.

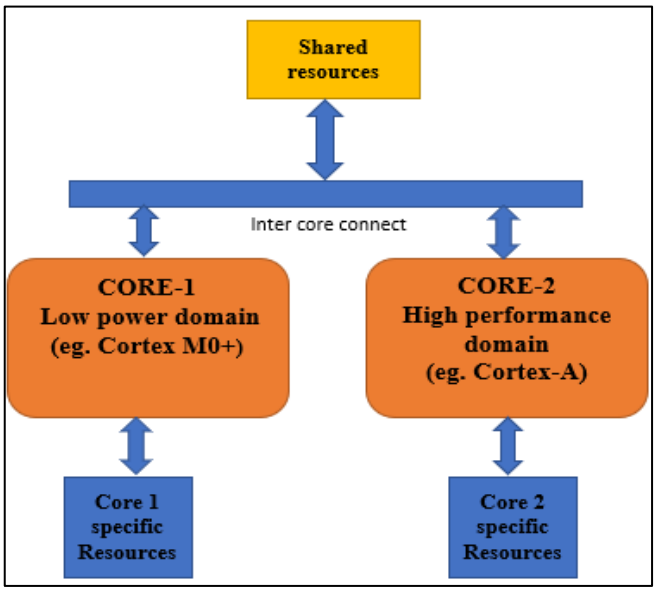

Fig 11: Power Optimized ARM based Multicore Architecture

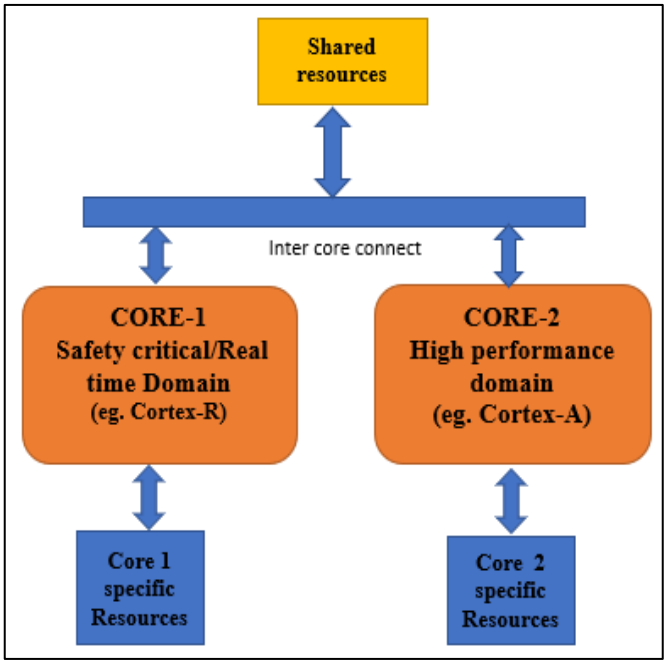

Fig 12: Performance and Realtime ARM based Multicore Architecture

The fig 10 shows an ARM based multicore architecture that can be considered where optimum power utilization is main use case. There is always a tradeoff between power and performance whenever we consider any system.

The fig 11 represent a multicore architecture in which the cores are selected based on the performance of the cores. The cores used for an application can also be a core cluster where two or more homogeneous cores can be used.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY458

VI. SUMMARY

In modern automotive applications like ADAS, both homogeneous and heterogeneous categories of multi core processors are widely used based on application. Using multicore system, we can increase the performance of a system without increasing the clock frequency. In this paper, different parameters, considerations and factors like core-connect buses, isolation and energy efficiency, are discussed with reference to homogeneous and heterogeneous multicore systems. In general, where performance optimization, system reliability and reduction of power consumption are the focus areas despite architectural and design complexity, heterogeneous systems are beneficial. However, if architectural & design simplicity is the key criterion, homogeneous systems are a better choice. The combination of both heterogenous and homogenous cores also can be considered for system designs. All these aspects need to be evaluated in a balanced way to select an appropriate multicore system based on the application needs for migrating from an existing single core system to a multicore system. These topics can be part of further study by considering specific multicore SoCs.

