VIBGYOR ePress

International Journal of
Robotic Engineering

ISSN: 2631-5106

Comparative Advantages of System on Chips in Intelligent Traffic System

***Mohammad Mujtahid*[1,2*], *Shabir A Sofi*[1] and *Tariq Bashir*[1]**

[1]*Department of Information Technology, National Institute of Technology, India*

[2]*Department of Automatic Control and Robotics, Ecole Centrale de Nantes, France*

Abstract

Automation technology is the backbone of Information and Communication Technology (ICT) programs, aimed at building smarter, better and safer niches. With Internet of Things (IoT) and embedded computing, automation is already doing wonders in providing for the assistance of human kind. With more advances in the field of automation the microcontroller aspect is now being overtaken by system on chips (SoCs) or Programmable System on Chips (PSoCs). SOCs break beyond the limitations of microcontrollers by providing for advanced peripherals, higher processing power, GPUs, Wi-Fi modules or Coprocessors etc. In this paper we aim at analyzing the impact of SOCs in automation of Intelligent Traffic System (ITS). ITS being one of the most dynamic parts of ICT program spreads over a vast technological expanse. So its study provides an insight of almost all required aspects of ICT. In this paper we analyze why SoCs are more suitable for ITS compared to Single Board Computers (SBCs) and microcontrollers. We base this analysis on the implementation of Raspberry Pi 2 Model B for development of ITS, which includes, augmented vehicle system and intelligent driving platform.

Keywords

System on chip, Intelligent traffic system, Automation

Introduction

With overcrowded roads and traffic jams being one of the primary worries of big cities, ICT programs lay much emphasis on ITS. It utilizes the computer resource and communication networks to improve vehicles, roads and traffic conditions. Efforts for developing automated systems in the field of transportation started back in the 1960s. Back then enthusiasts of artificial intelligence (AI) began dreaming of cars smart enough to navigate ordinary streets on their own. Pioneering steps towards the accomplishment of this dream started with the advent of programs like DARPA Grand Challenges. Currently GOOGLE and UBER programs are the most visible efforts in this field [1].

With sensing, processing and reacting being the defining automation components of ITS, processors are needed to accept sensed data, process it and provide appropriate commands for reacting. This job of sensing, processing and reacting for a long time has been quiet gracefully been handled by microcontrollers. But with the increase in number of sensing inputs the requirement for processing power increased. Also with more data sets and machine learning the implementation of automation using microcontrollers became difficult. To traverse these limiting parameters SOCs provided for an optimal alternative.

From technological perspective various efforts were made to create a computational device with high processing power and reduced size, energy usage etc. Some

***Corresponding author:** *Mohammad Mujtahid, Department of Information Technology, National Institute of Technology, Srinagar, India; Department of Automatic Control and Robotics, Ecole Centrale de Nantes, Nantes, France, E-mail: mujtahid.nit@gmail.com*

Received: December 28, 2017: **Accepted:** February 15, 2018: **Published:** February 17, 2018

Citation: Mujtahid M, Sofi SA, Bashir T (2018) Comparative Advantages of System on Chips in Intelligent Traffic System. Int J Robot Eng 3:004

Technology	Basic Paradigm	Basic Method of Improvements within Technology Paradigm
Chips on Board (COB)	Mounting of IC chips directly on PCBs	Substituting different materials to reducing interconnect delays
System in Package (SIP)	Stacked chips or packages for reduced form factor	Improving performance and power efficiency by short direct connection channels
System on Chip (SOC)	Complete system on a chip	Reducing form factor, power consumption, heat dissipation, analog mixed signal integration

Figure 1: Technology paradigm [4].

of the notable efforts in this regard are: Chips on board, system in package and also system on chip. With chips on board being simple integration of chips on a printed circuit board (PCB), system in package (SiP) is integration of various ICs in a single package and SoC is the integration of the whole system on a single chip. SiP found its application in performing almost all electronic system functions especially mobile phones and music players, whilst system on chip (SoC) revolutionized all computational aspects requiring high processing in lower space, for example, mobile phones, Internet of Things (IoT), ICT etc. [2,3].

In Figure 1 we have a tabular comparison of various technological paradigms and the basic scope of improvement for them.

Functionalities and Advances Provided by SOCS

Flexibility in interfacing, better power efficiency and ability to reconfigure hardware are some of the advantages provided by SoCs. A technological innovation to reduce the size of devices for integrated systems is what is provided by SoC architecture. In SoCs an entire system resides on a single chip. The plug and play interfaces provided by SoCs make them more convenient for system designing. Other than this, SoCs from power perspective are much efficient devices, since they are void of PCB tracks and burdening capacitances. Also these factors lead to overall improved system speed [5]. SoCs also provide interface for high current requirement peripherals viz, motors, servos, stepper motors etc. Interfacing a sensor to a microcontroller or a general purpose computer is quiet complex, whilst interfacing sensors to a SoC is a much simplified task.

