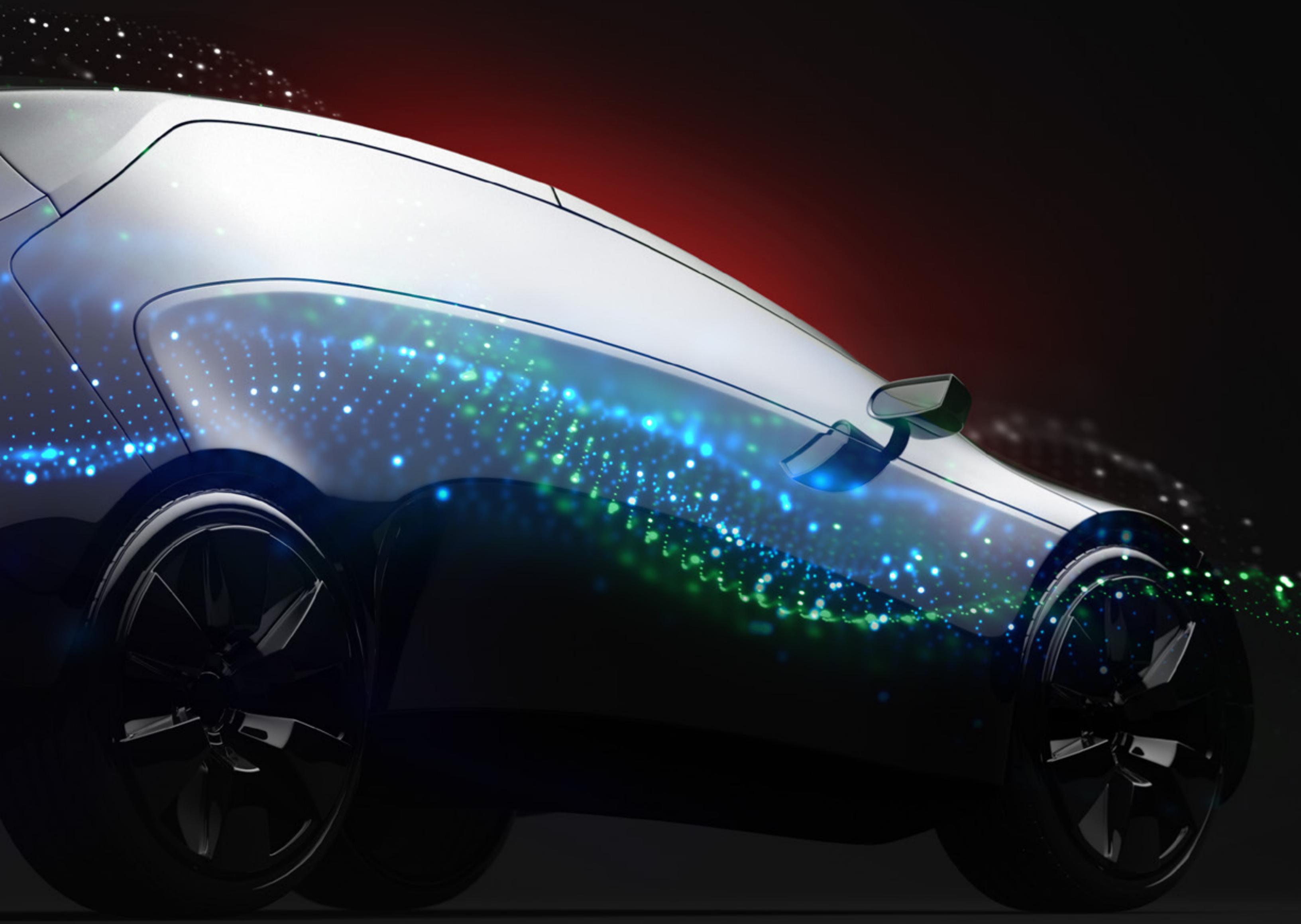
Reach Further™



Top 6 Autonomous Vehicle Use Cases You Need to Read

Top 6 Autonomous Vehicle Use Cases You Need to Read

Automotive solutions rely on electronic systems to implement autonomous capability in vehicles. With this capability, the electronic system can meet real-time processing requirements and interface with a range of high- and low-bandwidth sensors. Autonomous capabilities provide a secure and safe implementation that prevents unauthorized access and modification — and ensures safe operation and graceful degradation.

Within the autonomous vehicle space, several key systems must be developed:

- Autonomous vision systems
- Automotive RADAR
- Sensor aggregation and processing
- Safety certification
- Occupant monitoring
- Vehicle-to-vehicle and infrastructure communication

This e-book explores the challenges engineers face in the creation of these key systems, and identifies how Xilinx technologies and ecosystem can help overcome such challenges.

TABLE OF CONTENTS

Learn more about Avnet at
www.avnet.com

INTRODUCTION

Automotive safety and technology seem to have advanced light years since the first automobiles were created in the 18th and 19th centuries — with the rate of innovation and adoption growing increasingly rapid.

Today's autonomous vehicles are complex systems on wheels, operating at different levels of autonomy and integrating AI inside the cabin and out. Cameras and sensors all around generate and process tremendous amounts of data simultaneously. Complexity brings big challenges, however; these systems must have real-time performance, reliability and be able to communicate with each other.

Xilinx automotive-grade (XA) devices address the challenges of interfacing, performance, safety, latency and determinism that the autonomous vehicles of today and tomorrow require. The heterogeneous system-on-chip XA Zynq UltraScale+ MPSoC featured in this e-book, for example, offers a processing system and programmable logic along with any-to-any interfacing capability and the ability to accelerate functions such as machine learning inference in a single device.

INSIDE THE SYSTEM-ON-CHIP

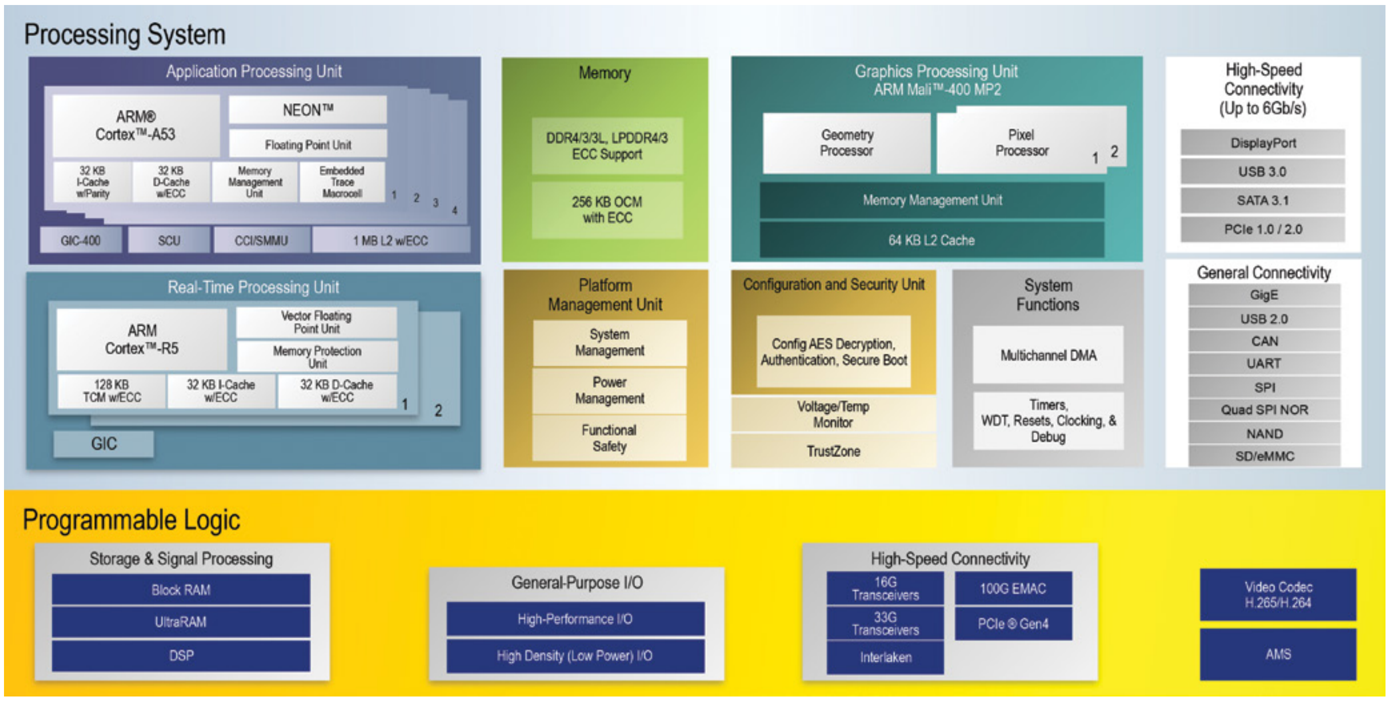


Image: Xilinx Zynq UltraScale+ MPSoC Architecture

The processing system inside the Zynq UltraScale+ MPSoC contains:

- Application processing unit (APU)
- Real-time processing unit (RPU)
- Platform management unit (PMU)
- Configuration security unit (CSU)

The 16nm programmable MPSoC contains a configurable logic block, block RAM, DSP elements that can operate at up to 890MHz, and more.

ANY-TO-ANY INTERFACING

Xilinx SoCs provide the developer with a range of interfacing solutions that can be accessed from the processing system. These range from low-speed, low-bandwidth interfaces used for communication and sensors (e.g., SPI, I2C and UART), to high-bandwidth, high-speed interfaces (e.g., USB, SATA and PCIe). The right PHY programmable logic provides any-to-any interfacing that enables bespoke, legacy and rapidly evolving standards to be implemented with ease.

REDUCING SWaP-C

The tightly coupled integration between the processing system and programmable logic enables designers to create solutions that meet the demanding size, weight, power and cost (SWaP-C) requirements in automotive systems. Such integration reduces circuit board area and the need for many devices, while lowering power consumption through less chip-to-chip transmission and fewer active deployed devices.

Xilinx automotive-grade devices are screened in accordance with AEC-Q100 for part approval. This increased component quality boosts overall system reliability and supports functional safety.

HELPING AUTOMOTIVE DESIGNERS MAKE THE SHIFT

Xilinx's broad portfolio of programmable technology and robust ecosystem of tools, IP libraries and frameworks are here to help designers adapt to the evolving autonomous vehicle landscape.

VISION SYSTEM FOR AUTOMOTIVE APPLICATIONS

INTRODUCTION

Autonomous vehicle capability requires the vehicle to be able to understand its environment. One of the most common ways of achieving a significant understanding of its environment is the use of embedded vision systems.

These vision systems enable the vehicle's systems to not only understand its surroundings, but also process images to detect and classify vehicles and objects. Such capability is the cornerstone for many autonomous operations, including traffic jam assistance, self-parking and stop/start highway driving and higher SAE levels.

ARCHITECTURE FEATURES AND CONSIDERATIONS

A common vehicle architecture is to use several cameras connected to a camera processing module. This camera processing module will provide the camera control, image capture and pre-processing before implementing higher-level algorithms such as machine learning inference.

The output from these algorithms is then communicated to the appropriate vehicle system for action. Typically, data extracted from the images will be passed to a vehicle's central processing system to further process and fuse the data with other sensors prior to making decisions that affect the vehicle controls such as maneuvering or braking.

SAFETY AND SECURITY

Due to the nature of the applications in which automotive vision systems are used, consideration must also be made for safety and security. This often means alignment with the ISO 26262 safety standard, so the architecture of the system needs to prioritize safety.

