Roadmap (this includes objectives and ideas)

1. Understand the board better.
    a. What goes on in the board.
    datasheet of the board
    b. How do you program the board
    c. Specifications.
2. Understand the objective of the project.
    a. What standards will we use to compare the multicore and FPGA (metrics).
    Highlighted *parts are subjective, numeric metric might not be available.*
    Use FPGA-specific performance modeling tools provided by the FPGA vendor to estimate resource utilization, throughput, and latency for different model configurations.
    Performance:
        Throughput: Measure of how much work can be accomplished in a given amount of time.
        Latency: The time it takes for a system to respond to a stimulus or complete a task.
        Speedup: Ratio of the time taken to execute a task on a single core versus multiple cores or FPGA logic.
    Power Consumption:
        Static Power: Power consumed by the device even when it is not actively processing any data.
        Dynamic Power: Power consumed while actively processing data.
        Total Power: Sum of static and dynamic power.
    Flexibility and Programmability:
        FPGA Flexibility: FPGAs can be reconfigured for different tasks, providing flexibility in applications.
        CPU Programmability: CPUs can run a wide range of software, offering flexibility in application development.
    Resource Utilization:
        FPGA Resource Usage: How efficiently FPGA resources such as logic elements, DSP blocks, and memory blocks are utilized for a given task.
        CPU Core Utilization: How efficiently CPU cores are utilized for parallel processing tasks.
    Cost:
        Initial Cost: The upfront cost of purchasing the hardware.
        Operating Cost: Cost of power consumption, maintenance, and potential licensing fees for tools or software.
    Ease of Development:
        Toolchain Complexity: Complexity of development tools required for programming and debugging.
        Programming Model: Ease of programming for both FPGAs (using languages like Verilog or VHDL) and CPUs (using languages like C/C++).
    Parallelism and Concurrency:
        FPGA Parallelism: FPGAs inherently support massive parallelism, allowing for concurrent execution of multiple tasks.
        CPU Multithreading: CPUs with multiple cores can execute multiple threads concurrently, although they may not match the parallelism level of FPGAs.
    Application Suitability:
        Task Dependency: Certain tasks may be better suited for execution on FPGAs due to their parallelism or hardware-accelerated nature.
        Software Dependencies: Tasks that heavily rely on existing software libraries or ecosystems may be better suited for execution on CPUs.
    Development Time:
        Time-to-Market: How quickly a solution can be developed and deployed using either FPGAs or multicore CPUs.
        Design Complexity: Complexity of the design process for implementing a solution on FPGAs versus coding and optimizing software for multicore CPUs.
    Scalability:
        FPGA Scalability: Ability to scale up FPGA resources for larger and more complex designs.
        CPU Scalability: Ability to scale up the number of CPU cores in a system for increased parallel processing power.
    b. What is the number of models and what kind of application to be used.
    Models supported by vitis AI
    ykpgrr/Lane-Detection-with-Implementation-on-FPGA
    This model uses the SDSoC and Vivado platforms to implement a lane detection algorithm on FPGA. The preprocessing part is implemented in hardware, and the Vivado platform uses low level code (Verilog) while the SDSoC platform uses high level code (C++).
    c. How many camera inputs to use. Do we only have one type of sensor?
    For Lane detection 1 forward camera is used, unlike Subaru which uses 2.

Table 1 Mounting location and function of each camera

Equipment name	Mounting location & function
Sensing camera	Installed in the upper part of the windshield, it senses a wide area to ensure the safety of the front of the vehicle.
Surround view camera	Camera units are installed on the front, back, left, and right of the body of a vehicle to sense approximately 1 m around the entire perimeter of the vehicle and ensure safety in the vicinity of the vehicle.
Driver monitoring camera	Installed near the meter cluster of a vehicle, it senses the driver's health condition to ensure safe driving.

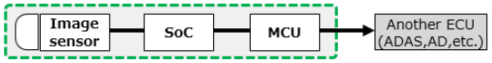
Link of the image

Sensing camera, driver monitoring camera.

An image sensor refers to a sensor that converts the light entering through the lens of a camera into electrical signals. The data sensed as signals are processed through an SoC (System on a chip). In addition, these cameras are equipped with a microcomputer to issue control commands to an external ECU.
USED FOR:
Lane Keep assist, Lane center Assist, Automatic emergency breaking

Surround view camera.
Unlike sensing cameras and driver monitoring cameras, this type of camera is equipped with multiple image sensors. This is because camera units are installed in multiple locations on the vehicle body to make up this type of camera. Acquired image data are integrated (synthesized) into one omnidirectional image by another ECU (surround view ECU, etc.).

Table 3 Internal configuration of each camera

Equipment name	Circuit configuration
Sensing camera Driver monitoring camera	
Surround view camera	

3. If the application is lane detection,
    a. What dataset?
    b. What model should run on core and fpga?
    c. How many sensors.
        1 Image Sensor. 640 x 480!!!
4. Understand and decide model specifications.
    a. Layers of model
        - Quantization and Precision Reduction:
          Use fixed point operation which reduces computation but can maintain accuracy.
          Quantize the weights and the activation function in order to reduce precision but should not drastically affect accuracy.
        - Decompose complex layers (e.g., convolutional, or fully connected layers) into smaller, simpler operations that map efficiently to FPGA hardware resources.
        - Optimize the model architecture to leverage FPGA-specific features such as block RAMs for weight storage, DSP blocks for efficient multiplication, and distributed memory for data storage.

    b. What type of model
    c. Can a heavy model run on fpga?
        Yes, needs to be altered!
    d. Can we make it take real time data?
5. Understanding the vitis software.
    a. Learning and programming with it.
    b. How it works
        i. Converting c code to HDL to fpga deployment.