SoCs have enabled us to produce much reduced footprints for wearable and smartphone applications e.g. touch sensors such as sliders and wheels can be connected directly to SoCs without using any external circuitry [5]. Also complicated systems like audio players and automation based devices are being built upon SoCs. Also added on PWM modules to SoCs make it simple for programmers to manipulate signals by varying duty cycles of a PWM without writing any software component for this task [5].

Summing up the point, SoCs have many advantages over microcontrollers and general purpose computers. Other than above mentioned properties some of the notable advantages are:

- Size: Many functions and features in much reduced package.
- Flexibility: It is hard to find any parallel to SoCs in terms of the flexibility they provide for board size, power and efficiency.
- Price Factor: SoCs provide various inbuilt functionalities and ease in many aspects like interfacing. Hence SoCs save much of effort and time, which in turn reduces developmental expenditure.
- Configurability: In SoCs a single pin can be configured to perform various tasks. This feature is very useful whilst taking into consideration PCB design.
- Ease of Hardware of access: SoCs mostly use Linux operating systems and provide easy interfaces for hardware support. Hence making life easy for programmers by providing hardware access with easy user interface.

Architecture

In this project we have developed an intelligent vehicle model which would aid in ITS. This project is SoC based. We have used Raspberry Pi for accomplishing this project. The architecture of the project has three main components: Sensing component, processing component and implementing component.

Sensing component includes ultrasonic sensor and Pi camera. While processing component includes Raspberry Pi 2 and implementing component includes Motors and L298N driven Dual H-Bridge.

Ultrasonic sensor provides the distance between vehicle and the obstacle, while Pi camera helps in color detection for traffic signal sensing. Also it helps in processing image data for providing further more inputs for automation. The image processing is performed using open CV and python.

Raspberry Pi is a small, lightweight and powerful ARM based SoC. Its amazing graphic capabilities and available HDMI video output provide it an upper hand for multimedia applications. Raspberry Pi version 2 model B, the one used in our project, has a 900 MHz quad-core ARM cortex A53 processor and 1 GB RAM. It has extended 40 pins for interfacing and is also provided with 4 USB ports and an Ethernet port. Software's relying upon multicore processing can speed up 4X, whilst for a multi-thread friendly code it can go up to 7.5X. It is a magnetic piece of technology for innovative programmers [6].

The operating system (Raspbian) used, is a free working framework based on Debian. Working framework implies the fundamental utilities and programs that run the system on which it is installed. It is designed for Raspberry Pi module. Raspbian Operating System is designed to enhance the speed of the Raspberry Pi based on the improved instruction execution speed of ARM processor CPU of Raspberry Pi [7]. The Ultrasonic sensor used is HC-SR04. This sensor usually is for distance measurement for distances from 2 - 400 cm. It has four pins: VCC, TRIG, ECHO and GND. VCC is the 5 V input power and GND is the ground pin. TRIG pin is to produce the ultrasonic pulse output, whilst ECHO receives the reflected input. From Raspberry Pi we generate a short 10 µs pulse to trigger input to start the ranging. Sensor sends out an 8 cycle 40 KHz ultrasonic burst and raises its echo line to high. Once it detects an echo, it lowers the echo line. Thus echo line is a pulse proportional to distance up to the object. The module can be triggered at a very fast rate of 20 cycles per second. We need to provide a 50 ms sleep time, in order to ensure that the ultrasonic beep has faded away and will not cause a false echo on next ranging [8].

Pi camera is 5 MPixel and connects to Raspberry Pi via a ribbon connector to the CSI connector. This combination of Pi camera and raspberry Pi is overwhelmingly fast, having the capacity to send 1080 p images at 30 frames per second or at higher frame rates for lower resolution images. The video and still image quality for Pi cam is better than USB cams of same price. It is void of IR filter and hence can see near-IR wavelengths (700-1000 nm), with a tradeoff of poor color rendition. This 5 MPixel camera has fixed-focus and supports VGA 90, 720p60 and 1080p30 video modes and still captures as well. It can be through MMAL and V4L APIs and also there are third party libraries like Pi Camera Python that have been built for it [9].

H-Bridges are typically used in controlling motor speed and direction, but can also be used for other projects. An H-Bridge is a circuit that can drive a current in either polarity and can be controlled using Pulse Width Modulation (PWM). The H-Bridge used is driven by