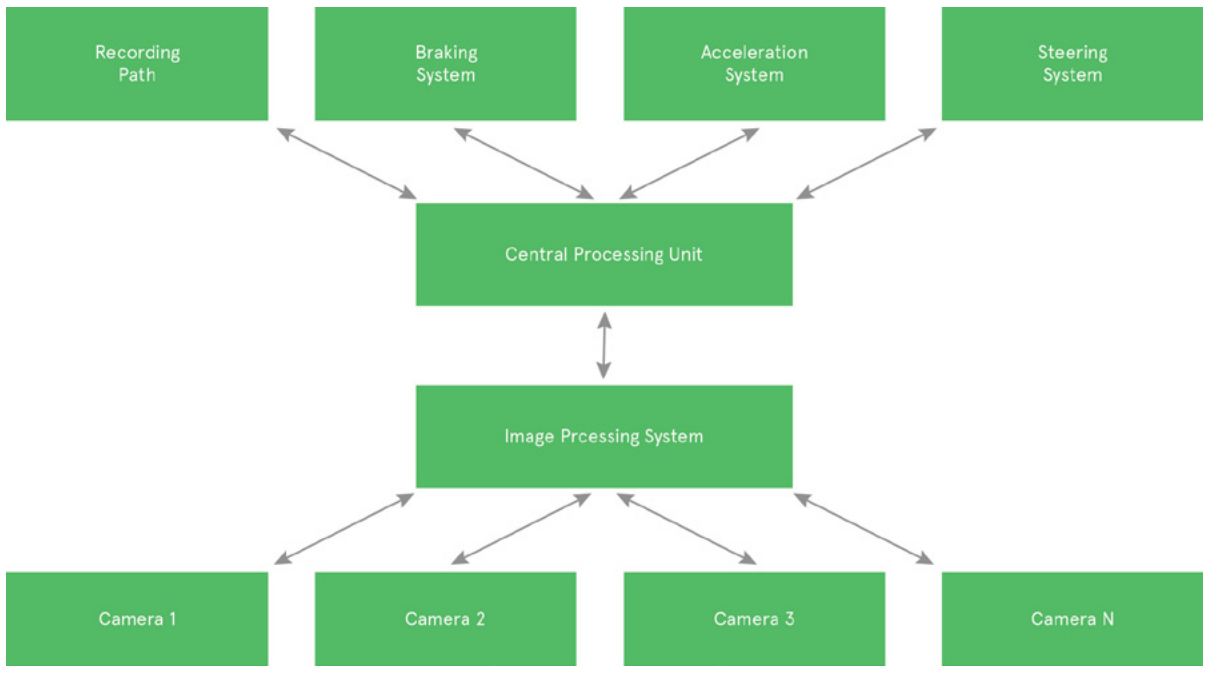


Figure 1: Vision system architecture in the average vehicle

In vehicle applications, minimizing cabling is critical as it adds significant mass and cost. Cameras — with their wide output buses, control channels and power — make this challenging, especially when several cameras are intended to be deployed, and as such, manufacturers use cameras based upon MIPI CSI-2 standard. Such an approach enables a reduction in cabling, while still achieving the high data rates needed for high-resolution, high-frame-rate cameras. To achieve this, manufacturers use a single coax cable for each camera and supply power, control and video using Gigabit Multimedia Serial Link (GMSL) drivers and receivers. These drivers and receivers convert between GMSL and MIPI at the point of use, enabling interfacing with a wide range of cameras and processing systems.

Such capability leads to the need to be able to interface with and control several cameras while also being able to control the GMSL links. Of course, the system must also be able to communicate with the downstream vehicle control system using standard automotive interfaces such as CAN, CAN-FD, Automotive Ethernet and FlexRay.

Being able to interface with several high-speed MIPI interfaces, while providing the necessary downstream communications interfaces, can lead to design and device tradeoffs to achieve the required interfacing. Typically, the tradeoff may be between interfacing for performance with an off-the-shelf solution or achieving the required interfacing and performance at significant expense by creating a custom solution. This is especially true as the MIPI DPHY can be challenging to implement as it needs to support low-swing high-speed differential signals and single-ended low-speed control signals in the same Input / Output cell.

The ability to process and extract information from the sensors requires significant processing capacity, enabling decisions to be taken by the vehicle in a timely manner. As such the image processing system is required to provide a low-latency and highly deterministic response. Often this can lead to the need for a dedicated high-performance processor or ASIC to achieve these requirements. Architectural decisions, such as the use of external DDR memory to store the image stages in the processing pipeline, will significantly impact the overall latency and determinism due to the nature of DDR being a shared system resource and the time taken for access.

Running on top of the processors is the software framework, which implements the image-processing algorithms. To minimize development time and focus on value-added activities, one commonly used framework in image-processing applications is OpenCV. OpenCV provides C, C++ and Python image-processing functions that can be quickly and easily deployed. This approach saves time in the development process as it enables developers to focus on their value-added activities and specialist algorithms.

ADDRESSING THE CHALLENGES

The challenges of interfacing, performance, latency and determinism can be addressed using a heterogeneous system-on-chip such as the Xilinx Automotive XA Zynq UltraScale+ MPSoC. These devices offer a diverse range of processing system and programmable logic coupled with any-to-any interfacing capability and dedicated resources for acceleration of functions such as machine learning inference.

THIS SPLIT BETWEEN THE PROCESSING SYSTEM AND PROGRAMMABLE LOGIC ENABLES THE DEVELOPER TO IMPLEMENT A COMPLEX IMAGE-PROCESSING PIPELINE IN THE PROGRAMMABLE LOGIC WHILE RUNNING HIGH-LEVEL ALGORITHMS IN A HIGH-PERFORMANCE PROCESSOR SUCH AS AN ARM CORTEX-A53 OR CORTEX-A72.

Due to the flexibility of programmable logic I/O the high-performance I/O Bank of Xilinx Zynq UltraScale+ devices can directly support the MIPI DPHY — removing the need for an external PHY, saving on-board area and providing for a more compact solution when several MIPI interfaces are used. The large number of I/O pins available in the programmable logic enables a significant number of MIPI interfaces to be implemented, freeing the designer from the constraints imposed by more traditional CPU and GPU solutions.

Within the programmable logic design, the MIPI video stream can be decoded using a MIPI-CSI2 IP cores. The output of the MIPI-CSI2 IP block will be video-formatted as an AXI Stream, and this use of a standard streaming interface within the programmable logic enables the image-processing pipeline to be easily constructed by utilizing existing IP blocks from the Xilinx IP library, third-party IP providers or, alternatively, by leveraging High Level Synthesis if custom algorithms are required. This enables the creation of a true image-processing pipeline within the programmable logic, significantly reducing latency and increasing determinism.

xfOpenCV LIBRARY

To be able to leverage the high-level algorithmic models created in frameworks such as OpenCV, Xilinx provides the xfOpenCV library. The xfOpenCV library contains several commonly used OpenCV functions that can be synthesized using High Level Synthesis in the Xilinx Unified Software Development Tool Vitis into the programmable logic. This enables high-level modelling using OpenCV and then quickly and easily the same functions can be implemented within the programmable logic pipeline without the need to write a line of hardware description language.

Should the image-processing or downstream processing require H264/H265 encoding or decoding, the EV variant of the XA Zynq UltraScale+ MPSoC offers a built in Video CODEC Unit along with associated UltraRAM for buffering streams.

Many autonomous automobile capabilities require system intelligence to be able to identify and classify objects detected by the image-processing algorithm pipeline. For Xilinx devices, Vitis AI enables the acceleration of commonly used ML/AI frameworks including Caffe and TensorFlow. Vitis AI also provides a Model Zoo, AI Compiler, Optimizer, Quantizer and Profiler to deploy applications on to the Deep Learning Processing Unit.

The information extracted from the image processing pipeline, machine learning and higher-level algorithm can be communicated to the central vehicle controller using one of the supported standards, e.g., Automotive Ethernet, CAN or CAN-FD using interfaces available in either the processing system or the programmable logic.

Should a high-performance test or display interface be required for prototyping and testing of the vision application, such interfaces can be easily implemented using the high-speed interfaces available on the processing system or the programmable logic.

EXAMPLES OF SUCH INTERFACES INCLUDE GigE VISION, 10 Gig ETHERNET AND PCIe, WHILE THE DISPLAYPORT INTERFACE CAN BE USED TO OUTPUT BOTH LIVE VIDEO FROM THE PROGRAMMABLE LOGIC OR PROCESSED FRAMES FROM THE PROCESSING SYSTEM.

As safety is at the heart of the autonomous vehicle operation and especially automotive vision, the platform management unit can be used within the XA Zynq UltraScale+ MPSoC to monitor the device's internal supply voltages and die temperatures. The device also provides an internal segmentation between the processing system and the programmable logic to remove common cause failures, enabling the processing system and programmable logic to be independent of each other monitoring performance and status.

Also, located within the processing system is the real time processing unit (RPU), which has been designed with ISO 26262 applications in mind and contains dual Arm Cortex-R5 processors which operate in lockstep. The RPU provides the ability to implement safety processing if necessary, directly within the vision processing system. A typical example might include monitoring the status of cameras and communication links. To support communication between the application processors and the real-time processors, the OpenAMP framework can be used to manage communications in the multiprocessor environment.

The SATA and PCIe interfaces provided within the processing system provide the capability to record system information and even video frames if required for system monitoring, logging and maintenance.

SUMMARY

Vision systems play a critical role in the development of autonomous vehicle operation, and they come with several constraints due to the large number of cameras deployed. These challenges include interfacing, low-latency processing and determinism. Using heterogeneous SoC such as the XA Zynq UltraScale+ MPSoC, designers can leverage not only the capabilities of the device but also the supporting ecosystem to create a solution that can meet the challenges of interfacing, lower latency, increased determinism and safety required for automotive vision applications.

RADAR FOR ADAS

INTRODUCTION

RADAR is a keystone technology for autonomous vehicle operation. Unlike LiDAR and vision systems, RADAR works across all lighting and weather conditions including fog and heavy rain. However, to provide sufficiently accurate information for operation at the higher SAE autonomous levels four-dimensional RADAR often used, 4D RADAR provides information on azimuth, elevation and slant range but also the Doppler frequency. The Doppler frequency enables the determination of the target's velocity to be detected.

4D RADAR therefore provides a system that can have a large field-of-view (100 degrees), fine spatial resolution (1 degree) and longer-range capabilities (circ 300 m). This enables autonomous operation at SAE levels 2 and 3 with capabilities such as traffic jam assist, highway driving and self-parking.

RADAR SYSTEMS REQUIRE THE ABILITY TO GENERATE, TRANSMIT, RECEIVE AND PROCESS RADIO FREQUENCY SIGNALS.

ARCHITECTURE FEATURES AND CONSIDERATIONS

RADAR systems require the ability to generate, transmit, receive and process radio frequency signals. Traditionally, this requires the use of baseband digital processing for signal generation and processing along with an RF front-end, which provides the up and down conversion from the baseband. Signals are converted to and from the analog domain using analog to digital converters (ADC) and digital to analog converters (DAC). The signal processing within the digital baseband includes the RADAR chirp generation along with the post processing of the results.

WITH 4D RADAR, HOWEVER, A MONOPULSE, BEAM-FORMING APPROACH IS OFTEN USED TO STEER THE BEAM ACROSS ITS SWEEP. ON EACH SWEEP, SEVERAL CHIRPS ARE SENT OUT CONTINUALLY IN A FRAME THAT COVERS A DEFINED TIME PERIOD.

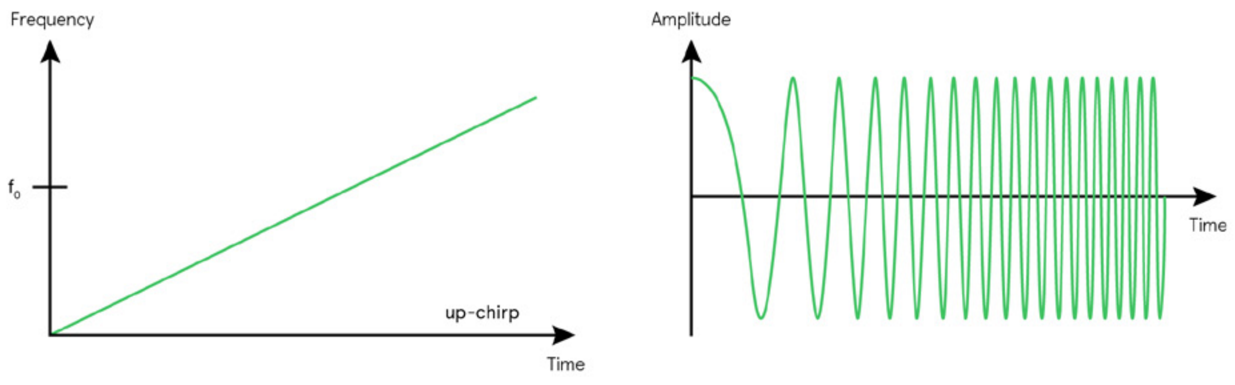


Diagram 1: Chirp Signal

The baseband processing is then able to process the received chirps using a two-dimensional FFT. The first FFT on the chirp responses provides the range information — typically this FFT will be the larger of the two, e.g., 4096 points. The results from the first FFT are subjected to a second FFT, which looks across the entire frame and provides the velocity information. This second FFT is smaller than the first, typically less than 1024 points. The azimuth and elevation can be determined from the sum and difference channels provided by the monopulse receiver.