ACKNOWLEDGEMENTS

We sincerely wish to thank Mr. Sudeepth Puthumana, Head of Market Segments, Autonomous Mobility (AM) India, Continental Tech Center India (TCI), Bangalore; Mr. Vinayaka Nagaraja, Head of Market Japan, Korea and India; Mr. Praveen Kumar BL, Head of Application Projects, Japan, Korea and India Segment for providing this opportunity and giving valuable support to us for working on this study. We would like to thank all the members of Market Japan, Korea and India SW Architecture team, Mr. Srinath Murthy, Head of Market Europe and our sincere thanks to Corporate Communications Team and Technical Paper Publishing team at TCI for their cooperation.

REFERENCES

[1]. A.S. Radhamani, "Performance Analysis of Homogeneous and Heterogeneous Multicore Processor Using Static and Dynamic Schedulers," Asian Journal of Information Technology
[2]. C. Leech and T. J. Kazmierski, "Energy Efficient Multicore Processing," Electronics
[3]. Sergio Saponara and Luca Fanucci, Homogeneous and Heterogeneous MPSoC Architectures with Network-On-Chip Connectivity for Low-Power and Real-Time
[4]. IntervalZero, "How to Optimize the Scalability & Performance of a Multicore Operating System," IntervalZero.com
[5]. Ajeya Naithani, Stijn Eyerman, Lieven Eeckhout. Reliability-Aware Scheduling on Heterogeneous Multicore Processors
[6]. M. Mitić and M. Stojčev, "A Survey of Three System-on-Chip Buses: AMBA, CoreConnect and Wishbone," International Journal of Electrical and Computer Engineering
[7]. J. Zamorano and J. A. de la Puente, "Memory Isolation in Many-Core Embedded Systems,"
[8]. N. Aggarwal, P. Ranganathan, N. P. Jouppi, and J. E. Smith, "Configurable Isolation: Building High Availability Systems with Commodity Multicore Processors,"
[9]. J. Cong and B. Yuan, "Energy-Efficient Scheduling on Heterogeneous Multicore Architectures,"
[10]. A. Merkel and F. Bellosa, "Memory-aware Scheduling for Energy Efficiency on Multicore Processors."
[11]. H. Omar, H. Dogan, B. Kahne, and O. Khan, "Multicore Resource Isolation for Deterministic, Resilient, and Secure Concurrent Execution of Safety-Critical Applications," IEEE Computer Architecture Letters, vol. 17, no.2, July Dec.2018
[12]. R. Usselmann, "OpenCores SoC Bus Review," Rev. 1.0, January 9, 2001
[13]. "Specification for the: WISHBONE System-on-Chip (SoC), Interconnection Architecture for Portable IP Cores," Revision: B.3, Released: September 7, 2002.
[14]. "Wishbone B4, WISHBONE System-on-Chip (SoC) Interconnection, Architecture for Portable IP Cores,"
[15]. Hwang-cheng Wang and Alagan Anpalagan, "Energy-efficient tasks scheduling algorithm for real-time multiprocessor embedded systems," Journal of Systems Architecture, vol. 57, no. 5, pp. 498-505, May 2011. doi: 10.1016/j.sysarc.2010.10.003.
[16]. Nik Jedrzejewski, "Three Reasons Why Embedded Heterogeneous Systems Are More Efficient," NXP Blog, Jan. 30, 2019.
[17]. ARM Cortex-A53 MPCore Processor Technical Reference Manual, Ver:r0P4,https://developer.arm.com/documentation/ddi0500/j/Introduction/About-the-Cortex-A53-processor.
[18]. IJSRD - International Journal for Scientific Research and Development. (2014, May 24). A comparative Study of Different system-on-Chip Buses based on Industry standards: AMBA, CoreConnect and Wishbone.
[19]. IJEERT - Kolte, Mahesh. (2014). Design and Verification Point-to-Point Architecture of Wishbone Bus for System On Chip.International Journal of Emerging Engineering Research and Technology. 2. 155-159.
[20]. ARM, "AMBA,"https://developer.arm.com/ip-products/system-ip/amba.
[21]. J. Yiu, "CHAPTER 6 - Cortex-M3 Implementation Overview," The Definitive Guide to the ARM Cortex-M3 (Second Edition), Newnes, 2010, pp. 99-108