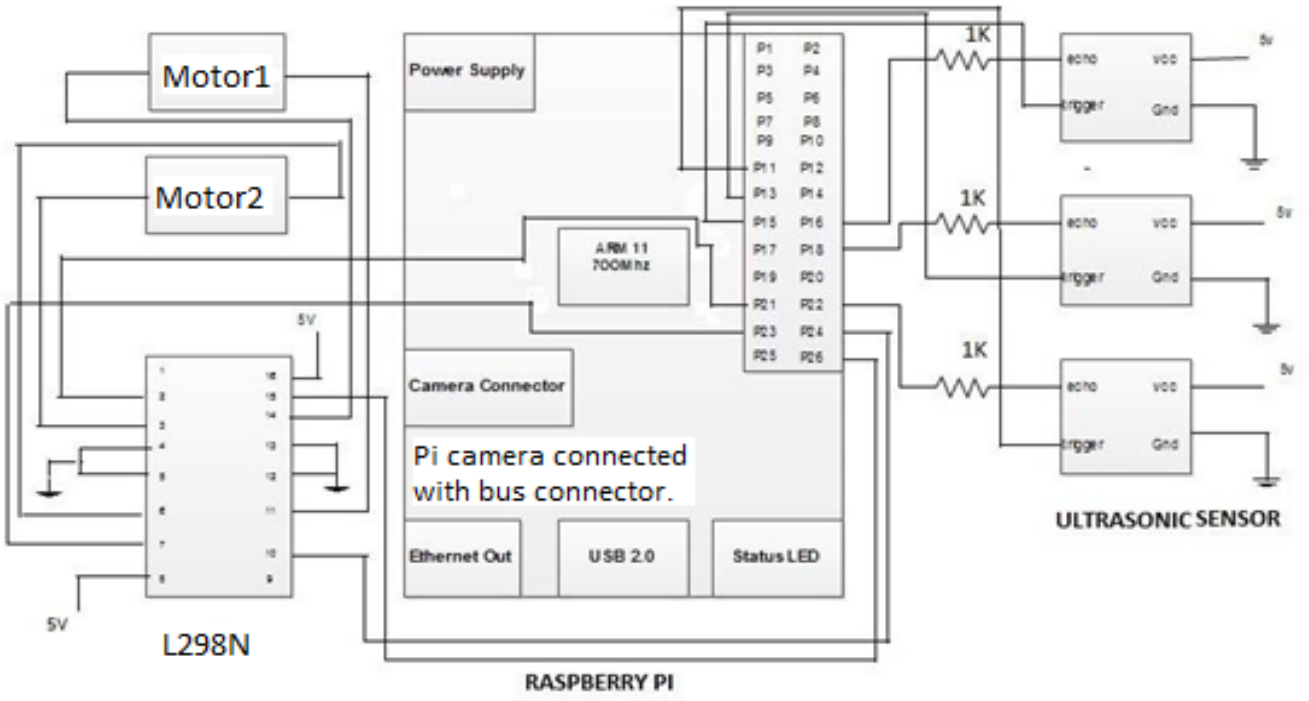

Figure 2: Hardware architecture of the project.

L298N chip. Its logical voltage requirement is 5 V and the drive voltage ranges from 5 V - 35 V, though it is usually marked as 12 V. It has various pins for control signal input from the processor and output signal to the motors [10].

These components are interfaced together for getting to the desired result. The drive voltage provided via Dual H-Bridge to the motors is derived from an external battery attached to the circuitry.

Figure 2 represents the hardware setup of the aforementioned application.

Software Architecture: The Project has been implemented on the very conventional software model of "sense-plan-act". To sense the environment, we use sensors and in the project we are using camera and ultrasonic sensor as environment understanding sensors. So the model has a multi-sensor sensing module. These sensor inputs are fused for proper perception of the environment around.

Once the sensors have sensed the surroundings and the fusion of their data in perceivable form is received we now need to plan our course of action according to the data. In planning we perform two tasks;

Mapping: We create a logical equivalent of the surrounding environment from the sensed data.

Setting Target (Mission): Here we according to the logical map decide our target position.

Once done with planning we need to act so as to achieve the target. In this scenario our target is to reach the decided location. So after planning to act to reach the target we have to perform motor control. In motor control we need to control speed, direction and turning phenomenon. Once we reach our target we start from sensing again and continue the process in a recursive fashion.

This schema of the software architecture is represented in Figure 3.

Algorithm Overview: Since we are using "sense-plan-act" module for the implementation of the project, so its various steps in correspondence are:

Sensing Environment (Sense) - In this part we simply use the camera and ultrasonic sensors to collect the data.

Sensor Data Fusion (Sense) - In this part we combine the sensor inputs, so the required action can be taken according to the priority.

Trajectory Generation (plan) - This is an input based module of the working model. It detects the objects in the motion path and accordingly forms the motion trajectory.

Motion Planning (plan) - Once we have the environment map, we plan the motion of the vehicle in this segment of mapping.

Motor Control (act) - In this part we program the motors to take the required action, be it stopping, turning or accelerating on the basis of the planned trajectory.

Per-segment Planning (plan) - We use this method to improve the efficiency of the algorithm. Figure 4 explains the advantage of using per-segment planning.