In addition to the chirp generation and 2D FFT, the baseband processing also needs to be able to implement decimation and interpolation, algorithms such as constant false alarm rate (CFAR) adaption as well as processing and communication with downstream modules in the vehicle to enable the vehicle to address identified targets safely.

INFORMATION FROM THE RADAR IS REQUIRED FOR THE VEHICLE TO OPERATE SAFELY WITHIN ITS ENVIRONMENT, SO THE ALGORITHMS IMPLEMENTED WITHIN THE BASEBAND PROCESSING MUST BE ABLE TO PROVIDE A LOW-LATENCY RESPONSE.

A visual plot of the RADAR system may also be required to be generated, either as part of the prototyping and commissioning or as part of the vehicle infotainment system. This plot should display the positions, velocities and tracks for the detected targets.

At the electronic level, the baseband processing must be capable of interfacing with high-speed ADC and DAC devices. This brings with it risk in the circuit board design, including signal integrity for the high-speed routing and segmenting digital signals away from the low noise analogue and RF sections — all of which must be achieved within the tight size, weight and power, and cost (SWaP-C) target for the overall system.

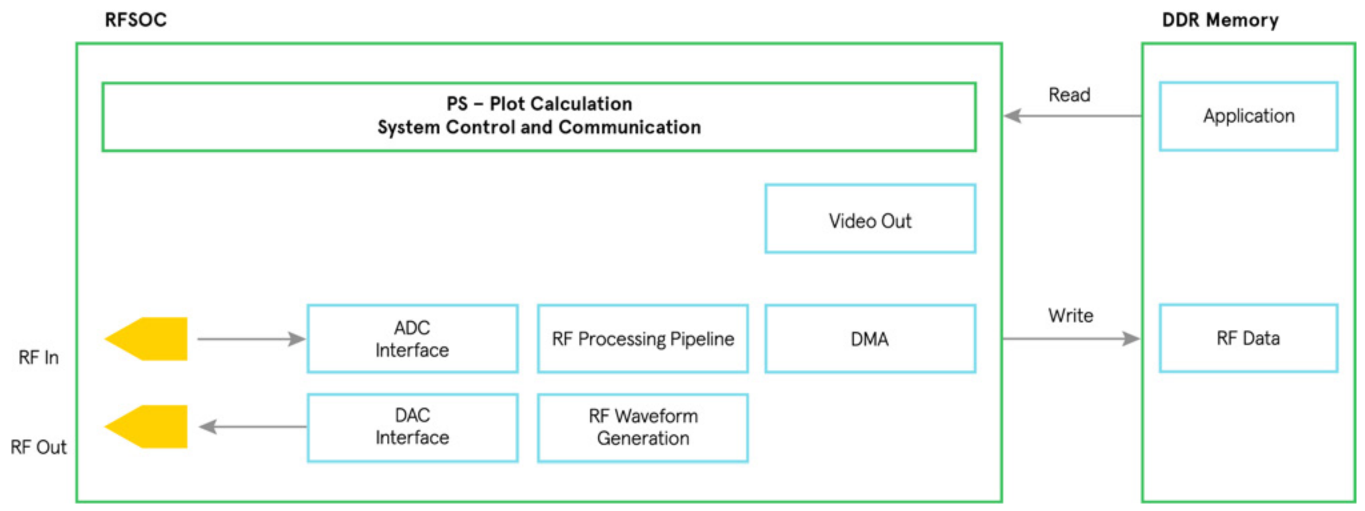
Addressing these performance and SWaP-C requirements can be challenging and lead to tradeoffs in the overall solution architecture.

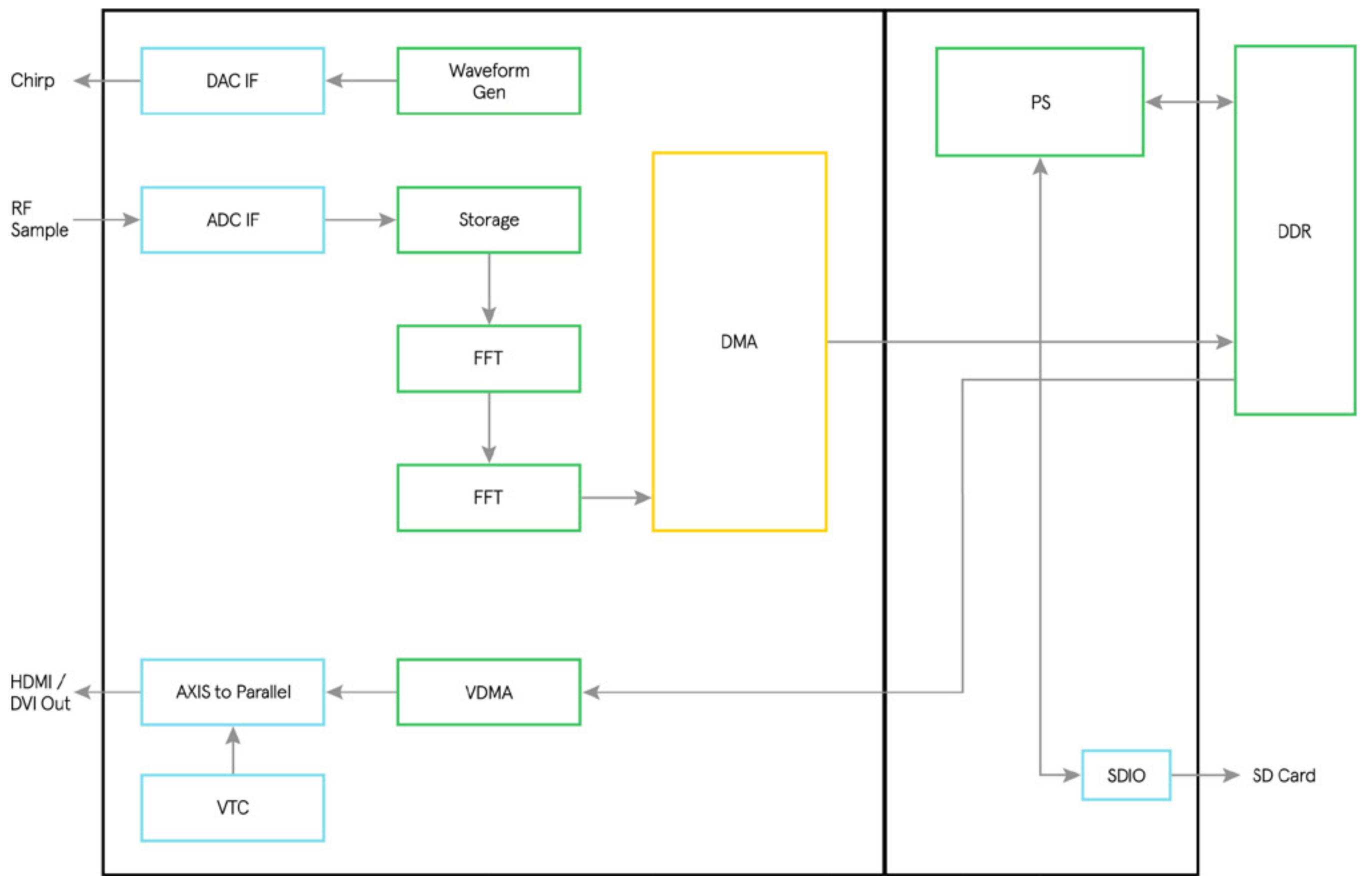
ADDRESSING THE CHALLENGES

The Xilinx Zynq UltraScale+ RFSoC* can help the designer address both the performance and SWaP-C challenges. The RFSoC range of devices contains not only a processing system with high-performance quad core Arm Cortex-A53 64-bit processors, but also programmable logic that includes gig sample per second ADC and DAC along with necessary supporting infrastructure, including up and down converters, complex mixers and interpolators and decimators. This significantly minimizes the complexity of interfacing with the RF front end, while also reducing the overall required implementation area.

Such a system enables the application processing unit (APU) within the processing system to implement high-level applications and communication frameworks with the downstream vehicle systems using standard interfaces such as CAN and automotive Ethernet. Applications running within the APU may make use of embedded Linux operating systems and the Xilinx PetaLinux build flow. Alternatively, other operating system may also be deployed with the appropriate stacks for example RTEMS or QNX.

*Zynq UltraScale+ RFSoC devices are not part of the Xilinx Automotive-grade portfolio.





Diagrams 2 & 3: Block diagrams on how to address both performance and SWaP-C challenges with RFSoCs.

To provide real-time control of the RADAR system, the real-time processing unit within the processing system can be used. The RPU, designed to implement safety-critical functionality required to achieve ISO 26262 certification, contains lockstep dual core Arm Cortex-R5 processors. These processors within the RPU can be used to implement safety functionality that monitors in conjunction with the System Monitor and Platform Management Unit for single point and latent faults.

Supporting a low-latency solution is the availability of not only configurable logic blocks within the programmable logic, but also dedicated DSP elements and diverse memory storage arrangements from distributed RAM to Block RAM and UltraRAM.

The RFSoC range of devices provides between 3145 and 4272 dedicated 48-bit DSP elements, distributed within the programmable logic region. These DSP elements are designed to support implementation of FFTs, systolic and multi-rate FIR filters, CIC filters and both real and complex multipliers and accumulators. If 48-bit resolution is not required, the designer can leverage the Single Instruction Multiple Data (SIMD) operation mode and perform dual 24-bit operations or quad 12-bit operations.

PROGRAMMABLE LOGIC

The programmable logic element of the device enables the implementation of the chirp generation and resulting signal processing pipeline including 2D-FFT, CFAR and location extraction. Programmable logic enables a parallel implement of the algorithm as such the implementation offer a low-latency response and increased determinism.

Local storage of data enables higher system performance. To support this, Block RAM and UltraRAM elements within the programmable logic can be used.

Block RAM is idea for small buffers, memories and FIFOs that store data between algorithmic stages, while UltraRAM is designed to enable the replacement of external memories when larger quantities of data need to be stored. UltraRAM blocks are larger than Block RAM at 4K by 72 bits providing 288Kb per block compared to 36Kb for Block RAM. Architecturally UltraRAM blocks can be cascaded to create large internal memories used to store data captured from the ADC or for output on the DAC. Within the RFSoC range of devices, UltraRAM capacities vary between 13.5 Mb and 22.5 Mb depending on device.

THE MAIN ELEMENT OF THE RFSOC IS THE ADC AND DAC CONVERTERS, WHICH CAN OPERATE AT GIGA SAMPLE PER SECOND (GSPS) SAMPLING RATES.

Depending upon the device generation, this sample rate can vary between 2 GSPS for the ADCs in generation one to 5GSPS for generation three devices, with supporting RF input bandwidth to enable operation across all four Nyquist zones. Sampling rates for the DAC range between 6.5 GSPS in generation one to 10 GSPS in generation two. This combination of sampling rate and wide RF input bandwidth enables a significant reduction in the complexity of the RF Tx and Rx paths.

To ease baseband processing within the programmable logic design, each ADC or DAC is supported by digital up and down converters, decimators and interpolators, complex mixers and numerically controlled oscillators. This allows the RF front-end design to be configured following a higher-level system approach rather than a low-level Hardware Description Language (HDL).

To enable system optimization and configuration at run time, the setting of the ADC, DAC and supporting up and down converters, mixers and NCO can be set via the software executing on the processing system.