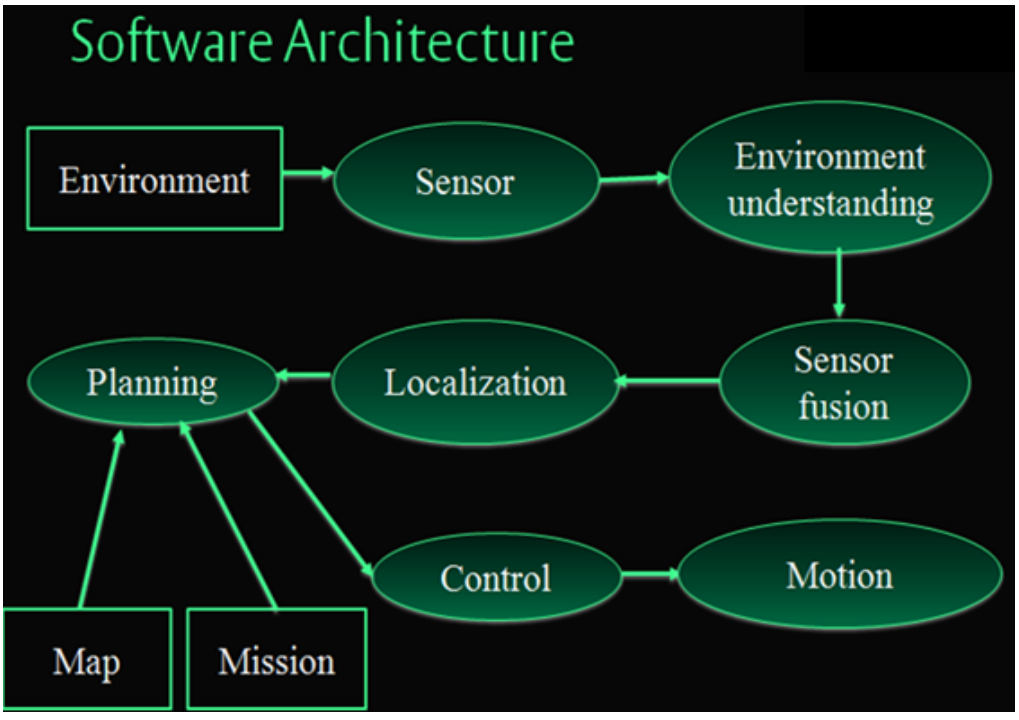

Figure 3: Software architecture.

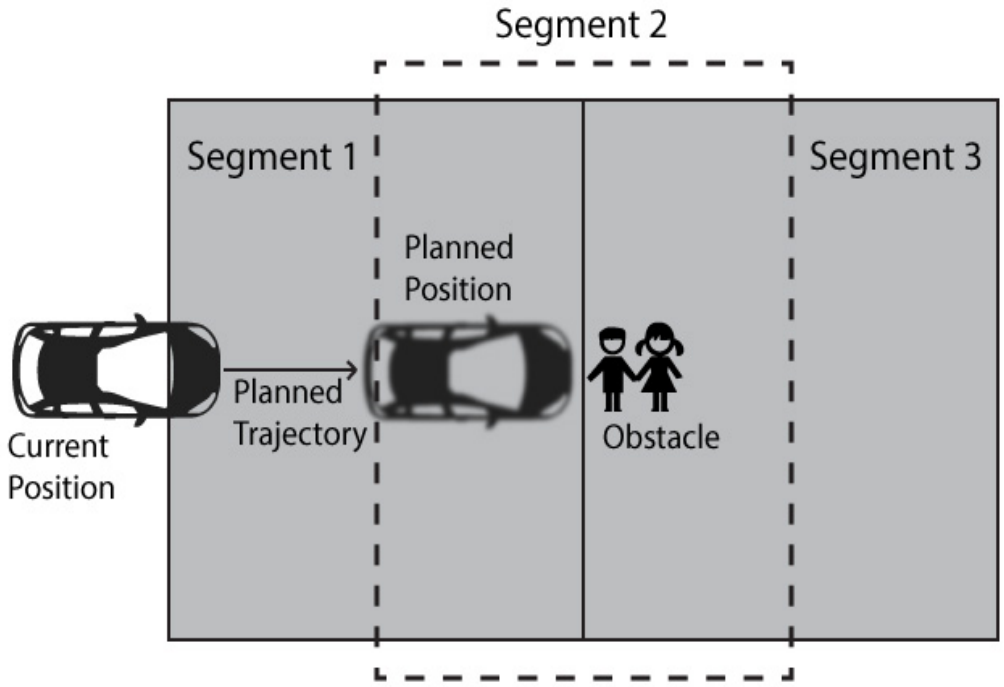

Figure 4: Per-segment planning.

In Figure 4 we demonstrate the overlapping segment breakup. Moving by planned trajectory in segment 1, with a segment 1 only vision would result in the vehicle coming too close to the obstacle. Hence segments are overlapped and the trajectory is re-planned at the entry of segment 2.

Evaluation

Microcontrollers have been the most dominant element in the field of augmentation of driving vehicles. With a typical ford vehicle having 25-35 electronic control units (ECUs), whilst a luxury car like BMW 7 series has 60-65 ECUs [11]. The type of microcontroller depends on the application, be it vehicle control, power train or driver information. Some examples of microcontrollers used in automobiles are Tri-core microcontrollers (assembled in over 50 automotive brands), AVR microcontrollers (with inbuilt ADC), PIC microcontrollers and Renesas microcontrollers (Latest in the automotive microcontroller family) [11]. Advanced driver assistance systems (ADAS) today employ a microcontroller with each sensor and process the data acquired from the sensor, this reduces the data to be sent to the central processor and also the computational load of the central processor. This schema of augmentation of driving vehicles has been widely used until now. The various problems with this approach are, local processing of the data introduces delay before sending it to the central processor, problems in comparing data from multiple sensors to distinguish an aspect etc. Today engineers feel that to make these judgements the fusion of sensor data is best handled by a central controller, which also processes the raw data from the sensors simultaneously [12].

In 2017 the introduction of DRS360 autonomous driving platform by Mentor is an approach towards level five automation using centralized fusion of raw data. In contrast to the previous distributed computing based approach, the centralized raw data based approach uses the entirety of all the acquired data and hence is capable of generating a more optimal view of the vehicle environment. Using raw and unfiltered data is the ideal method to obtain a complete, spatially and temporally synced view of the vehicular environment, and achieving this target was made possible by bringing forth together the FPGA and CPU technologies on the same platform. Nearly any camera, LIDAR, radar or other sensor types can be adapted to the platform, which is open and transparent. Mentor's sensor fusion and object recognition algorithms are advanced, but companies can apply their own algorithms and even chose their own chips and SoCs for the processing [12].

Now in a more general perspective let's compare microcontrollers and SoCs. Interfacing sensors and cameras and programming them with SoCs is a much simpler task when compared to microcontrollers and general purpose computers. Microcontrollers have their set of limitations in processing capacity and programming feasibility. While general purpose computers do not provide customizable plug and play interfaces for hardware peripherals like sensors and cameras. Hence SoCs provide an optimum mix of microcontrollers and general purpose computers. In Table 1, we have tried to give a basic comparison of microcontrollers and SoCs based on a specimen for each category.

Though performance and interactive operating system provided by single board computers is better than SoCs, but they don't provide any plug and play interfaces and also they are comparatively very costly. Nor is there any availability for customizing pins for interfacing peripherals like sensors, motors etc. Hence we do not use single board computers for function specific automation systems. The feasibility and low cost with a tradeoff of performance by SoCs is a very optimal solution for the function specific automation systems. Figure 5 gives a statistical analysis of instructions per unit time for various raspberry Pi boards. The instruction rate of all raspberry Pi boards is compatible for ITS, with later versions being more efficient.

Automation systems are based upon sensing modules for sensing environment using sensors, computer vision provided by cameras and processing platforms, central processing space that retrieves data from sensors and processes it for ambient response. The processed data performs response via actuators like motors, servos etc. An automation system has its specific functional domain hence it needs a moderate processing space at feasible price with facilities for customizing pins to interface sensors and actuators. This setup is best provided by SoCs in comparison to microcontrollers and single board computers.

Table 1: Comparison between SOC and microcontroller.

Chip	Microcontroller (Arduino)	SoC (Raspberry Pi)
Cost Per Unit	$10	$50
Performance	**16 MHz**	**1.2 GHz**
Flexibility	High	High
Sensor Interfacing	Feasible	Feasible
LAN Connectivity	Not available	Built-in Port
Operating System	**Not available**	**Linux based and Windows 10 IOT Core**

To interface an ultrasonic sensor with our computers and laptops is a very complicated task to accomplish. While interfacing ultrasonic sensors with Arduino is feasible but not simple enough. Microcontrollers do not have an operating system, so to get our sensor integrated and running with Arduino we need to connect it to another computer via a serial connection and download an Arduino IDE in it. Then either upload a predefined sketch in IDE for ultrasonic sensor or write a new code for ultrasonic sensor to Arduino. So Arduino is not an interactive enough option for programming and interfacing sensors. In contrast SoCs have feasible plug and play pins like microcontrollers and an interactive Linux operating system to work upon. Raspbian operating system contains python setup, hence providing an easy option for programming the ultrasonic sensor. With built-in libraries for Raspberry Pi in python it is easy to configure GPIO pins for sensor interfacing. Also with the vast functionalities provided by python libraries it makes system designing easy for the programmer.

The data input from the ultrasonic sensor helps us determine the course of action that we should take to avoid possible collision in case of presence of an obstacle. So we import the data from sensors to a separate class to manipulate the action performed by the motors. Unlike microcontrollers we need not create a separate sketch and then upload or write it back, rather we need to create a separate program and simply import the data retrieved by sensors