Development of the programmable logic contents can leverage the extensive IP catalog provided with the Vivado Design Suite. Alternatively, should specialist algorithms be required, these algorithms can be implemented using a high-level language such as C, C++ or even MATLAB. These tools leverage the ability of high-level synthesis tools, such as The Vitis Unified Software Platform.

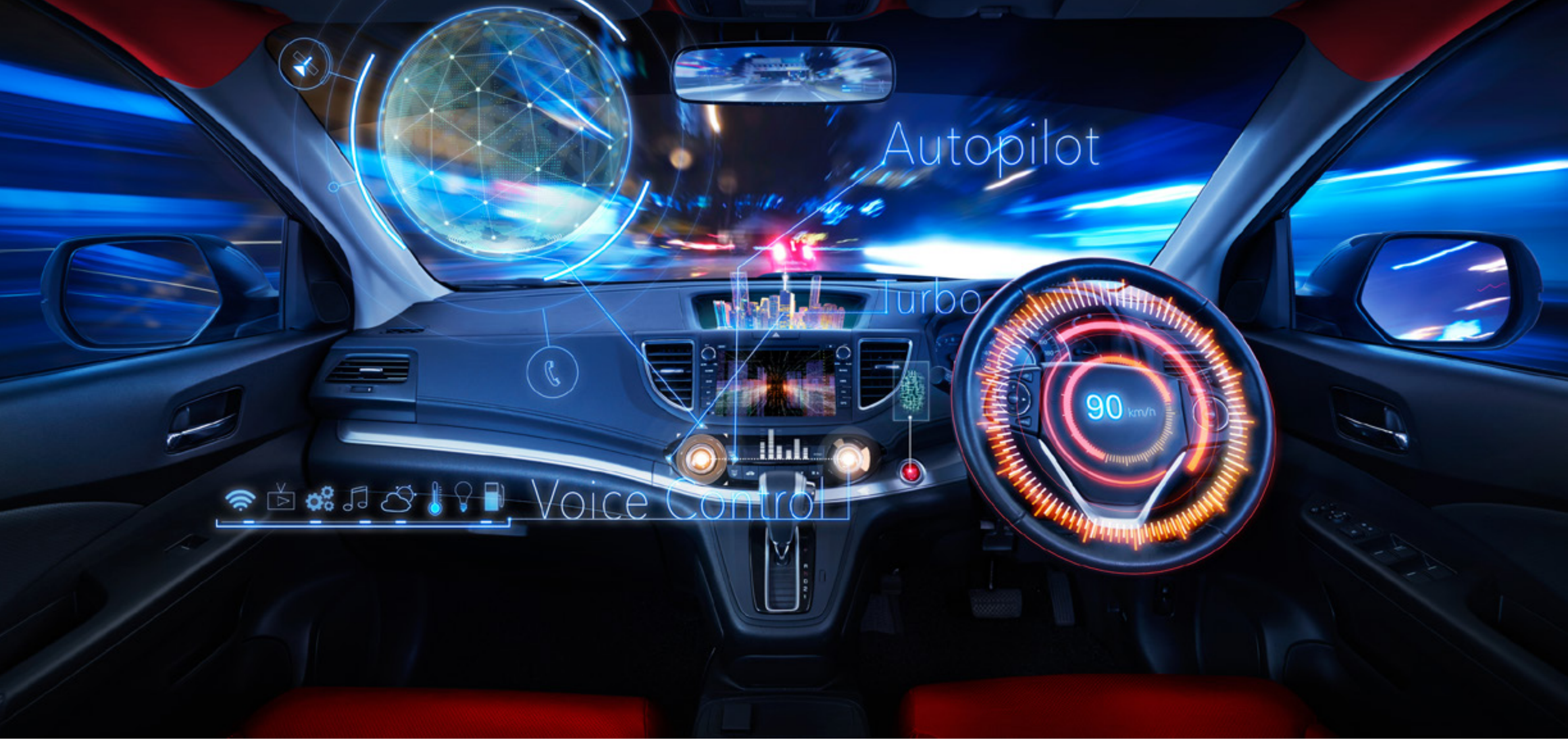
Should a visual RADAR plot be required by the system, the DisplayPort interface within the processing system can be used to generate this output. The DisplayPort output is capable of outputting video frame from both the processor system and programmable logic. Alternatively, the flexible programmable logic I/O can be used to implement HDMI/ DVI or other bespoke video output standards.

SUMMARY

The tight integration of processors, programmable logic and RF data converters enables the creation of a system that offers the high performance and low latency required for automotive 4D RADAR. The ecosystem of supporting design tools enables the creation of processor, programmable logic and system applications with ease.

VITIS

Vitis is a unified software development environment that enables the creation of system applications running in both the application processor unit and the real-time processor unit (RPU). Vitis also supports OpenCL which enables acceleration kernels to be created using high-level synthesis in the programmable logic.



AGGREGATION, FUSION AND ACCELERATION IN AUTOMOTIVE APPLICATIONS

INTRODUCTION

Autonomous capability is one of the hottest topics in automotive development, with existing and new automotive manufacturers developing autonomous vehicles. These developers are benefiting from Moore's law, which has enabled significant increases in processing capability and sensor technology while also lowering the cost.

This increase in processing and sensor capability is significant as it enables the vehicle to understand and safely interact with its environment in real-time using a diverse range of sensors including GPS, RADAR, LiDAR and vision systems. Different sensors, algorithms and processing capability enable different levels of autonomous capability.

Autonomous capability is a complex area that spans a wide range, from fully autonomous operation to shared control with the driver. The Society of Automotive Engineers (SAE) has defined several levels of autonomous capabilities:

SAE Level	Name	Examples	Vehicle Control	Monitoring	Fall Back Control	Vehicle Capability
0	No Automation	N/A	Human Driver	Human Driver	Human Driver	N/A
1	Driver Assistance	Adaptive Cruise Control / Lane Keeping & Parking Assist	Human Driver & Vehicle	Human Driver	Human Driver	Some Driving Modes
2	Partial Automation	Traffic Jam Assist	Vehicle	Human Driver	Human Driver	Some Driving Modes
3	Conditional Automation	Full Stop & Go Highway Driving Self-Parking	Vehicle	Vehicle	Human Driver	Some Driving Modes
4	High Automation	Automated Driving	Vehicle	Vehicle	Vehicle	Some Driving Modes
5	Full Automation	Driverless Vehicle Operation	Vehicle	Vehicle	Vehicle	All Driving Modes

Figure 1. SAE levels of autonomous capabilities

Each of these modalities provides data on the vehicle's overall environment, and forming a complete picture requires fusing these diverse elements together. The different sensor deployments and modalities will vary depending on the level of autonomy being implemented. However, cameras will be used for applications such as lane departure warning, blind spot detection and traffic sign recognition, while 4D RADAR can determine the distances, velocities and direction of tracked objects.

For partial automation (level two) and above, the ability to comprehensively understand the vehicle's environment is crucial to safe navigation as the vehicle identifies its location and surrounding obstacles. The system understands its environment using camera, RADAR and LiDAR along with GPS data. GPS data on its own is insufficient because its accuracy varies and it is easily blocked by buildings and infrastructure.

The different sensor modalities used present a significant challenge to developers of autonomous capability. Each sensor modality presents its data in a different format and using different interfaces (e.g., MIPI, LVDS, SPI or I2C).

High-bandwidth interfaces, such as RADAR/LiDAR and vision systems, require significant preprocessing to identify targets and classify and detect objects. At the same time, these high-bandwidth interfaces require dedicated processing complex chains that can impact the responsiveness and determinism of the overall system.

Once all the information from the different sensor modalities has been processed and collated, higher-level decision-making algorithms can process it and use it to safely interact with the vehicle's environment.

In the event of a failure, autonomous capability has the potential to result in death or injury to people or to incur significant environmental damage. To minimize the risk of such failures, the development of the system must comply with regulatory standards, such as ISO 26262.

ARCHITECTURE FUNCTIONS AND CONSIDERATIONS

At the heart of implementing an autonomous capability is the central processing system. To successfully implement autonomous capability, the central processing system must perform the following functions:

- Sensor Interface and Processing — basic processing, routing and switching of information between processing units and accelerators within the processing unit. Examples of post-processing include image-processing pipelines, object detection and 4D RADAR post-processing (e.g., two-dimensional FFT, CFAR and track extraction).
- High-performance serial processing — data extraction, sensor fusion and high-level decision-making. In some applications, neural networks will be implemented within the high-performance serial processing.
- Safety processing — real-time processing and vehicle control based on the detected environment provided by pre-processing in the distributed and parallel databases (DAPD) device and results from the neural network acceleration and high-performance serial processing elements. DAPD interfaces with the different sensor modalities performing basic processing, routing and switching of information between processing units and accelerators within the processing unit.
- Neural Network Acceleration — accelerates neural networks for object detection and classification (e.g., pedestrian and car detection, and road sign classification).

Due to the demands of the application, these functions could be implemented across multiple devices connected using an automotive communication standard such as Automotive Ethernet.

CHALLENGES

As autonomous systems will be mass produced, they must be able to comply with the size, weight, power and cost (SWaP-C) targets defined by the manufacturer. These challenges faced by developers of autonomous capability can be summarized as:

- Ability to interface with several high-bandwidth sensor modalities
- Ability to implement complex post-processing algorithms on high bandwidth sensor data, (e.g., image-processing pipelines, 4D RADAR, etc.)
- Processing capacity to implement the complex post-processing and high-level decision-making in real time
- Ability to comply with standards such as ISO 26262
- Achieve demanding SWaP-C targets

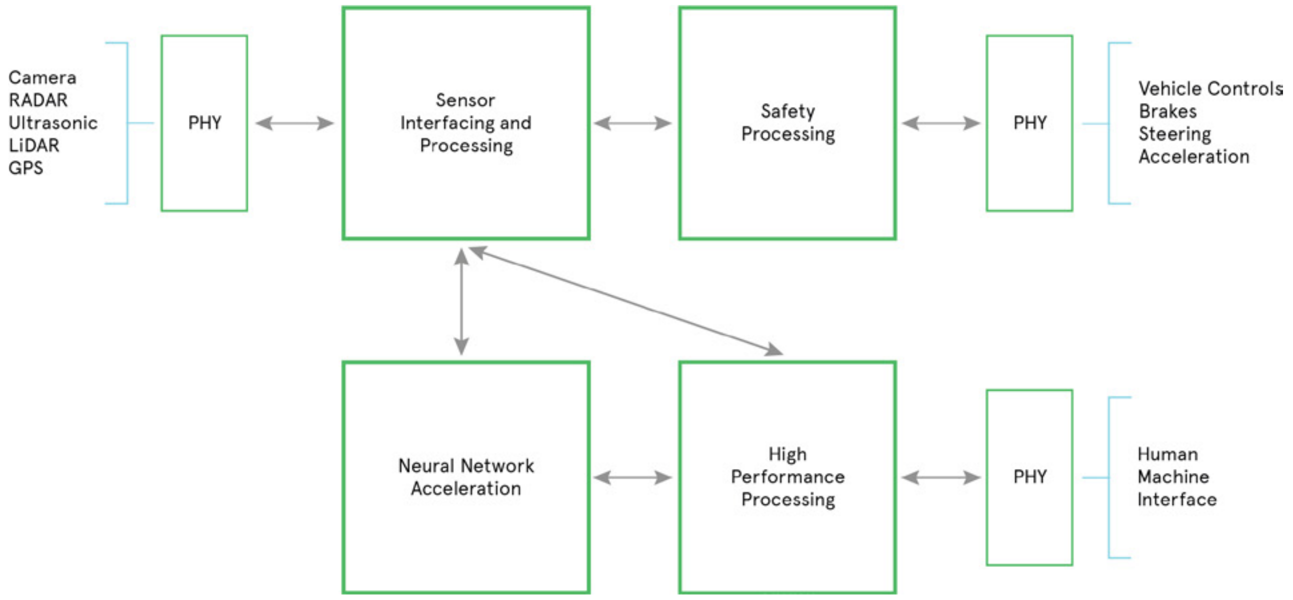


Figure 2. Devices connected using automotive Ethernet connections.