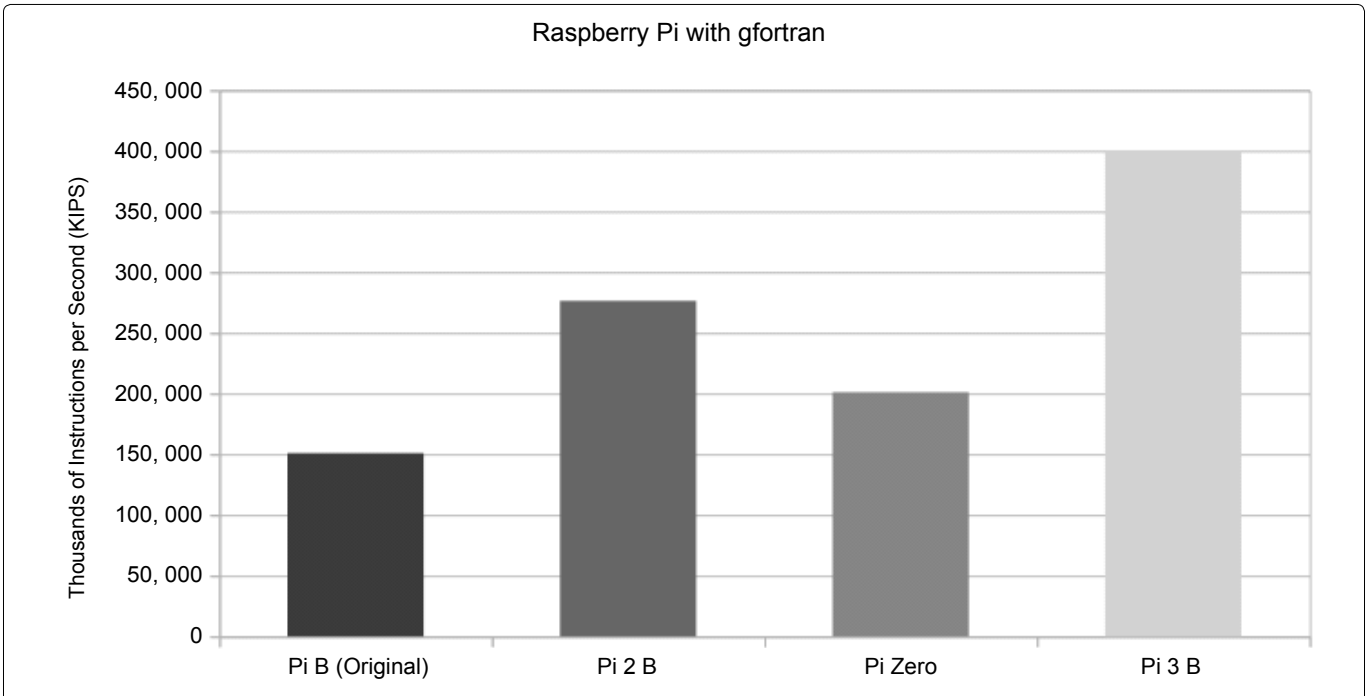

Figure 5: Whetstone double precision benchmark [13].

into this program. Then the output signal is carried on to motors via Dual H-Bridge. We can manipulate the working of H-Bridge with the help of pulse width modulation (PWM). In the defined scenario the sensor data needs to manipulate the driving speed and direction. Driving speed is based on motor speed, and driving direction is based on relative motor speed. We can manipulate motor speed by varying the duty cycle of each pulse width modulation. Higher duty cycles correspond to higher speed. The relative motor speed can be manipulated by setting up different duty cycles for different motors, using the specific GPIO pins for each motor. Though the manipulation via PWM can also be provided by microcontrollers like Arduino but every-time we create a new manipulation we have to create a new sketch and write it back on the microcontroller. However, for single board computers we are not provided with customizable pins which can be manipulated using PWM and hence we can't perform the required action.

Now for computer vision we use cameras and software platforms. In our project we have used Pi camera and openCV software. openCV (Open computer vision) software is designed for aiding computer vision. Pi camera was also developed to aid Raspberry Pi based computer vision. SoCs like Raspberry Pi made it possible to provide computer vision in low priced robots and automation devices, which otherwise was restricted to very expensive automation devices alone. Now for computer vision our camera should be able to capture all the activity on the road, at the junction. Minimum vehicle length is around 2 meters. So the effective length for the vehicle to be captured by camera is 4 meters. As the camera can capture either the head of the car or the tail of the car, thus providing us an effective length of 4 meters instead of 2 meters. Figure 6 explains the idea of effective length.

Now maximum speed that a vehicle can attain at a junction is around 120 KMPH, which is around 33 MPS.

Mathematically analyzing this situation, we have;

Vehicle speed = 33 m/s

Time taken to travel 1 meter = 1/33 s

Hence, Time taken to cover 4 meters = 4/33 s

So the camera needs to capture the image in less than every 4/33 of a second.

Thus, required camera framerate > 33/4 fps = 8.25 fps.

But this is not the absolute framerate of the camera, it is the framerate that the camera has to maintain in simultaneity with image processing algorithms for edge detection, filtering etc. Also to be on the safer side we would prefer an available framerate of 10 fps. The usual framerate of a USB camera is around 15 to 20 fps without any processing going on at the backend of it, and that too for a resolution less than 1080. This framerate goes plummeting down to mere 5-7 fps after applying processing algorithms and that too at lower resolutions. Now USB cameras with higher framerates come at very expensive prices. On the contrary Pi camera has an absolute framerate of 30 fps for 1080-pixel resolution and a higher framerate for lower resolutions. Applying processing algorithms takes down the framerate of the camera to around 15 fps at minimum. Also the price of Pi camera board is considerably less. Now let us analyze the three available prospects of microcontroller, SoC and SBC for a practical nodal situation.

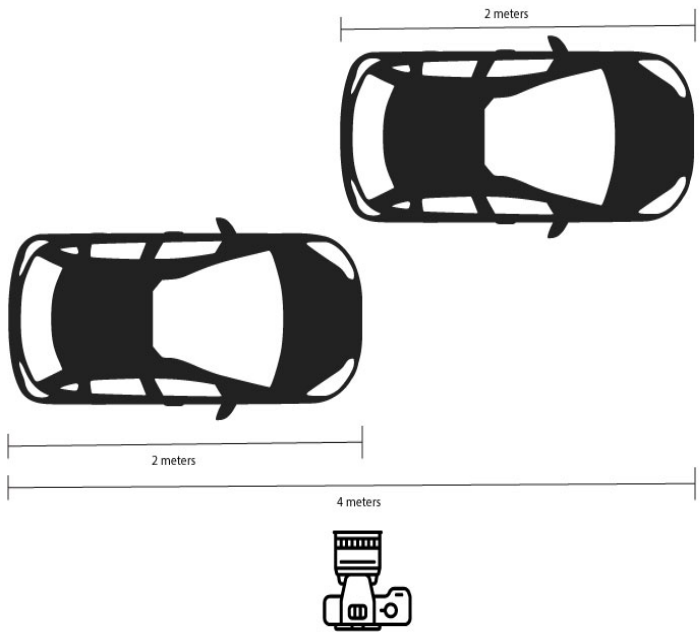

Figure 6: Vehicle effective length.

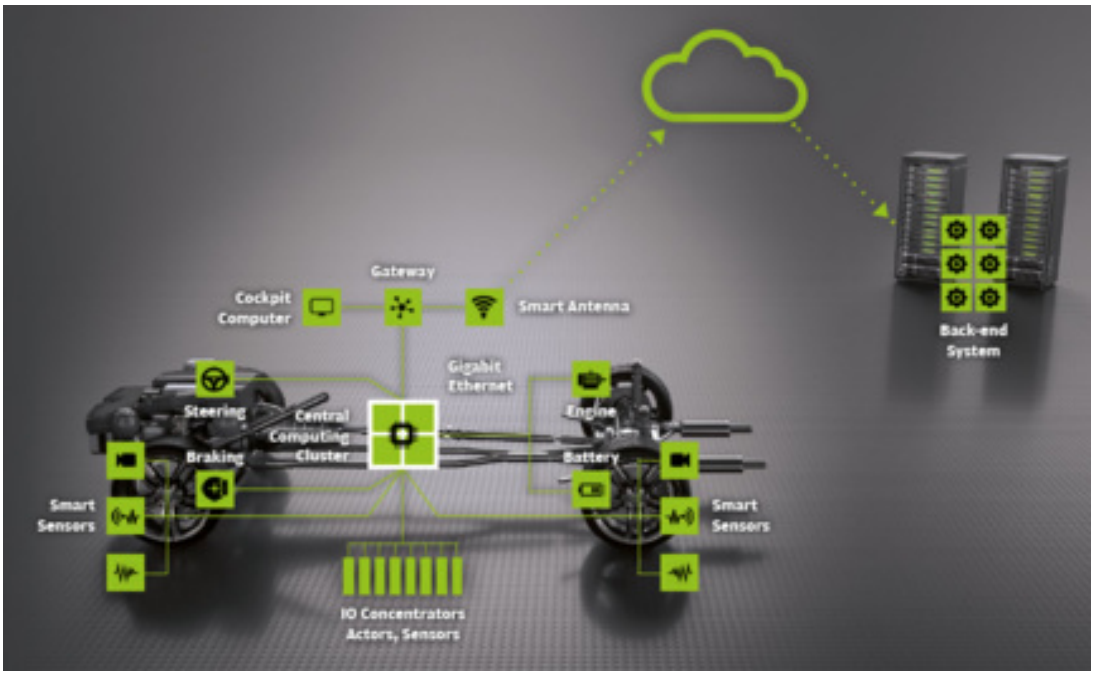

Figure 7: Future outlook of AV system [15].

Space Constraint: At a traffic junction the available space for implementing such devices is not so ambient for SBCs, whilst microcontroller and SoC setup can fit in.

Cost Constraint: Cost of a simple and portable SBC is at minimum around Rs 21000 ($330). Other than that the cost of high-framerate USB camera is also around Rs 6400 ($100). Summing up the main amount without other accessories like sensors PCBs etc. will be around $450. This amount is very high compared to the implementation price of similar functionalities using microcontrollers and SoCs.