ADDRESSING THE CHALLENGES

XA Zynq UltraScale+ MPSoCs provides not only sensor interface and processing but also high performance, safety processing and neural network acceleration within the same silicon, and can scale to support a multi-device implementation as the SAE level increases. This highly integrated and scalable solution improves the SWaP-C significantly.

For real-time control, the Zynq UltraScale+ MPSoC also provides:

1. A real-time processing unit (RPU) that contains lockstep dual Arm Cortex-R5 processors capable of implementing safety features up to ASIL C and intended for safety-critical applications. To provide the necessary functional safety, the RPU was designed with the ability to reduce, detect and mitigate single random failures, including both hardware and single-event induced. These devices enable efficient segmentation of the functionality between the processor system resources and/or programmable logic.

   One key challenge presented by the sensor interface and processing is being able to interface with a range of high-bandwidth sensor modalities, all of which come with different interface standards.

   A typical solution will interface with a range of sensor modalities that use high-speed interfaces such as MIPI, JESD204B, LVDS and GigE for high-bandwidth interfaces like cameras, RADAR and LiDAR. The sensor interface and processing must also interface with slower interfaces such as CAN, SPI, I2C and UARTs.

2. Implementing the safety processors presents another challenge. The safety processors must act on commands received from the DAPD and high-performance serial processing that enable the safe navigation of the vehicle. The safety processors interact directly with the vehicle controls, such as steering, acceleration and braking — a critical aspect of autonomous driving as errors here can result in the loss of life or damage to the environment. The Xilinx automotive-grade Zynq UltraScale+ MPSoC contains dual lockstep Arm Cortex-R5 cores within the RPU can be used to implement the safety processing.

   Along with the lockstep capability of the RPU cores, additional mitigation provisions in the Xilinx Zynq UltraScale+ MPSoC include the introduction of Error Correction Codes for the RPU, tightly coupled memories and the caches, while the DDR memory is protected with Double Error Detection and Single Error Correction codes.

3. Support for a range of industry-standard interfaces, including CAN, SPI, I2C, UART and GigE, while the flexibility of the PL I/O enables direct interfacing with MIPI, LVDS and gigabit serial links, allowing higher levels of the protocol to be implemented within the PL, often using IP cores. Implementation of the protocol within the PL also enables standards revisions to be easily incorporated, plus flexibility as to the number of specific sensor interfaces supported within a solution. The PL also gives the ability to implement any interface with the provision of the correct PHY in the hardware design, delivering a true any-to-any interfacing capability.

4. Acceleration of neural networks using the Xilinx DNNDK and Vitis AI. Algorithms can also be accelerated using Vitis, which enables OpenCL kernels to be deployed in the programmable logic.

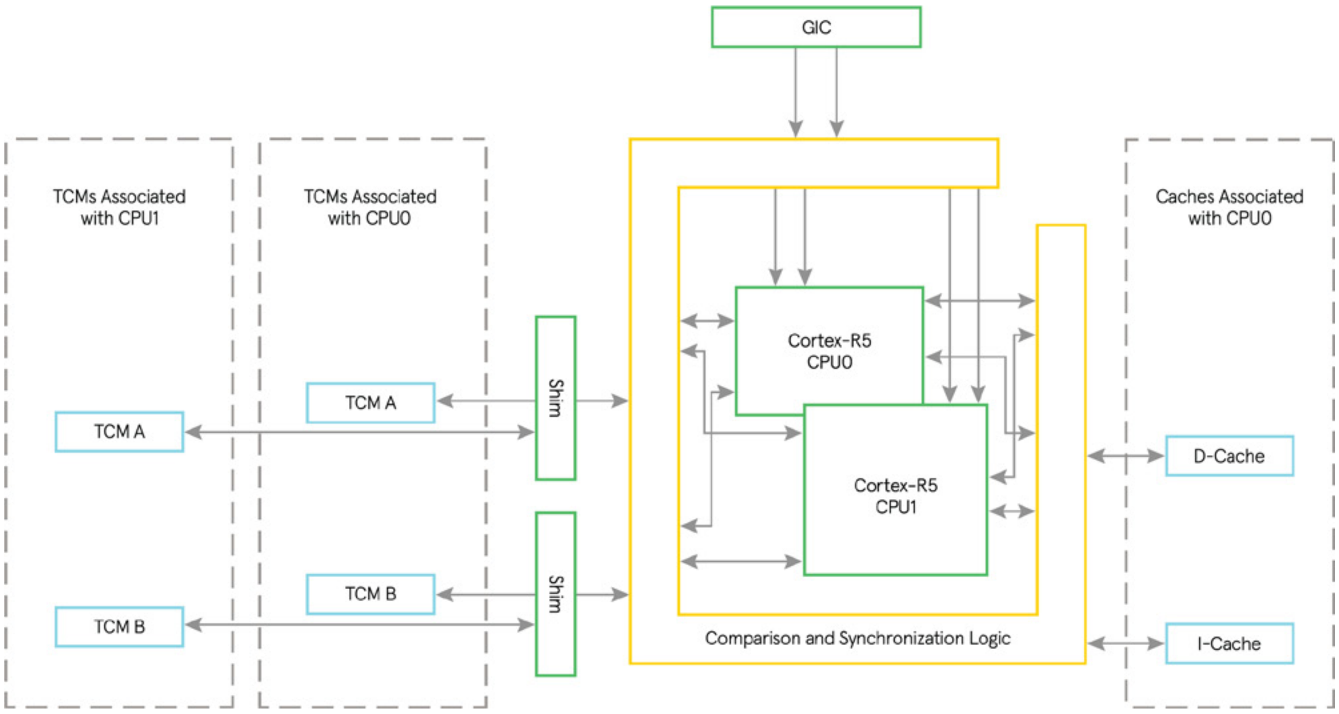

GIC
TCMs Associated with CPU1
TCMs Associated with CPU0
Caches Associated with CPU0
TCM A
TCM B
TCM A
TCM B
Shim
Shim
Cortex-R5 CPU0
Cortex-R5 CPU1
D-Cache
I-Cache
Comparison and Synchronization Logic

Figure 3. Real-Time Processing Unit Architecture

5. Built-in self-test (BIST) during power-on will ensure the underlying hardware has no faults prior to operation. Additional BIST can also be executed at the request of the user during operation. The architecture of the Zynq UltraScale+ MPSoC also provides the ability to implement functional isolation of memory and peripherals within the device.

6. Significant security features, including:
    - Secure configuration for privacy (AES encryption)
    - Key rolling to prevent differential power analysis
    - Configuration security unit to manage the security state of the devices

The inclusion of these facilities within the Xilinx automotive-grade Zynq UltraScale+ MPSoC enables safety processing to be implemented within the same silicon as the Sensor Interface and Processing, High Performance Processing and Neural Network Acceleration. It also provides a scalable solution should the architecture require segmentation across several devices.

Of course, higher integration also reduces the complexity of the PCB design and interconnect required in the final solution while offering a lower power solution.

SUMMARY

The provision of autonomous driving capability requires the implementation of a centralized processing module that faces several challenges, including interfacing with diverse sensor modalities and traditional SWaP-C issues. A highly integrated solution based on using the Xilinx automotive-grade Zynq UltraScale+ MPSoC for the Sensor Interfacing, Neural Network Accelerator and Safety Processor enables the creation of a smaller, lower-weight and more power-efficient solution.

AUTOMOTIVE FUNCTIONAL SAFETY

INTRODUCTION

One of the significant challenges automotive systems manufacturers face is designing a system that complies with relevant safety and quality standards.

For functional safety in the automotive world, this means compliance with one of the ISO 26262 Automotive Safety Integrity Levels (ASIL). As a risk-based safety standard, ISO 26262 applies to electric and/or electronic systems in production vehicles and can span driver assistance, propulsion, and vehicle dynamics control systems. Four ASILs indicate the level of hazard of the system.

ASIL	Criticality	Failure In TIme	Metric	Consequence
ASIL-D	Most Critical	< 10 FIT	Single Point Failure Metric > 99% Latent Fault Metric > 90%	Possible Fatalities in Community
ASIL-C		< 100 FIT	Single Point Failure Metric > 97% Latent Fault Metric > 80%	Possible Fatalities
ASIL-B		< 100 FIT	Single Point Failure Metric > 90% Latent Fault Metric > 60%	Possible Major Injuries or a Fatality
ASIL-A	Least Critical	< 1000 FIT	Single Point Failure Metric < 90% Latent Fault Metric < 60%	Potential for Minor Injuries
QM	Quality Managed	NA	Not Safety Related	

Figure 1. Automotive Safety Integrity Levels

The Single Point Failure Metric is the ability of the safety design to prevent risk from single point failures, whereas the latent fault metric is multiple point failures not detected by the safety design.

The ASIL level for a system is determined by performing a hazard analysis, which addresses:

- Severity — What are the potential injuries to the occupants?
- Exposure — How often is the system/vehicle exposed to the hazard?
- Controllability — What action can the driver/vehicle take to prevent the injury?

Each system in a vehicle will have a different ASIL rating. For example, critical systems such as braking, steering and airbags are typically ASIL-D, while headlights and ADAS systems are typically ASIL-B.

Creating a design that achieves the required ASIL requires significant design analysis and failure-mode consideration to be able to detect and mitigate single point and latent failures.

APPROACHES AND CHALLENGES

Achieving the desired ASIL requires consideration at the architectural design level, with the necessary mitigation and monitoring elements included from the beginning. Adding such mitigation and protection later in the development cycle can have significant impact on the project's schedule and cost.

One major element of the design is removing as many single point failures as possible from the design. At the architectural level, this may mean the implementation of hardware fault tolerance or diverse circuit elements of the design.

Typical single points of failure within an electronic system include input and outputs, clocks, power supplies, reset and power sequencing and configuration memories. Failures in these areas can result in failures of the overall system if the failure is able to propagate and affect the hardware fault tolerance or diverse path.

Single Point Failure	Mitigation
Input / Output	Duplication
Clock	Multiple Clocks
Power Supplies	Independent Supplies
Reset and Sequencing	Separate Reset and Sequencing System
Configuration Memories	Error Detection and Correction
Processors	Lockstep Operation
Logic Design	Triple Modular Redundancy

Figure 2. Single point failures and mitigations

Designers also need to consider the impact of failure propagation within the design —techniques like physical isolation are used to prevent one failure from affecting the mitigating circuit element.

Achieving the required ASIL needs to consider more than just the design of the circuit that must be analyzed and verified. Also important are the design tools used to create solutions for processors, FPGA and other programmable devices, which should be considered certified, if necessary. Such consideration is critical as errors within the tool could introduce or fail to detect errors in the design.

In the ISO 26262 world, the need to certify an Electronic Design Automation (EDA) tool or not is indicated by its Tool Confidence Level. To determine the tool confidence level, the following parameters must be analyzed for each EDA tool used:

- whether a malfunctioning software tool and its erroneous output can lead to the violation of any safety requirement allocated to the safety-related software to be developed
- the probability of preventing or detecting such errors in the output of the tool

To help determine the tool confidence level we can using Tool Impact (TI) and the Tool Error Detection (TD). Use a TI of one unless an error cannot be introduced by the tool, in which case a TI of zero should be used.

Tool Error Detection	Degree of Confidence
TD4	No
TD3	Low
TD2	Medium
TD1	High

Figure 3. Tool error detection and degrees of confidence

THE TI AND TD ARE USED TO DETERMINE THE TOOL CONFIDENCE LEVEL, WHICH INDICATES IF THE TOOL NEEDS TO BE CERTIFIED.

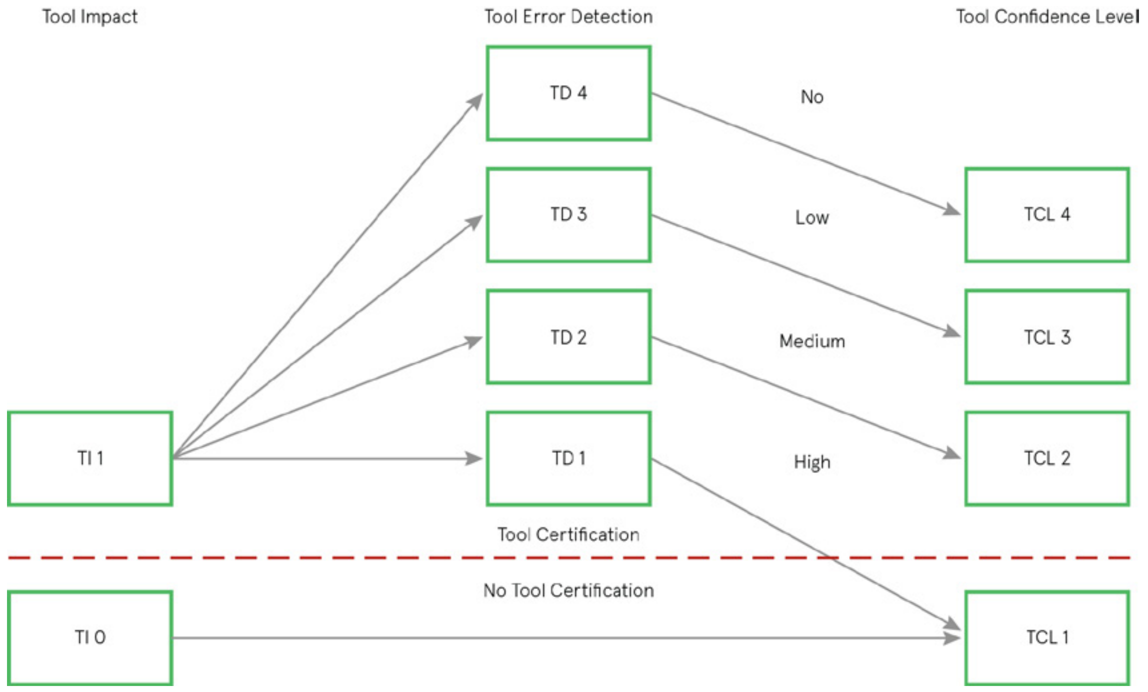


Figure 4. Determining tool confidence level

Of course, device selection, architectural design, design implementation and tool chain selection for an ASIL solution becomes challenging when a FPGA or SoC is used in the system.

Thanks to their high-performance, low-latency, increased determinism and power efficiency, FPGAs and SoCs are popular in many automotive systems including RADAR, vision-based systems, vehicle communication and ADAS.

ADDRESSING THE CHALLENGES

Many of the challenges faced by designers of automotive systems that must meet ASILs can be addressed by the Xilinx Automotive XA Artix-7 FGPA, XA Spartan-7 FPGA, XA Zynq-7000 SoC and XA Zynq UltraScale+ MPSoC families of devices, and the associated tool chains.

Selection of the appropriate device depends on the end application, interfacing, processing capability and resource requirements. However, all the devices with the XA portfolio are tested in accordance with AEC-Q100 for part approval. This increased component quality helps with the overall reliability, which supports the functional safety of the system.

XILINX PROVIDES SEVERAL INTELLECTUAL PROPERTY CORES, DEVICE ARCHITECTURE AND DOCUMENTATION SUPPORT TO HELP ACHIEVE THE DESIRED FUNCTIONAL SAFETY LEVEL.

Within the programmable logic, low-level support for reliable logic design is provided, including error detection and correction on Block RAMs. The provision for Triple Modular Redundant (TMR) or Lockstep MicroBlaze soft core processors is often deployed in the programmable logic to provide a diverse implementation. The XADC and Sysmon can be used to monitor device supply rails and die temperature with alarms raised if limits are exceeded.

Isolation between functions within the programmable logic can be implemented using Xilinx Isolation Design Flow and Vivado Isolation Verfier.

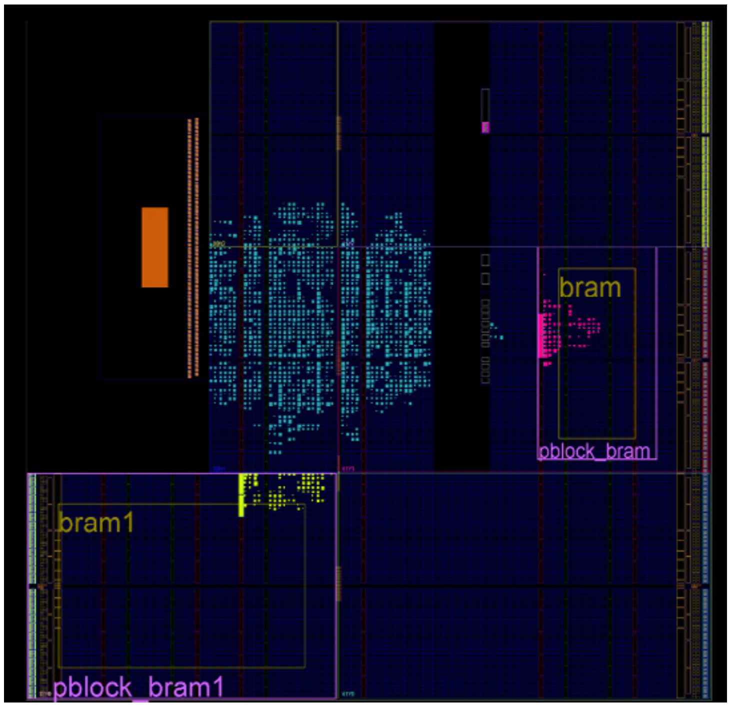


Figure 5. Isolation of BRAM and BRAM controllers

To protect the programmable logic configuration essential bits technology is provided which reports the number of critical configuration bits within the configuration stream. This can be used with the Single Event Mitigation IP core to monitor and correct errors within the device configuration memory.

The XA Zynq UltraScale+ MPSoC device architecture and safety manual is certified against ISO 26262:2011 to ASIL-C. Architectural features that enable the implementation of ASIL-C solutions include:

- Three independent domains in the low power, full power and programmable logic, each with its own power supply and clocking for hardware fault tolerance
- TMR boot safety, power and error management processor in the platform management unit
- Lockstep Arm Cortex-R5 processors within the low power domain
- Memory and peripheral protection units enable isolation within the processing system
- Error correction codes on critical memories
- Testable architecture including logic and memory built in self-test, error injection and software test libraries

One example implementation of an ASIL-C application based in a XA Zynq UltraScale+ MPSoC using a hardware fault tolerance of one: the implementation of two safety channels, one located within the low power domain and the other in the programmable logic. Each safety channel would interface to a sensor, and the results of the processing can be compared and voted on. Such an approach enables a diverse fault-tolerant hardware solution as shown in Figure 6.

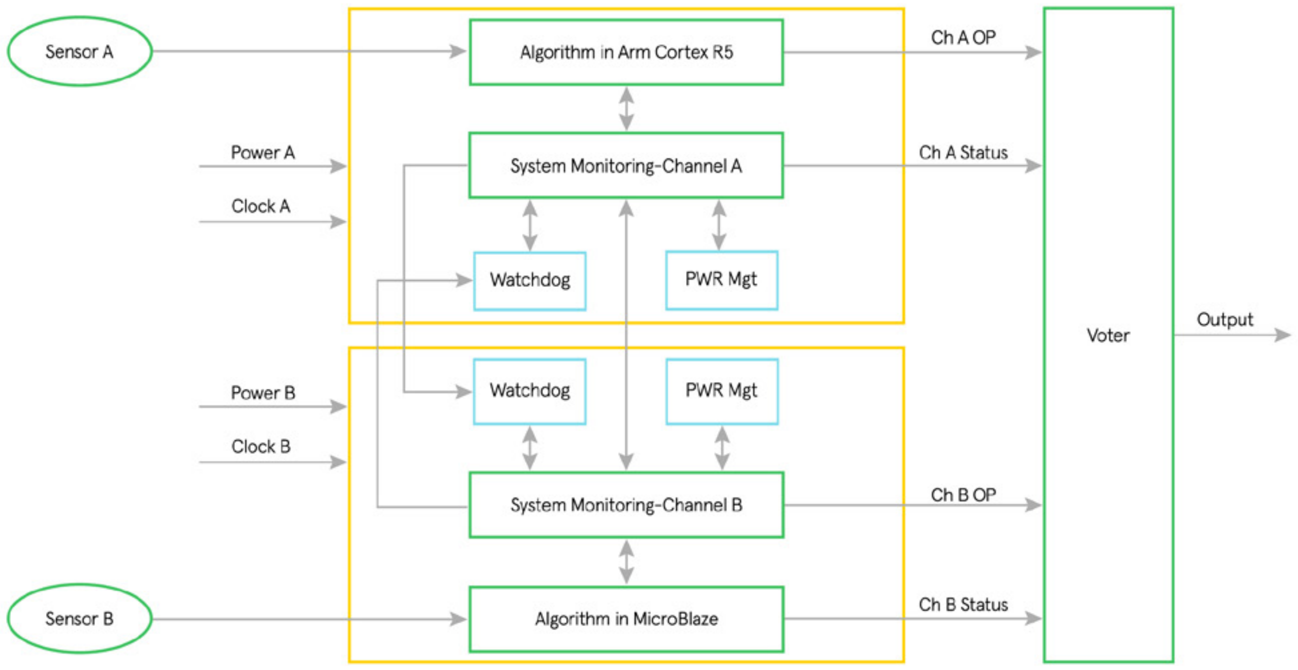


Figure 6. Diverse and fault-tolerant hardware solution

Supporting Xilinx Automotive devices are tools certified to ISO 26262-8:2011.

Certified tools enable developers to reduce overall development time — helping to achieve high-quality delivery of the project on time and cost effectively.

Several Xilinx system and analysis tools are designed to help achieve certification, including safety manuals, software safety user guides, a FMEDA tool and examples along with regular reliability reports. These resources are available via the Xilinx Functional Safety Package.

SUMMARY

Achieving the required safety standards defined by ISO 26262 requires consideration of not only the overall system architecture but the capabilities and qualifications of the implementation technology. Xilinx Automotive devices are qualified to AEC-Q100 quality standards and designed with ISO 26262 certification in mind, as demonstrated by the example applications and certified tool chains. This enables confident development of ISO 26262 solutions using Xilinx automotive components.

OCCUPANT MONITORING

INTRODUCTION

Providing a safe automotive experience requires that the vehicle not only monitors and observes its external environment, but that it is also able to monitor its internal cabin space, and does it with latency in mind. Such monitoring can improve safety — for example, by monitoring driver alertness or alternatively adjusting the cabin to be comfortable for the occupants.

Processing the video and voice commands and implementing control algorithms and feedback mechanisms takes considerable processing capability to be able to implement functions in near real time. A slow response will lead to an unfulfilling interaction and potential frustration in the case of gesture recognition and may even cause a potential safety event if the system does not react to a tired driver.

ARCHITECTURE FEATURES AND CONSIDERATIONS

The most critical of the occupant monitoring functions performed by the internal monitoring system is monitoring the driver for alertness and drowsiness.

A monitoring algorithm like this requires an image processing pipeline to connect to the camera and receive images. Images from the camera need further processing to detect the driver’s face and eyes before implementing blink detection.

Typical processing stages will include color space conversion from RGB to grayscale, and histogram equalization before the face, eye and blink detection implementations. Blink detection must be customizable to allow for different response times (the typical human blink response is 400 ms).

CONVOLUTIONAL NEURAL NETWORKS OFFER THE ABILITY TO DETECT BOTH THE FACE AND EYES.

Wider field-of-view cameras are used to monitor other occupants and their behaviors in the cabin. Further processing of that video supports gesture detection and recognition from the occupants. Detection of these gestures with the video frames requires the use of convolutional neural networks.

INTERNAL MONITORING

Implementing internal monitoring requires a diverse range of sensors, from cameras focused on the driver to wider-angle cameras that enable the entire cabin and everyone in it to be observed. Cabin settings may be determined by detection of the occupants or the occupants’ gestures or commands. Additional sensors may be required, such as temperature sensors for comfort, directional microphones for voice commands, touch sensors and haptic feedback devices on controls.

The system needs to be able to interface with and process several directional microphones.

The occupant monitoring system may also use voice commands to control systems within the cabin. Advanced systems may enable the user to request information and status, such as time to destination or current traffic conditions. While basic voice processing can be implemented within the solution, advanced features such as traffic awareness would require a connection to the cloud.