Feasibility Constraint: We need a setup providing feasibility for hardware connectivity and computer vision. Computer Vision setup needs device compatibility with camera and sufficient processing power for image processing. Microcontrollers though provide easy hardware access to sensors and have good plug and play interfaces, but they lack compatibility for cameras and the required processing power hence making them not so suitable for the situation. Whilst SBC provides very massive processing power and ready camera compatibility, it lacks the easy access towards hardware and does not provide the luxury of plug and play interfaces. SoCs on the other hand provide plug and play interfaces for easy hardware access and are also very good for computer vision with camera compatibility and enough processing power. So SoCs are an efficient and optimal choice for this case.

Now taking a look at more general advantages that the SoCs can provide in our case. Better performance due to its capability of exploiting hardware parallelism, provides us faster interfaces for retrieving and processing sensor data. Also the non-recurring expenditures (NRE) of an SoC are much lower, which save us a lot of maintenance cost. Parallel operations with lower clock rates and ability to sleep when inactive reduces the power consumption [14]. These advantages have led to the use of SoCs in many automation and smart-tech projects to envisage a better and safer future.

Future Scope: The field of SoCs has yet a vast expanse to cover. It has just started to impact the world of technology. Smartphones, research purpose boards, hand held devices and IoT architectures are a cogent proof of their technological excellence.

SoCs can go multi-core to make their processing performance even more efficient. We can let SoCs go for hyper-threading so as to make parallel processing look even more convenient on the boards. This will facilitate the centralized fused sensor based approach for level 5 automation of vehicular system.

Moreover, the processing setup of the SoCs can be made smart. This would enable auto turnoffs for power consumption efficiency and smart switching of processor to enable temperature management and much more [4].

We need to develop customized SoCs such that drive domain is separated from the interface domain. Then we need to connect the SoC to the Back-end system at the interface domain using a smart antenna. The connection to the Back-end system provides the vehicle with information like road condition, free parking spaces etc. Whilst the segregation of drive domain and interface domain is required to secure the vehicular safety even in case of intrusion or overburdening data input from the smart antenna [15].

Figure 7 gives an overview of such a customized SoC based AV system.

Another important aspect to work on is the potential for mixed signals. With upcoming semiconductor structures we might be working on different signal levels like 1.8 V for 28 nm process technology. We should be able to handle this system of mixed signals. We can work on generation of charge pumps for SoCs to handle such issues. This will make the AV system free to choose any sensor as per the requirement without worrying for signal specifications [4].

Hence SoCs with their current vast scope and application also have a lively and exponentially expanding scope for future improvements. With companies like Renesas, NXP and Nvidia building SoCs like R Car H3, Bluebox and Drive PX respectively and having joint efforts with automotive companies like, Tesla, BMW, Volvo etc. In future the SoCs will play a pivotal role in the field of vehicular automation and will give researchers an immense scope to work on.

Conclusion

SoCs are like the much needed anecdote that provided the intermediatory solution to the struggle of choosing between microcontrollers and computers. SoCs provide us with the required processing speeds, operating system access, easy programmability like computers and reduced cost, size and easy hardware interaction like microcontrollers. In the field of ITS and automation we always needed something which settled between computers and microcontrollers. SoCs seem to be tailor made for those projects and applications.

Acknowledgement

The author appreciates support from the Department of Information Technology, National Institute of Technology Srinagar. The author also extends his gratitude towards Muhammad Talha Bilal for his assistance in improving the graphic content of figures.

References

1. Jonathan Diclemente, Serban Mogos, Ruby Wang (2014) Autonomous car final report. Carnegie Melon University.
2. Abhishek Ghosh (2012) System on a chip (SOC): A brief details.
3. Global Industry Analysts, Inc. (2017) “System-in-Package (SiP) technology”. Global Industry Analysts.
4. Heng Sin Wei, Adrian Kong Yeng Hong, Chris Liu Chaofeng, Goh Chee Peng, Jason Koh Sheng Fa (2012) “System on chip (for mobile devices)”.
5. Nishant Mittal (2016) Design advantages of programmable, efficient SoC architectures.
6. (2017) Raspberry pi technology, working and its applications.
7. (2017) Raspberry Pi 2: Which OS is best? | element14 | Raspberry Pi 2.
8. Vivek Kartha (2015) “Interfacing HCSR04 ultrasonic sensor with raspberry Pi”.
9. https://elinux.org/Rpi_Camera_Module
10. https://www.bananarobotics.com/shop/How-to-use-the-L298N-Dual-H-Bridge-Motor-Driver
11. Tarun Agarwal “Different type of microcontrollers are used in automobile applications”.
12. Lee Teschler (2017) “Centralized processing for autonomous vehicles”.
13. Tim Holyoake (2016) “Benchmarking the raspberry pi 3”.
14. Paul Beckett (2015) “Some FPGA SoC applications and trends in advanced system on chip”. RMIT University, Melbourne, Australia.
15. Rudolf Grave (2017) “The vehicle architecture of automated driving level 2/3”.