Following the interpretation of gesture or voice commands, the system must be able to interface with controls to alert the driver in case of drowsiness and adjust climate settings or seat positions depending upon personal recognition. This requires the ability to drive motors, sensors and haptic feedback devices, all within a near real-time response.

The status of the cabin, its controls and its occupants may also be reflected by a high-resolution touch display. This enables both status reporting and configuration of gesture controls, along with the ability for backup setting of all controls.

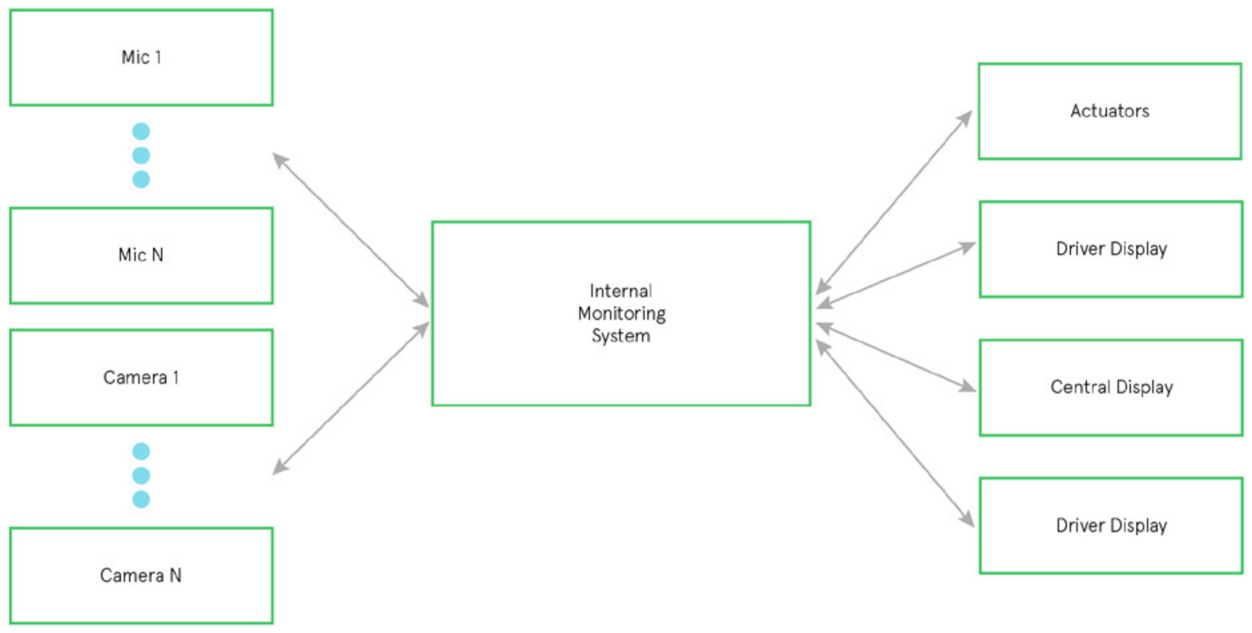


Figure 1. Occupant monitoring system

ADDRESSING THE CHALLENGES

The Xilinx XA UltraScale+ MPSoC heterogeneous system-on-chip can help achieve the architecture within a power- and cost-efficient solution.

The tightly coupled processing system and programmable logic domains can be used to implement the occupant monitoring system, which enables a low-latency and power-efficient solution.

The programmable logic domain would implement the camera interfacing and image processing pipelines. Thanks to the flexibility of programmable logic, several camera interface types can easily be supported, including MIPI, CameraLink, VGA and bespoke.

These image-processing pipelines can be implemented in parallel, reducing the overall processing latency. At the completion of the image-processing pipeline, the result can be transferred to the processor domain for the higher-level neural network application. Creation of the image-processing pipeline within the programmable logic can leverage the large range of Xilinx and third-party IP cores in the Vivado Design Suite library. If custom IP is required, the designer can create this using either a hardware description language, a high-level synthesis tool or high-level development tools like Xilinx System Generator or Model Composer tools leveraging Simulink. To enable high-level algorithmic models created in frameworks such as OpenCV, Xilinx provides the xfOpenCV library. The xfOpenCV library contains several commonly used OpenCV functions that can be synthesized into the programmable logic using High-Level Synthesis in the Xilinx Unified Software Platform Vitis. This enables high-level modelling using OpenCV — and then, quickly and easily, the same functions can be implemented within the programmable logic pipeline without the need to write a line of hardware description language code.

Implementation of the neural network can leverage Vitis AI. Vitis AI enables the acceleration of commonly used ML/AI frameworks, including Caffe and TensorFlow, using programmable logic.

To enable this, Vitis AI provides a Model Zoo, AI Compiler, Optimizer, Quantizer and Profiler to deploy applications to the Deep Learning Processing Unit that enable the face and eye detection networks to be used for driver detection. Additional units can also be deployed for occupants' gesture recognition.

Motor and Actuator control will often require high-level drive circuits, acting under the control of the MPSoC. As such we can use the Triple Timer Counter within the processing system to generate pulse width modulation waveforms used to drive motor H bridges, allowing the speed and direction of the motor to be controlled. A similar approach can be implemented if servo motors are to be controlled without the H bridge.

To provide the user displays, lower levels of the video output can be implemented using the DisplayPort output available within the processing system. If the display interface requires HDMI or lower level, such as LVDS or Parallel video, this output stream can be implemented within the programmable logic. In both cases, video can be generated by the processor or fed live from the programmable logic.

It is common for applications such as this to run a high-level operating system like embedded Linux. To be able to create embedded Linux solutions that adapt to the design implemented in the programmable logic design, Xilinx provides an embedded Linux build system in PetaLinux. PetaLinux simplifies the Yocto build flow to enable the developer to customize the desired application, drivers and frameworks.

Running PetaLinux on the APU cores enables the deployment of Vitis AI in conjunction with the Deep Neural Network Processor Unit (DPU) in the programmable logic. The APU can also be used to create the user display — frameworks such as Qt can be used to generate a professional-looking display. This interface will also be able to receive commands from touch-screen positions along with displaying high-resolution graphics.

Implementation of real-time control required for vibration or haptic feedback can be provided by the RPU in the processing system. Communication between the APU and RPU can be achieved using the OpenAMP framework.

LEVERAGING THE INTERFACE

Designers can leverage the interfacing provided by both the processor system and programmable logic. The processing system can interface with several sensors that use common embedded interfaces, e.g., I2C, SPI and UART, and the programmable logic can be used to interface with the directional microphones that use the I2S interface.

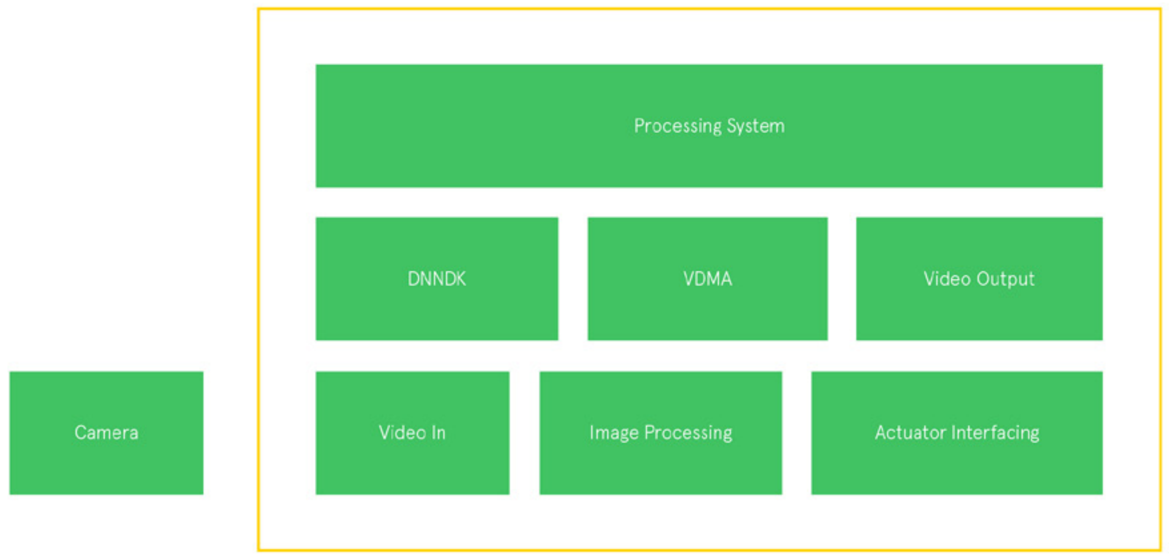


Figure 2. OpenAMP framework

SUMMARY

Monitoring the occupants of a vehicle enables not only safer vehicle interaction but also creates a more comfortable occupant experience. A power- and cost-efficient solution can be implemented using Xilinx Automotive XA Zynq UltraScale+ MPSoC devices, along with the supporting development and software ecosystem that enables the use of commonly used frameworks such as OpenCV, TensorFlow and Caffe.

VEHICLE TO EVERYTHING (V2X) COMMUNICATION

INTRODUCTION

The ability for vehicles to communicate between each other, with other road users and with infrastructure or roadside equipment such as signs, pedestrian crossings and traffic lights is thought to be one of the cornerstones of improving road safety.

These Intelligent Transport Communications (ITC) systems are short range. Due to their safety-critical nature, they must be low latency, capable of communicating events such as collisions or traffic lights changing quickly.

Currently two standards are used in vehicles to provide this communication standard. First is Dedicated Short Range Communication (DSRC), which is Wi-Fi based and uses the IEEE 802.11 standard. The other standard is based on the emerging 5G mobile communication standard, of which a large element of 5G usage is expected to be reliable low-latency communication.

Both communication standards have advantages and disadvantages. DSRC offers free access at the expense of lower data rates, while 5G solutions require a subscription but offer significantly faster and lower-latency communication.

BESIDE THE ACCESS COST AND DATA RATE, ONE MAIN DIFFERENCE IS THE RANGE OF COMMUNICATION. DSRC WILL HAVE A RANGE OF 300 METERS WHILE 5G WILL OFFER AN INCREASED RANGE.

As both solutions provide a path into a vehicle's systems — which impact the functionality and ultimately the safety of the vehicle and its occupants — the environment security and cyber security of the solution is critical.

SYSTEM MANUFACTURERS

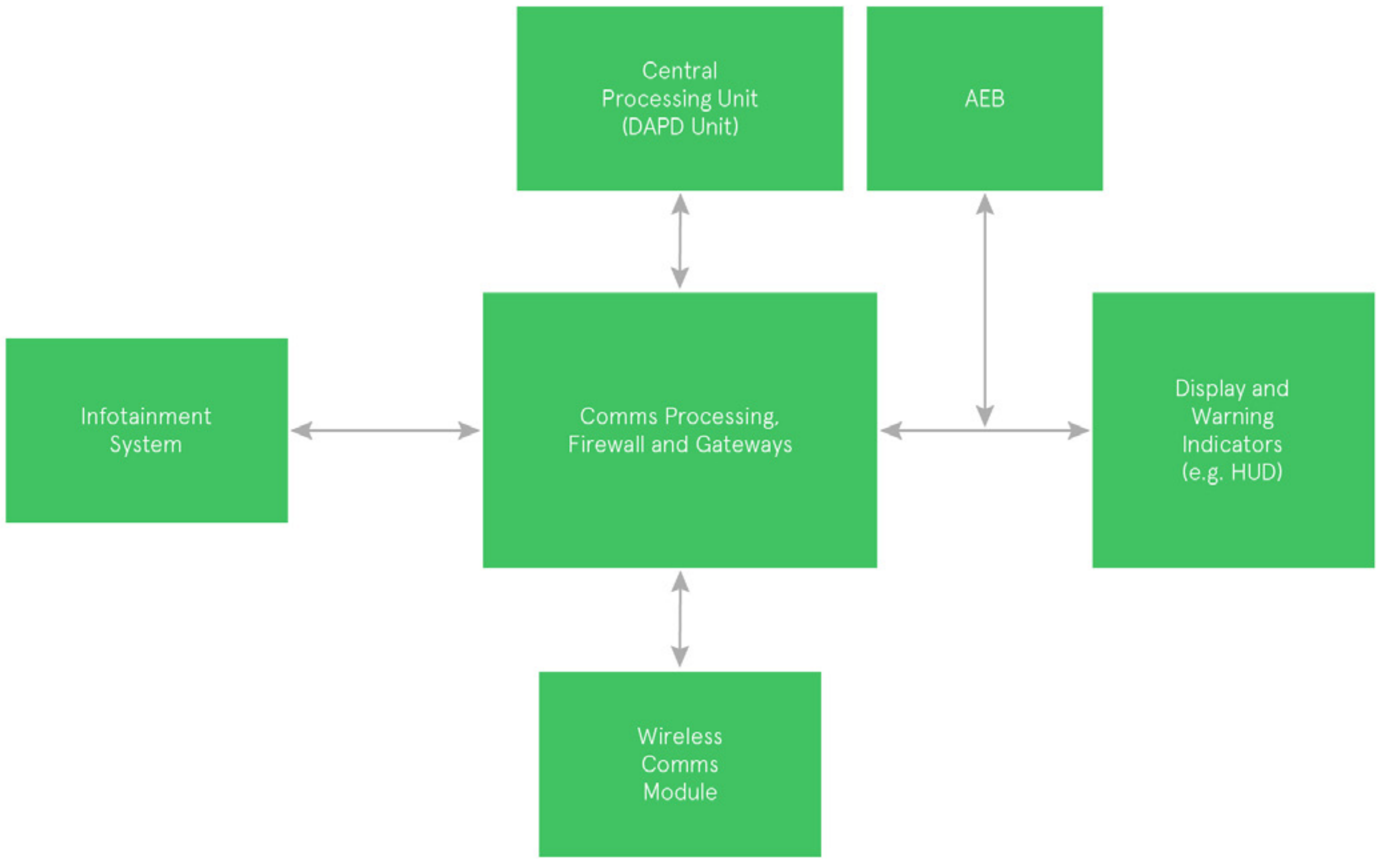
Both DSRC and 5G solutions provide several challenges, including:

- Meeting unit costs of circa $350 USD per vehicle in 2020 (expected to decrease to $200 USD by 2058)
- The ability to adapt to different frequency allocations in different geographic locations
- Achieving the demanding form factors and power budgets required in automotive applications
- As a critical system, achieving not only the required performance but also the reliability target
- The ability to adapt as standardization is performed across regions – they may need to support both DSRC and 5G solutions if deployed in complementary use cases
- Suitability for fitting to both new build and existing vehicles

ARCHITECTURE FEATURES AND CONSIDERATIONS

The architecture of a typical V2X module will consist of several different functional elements including:

- Communication Processing Module (CPM), the central processing hub of the system, which interfaces with one or more RF front ends to provide DSRC and 5G communication stacks. Along with implementing the communication stack, the CPM will connect into the existing automotive vehicle network including CAN, CAN-FD, FlexRay and Automotive Ethernet. One key role of the CPM is implementing physical and cyber security policies using gateways and firewalls, along with authentication and security techniques to prevent unauthorized access and ensure data security.
- Wireless Communication Module (WCM) connected to the CPM to implement the RF front-end and lower elements of the communications standards
- Interfacing, the ability to interface over existing automotive standards, e.g., CAN, CAN-FD, Automotive Ethernet, FlexRay to communicate with the infotainment system, Heads Up Display and Central Processing Units used for autonomous functions.


Central Processing Unit (DAPD Unit)
AEB
Infotainment System
Comms Processing, Firewall and Gateways
Display and Warning Indicators (e.g. HUD)
Wireless Comms Module

XA Zynq UltraScale+ MPSoC devices provide the capability to address the functional, security and power requirements with ease.

ADDRESSING THE CHALLENGES

Zynq-7000 SoCs and Zynq UltraScale+ MPSoCs can be as the central hub of the communications processing module.

This combination of a processing system and programmable logic enables a wide range of interfacing solutions with the external RF module. The processing system supports standards such as SDIO, which is commonly used for Wi-Fi interfacing, while the programmable logic enables any-to-any interfacing with the right external PHY. This is needed for 5G RF modem/module support.

The processing system and programmable logic also provide the ability to connect to existing vehicle networks using a diverse range of interface standards, including those identified in the following table. This table outlines the main interfaces supported by the processing system or the availability of IP cores for integration within the programmable logic.

These interfaces enable the CPM to be able to connect with the infotainment system, heads up display and the vehicle's central processing unit.

Interface Standard	Processing System	Programmable Logic
CAN-FD		x
CAN	x	
Local Interconnect Network (using UART)	x	
Gigabit Ethernet	x	
Automotive Ethernet		x
I2C	x	
SPI	x	
SDIO	x	
Time-Sensitive Ethernet		x
DisplayPort	x	
HDMI Out		x

Table 1. XA Zynq UltraScale+ MPSoC interfacing support

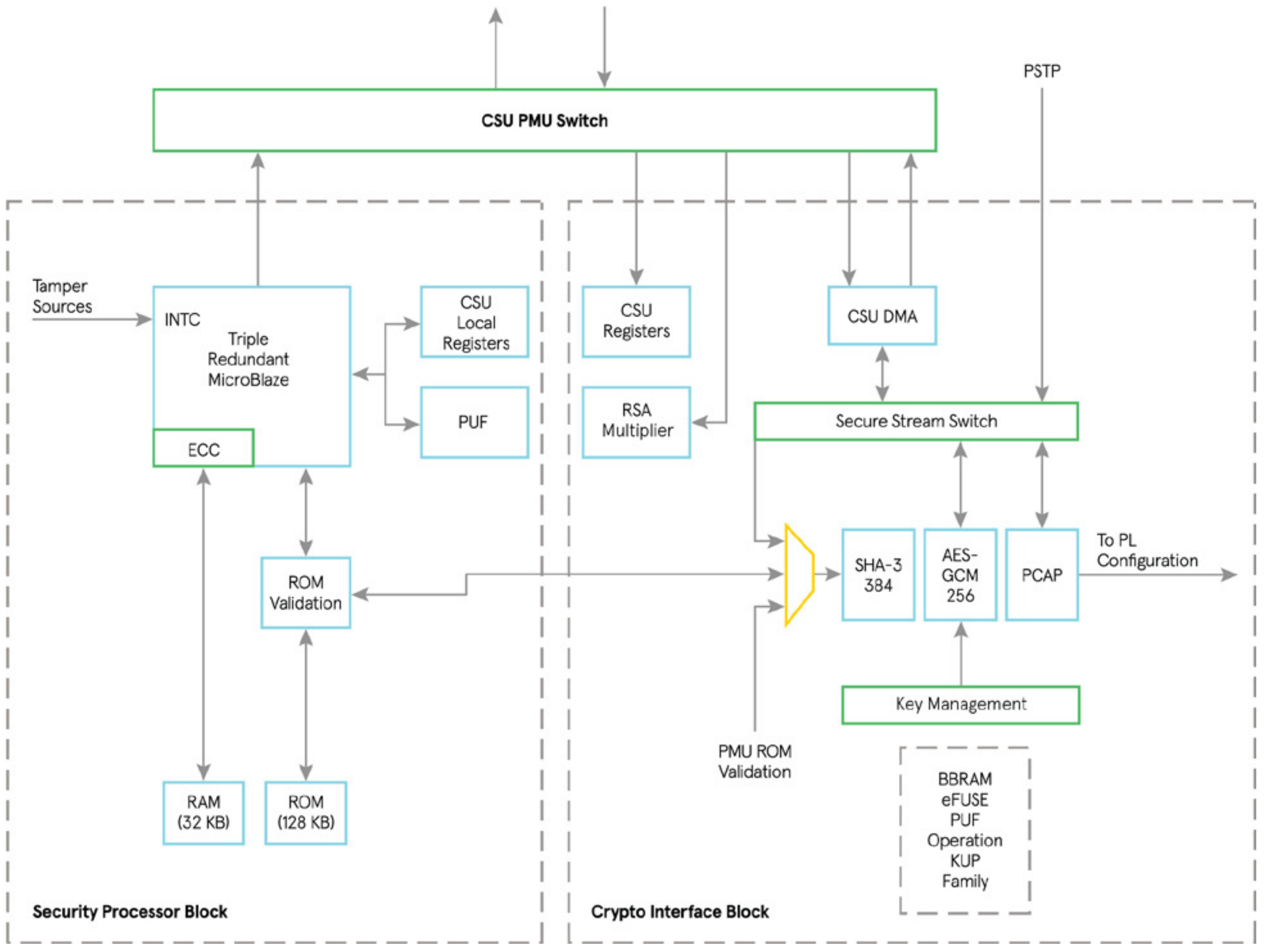
The algorithms, gateways and firewalls required for implementation of the safe communications can be implemented in both the processing system and the programmable logic. With implementation in the programmable logic comes not only flexibility but also increases in determinism and reduction in latency and throughput. Use of the programmable logic also enables more secure implementation of the firewalls and gateways than traditional software-based approaches. To support the implementation of firewalls and gateways, the processing system provides memory and peripheral protection thanks to the XMPU and XPPU, which enable apertures to be created with peripherals and memory regions to be created and isolated from each other.

IN ADDITION, PHYSICAL SEPARATION CAN BE IMPLEMENTED WITHIN THE PROGRAMMABLE LOGIC USING THE XILINX ISOLATION FLOW.

Further enforcing the security solution is the ability of the CSU within the processing system to create a layered security implementation. This enables a secure configuration of the device thanks to the CSU, which supports AES 256-GCM, 4096 RSA Multiplier, and SHA-384 providing confidentiality, authentication and integrity of the solution.

Run-time security support includes the provision of anti-tamper response thanks to the built-in system monitor which enables device voltages and die temperature to be monitored with alarms raised if limits are exceeded. The system monitor also has external connections that can be used for more physical anti-tamper protection at the enclosure level.

The CSU encryption engines AES, RSA and SHA blocks can also be leveraged at runtime to implement confidentiality, authentication and integrity functions. Because of the DMA available within the CSU, this can provide for very efficient processing. The CSU is also capable of implementing key management, including the provision for key rolling to protect against differential power analysis key attacks.



The processing system is also able to implement Arm Trust Zone enabling orthogonal secure and non-secure worlds in both the processing system and the programmable logic.

When it comes to developing algorithms for the communication module, the processing system in a device can be leveraged to implement the communication stack while the programmable logic can be used to offload acceleration features in addition to the firewalls and gateways, including the ability to generate audio and video streams for the infotainment system. This is especially true of the EV variant of the XA Zynq UltraScale+ MPSoC device which includes a H.264 / H.265 Video Codec which can be used for both encoding and decoding video streams.

The overall software solution for the CPM can be implemented using the unified software development platform Vitis. Vitis enables software to be developed as a complete system for the APU, RPU and PMU.

Within this system solution, designers can utilize embedded operating systems such as embedded Linux on the APU or real-time operating systems such as FreeRTOS to create the overall software solution. To support the multiprocessor environment the OpenAMP can be used to enable safe communication between the APU and RPU along with inter-processor interrupts (IPI) and support for short message buffers associated with each IPI.

In addition to the system development capabilities provided by Vitis, Vitis also providing support for acceleration using OpenCL.

SUMMARY

Standards are still uncertain in several regions. Varying standards will mean solutions need to be able to adapt. A SoC with programmable logic enables this flexibility to be addressed by the solution provider while also providing the functionality, security and interfacing capabilities to address the SWaP-C challenges needed for V2X applications.

OpenCL ENABLED

OpenCL enables acceleration kernels to be deployed within the programmable logic under the control of the APU. This means the acceleration of algorithms does not necessarily need to be implemented by engineers' Hardware Description Languages (HDL), instead High Level Synthesis based on C / C++ can be used to program the algorithm. Performance optimizations can then be implemented in the algorithm using pragmas to leverage the parallel architecture of programmable logic.

LEARN MORE ABOUT AVNET AT WWW.AVNET.COM

Avnet
2211 S. 47th Street
Phoenix, AZ 85034
1-800-332-8638
avnet.com