AUTOMOBILE DASHBOARD IMPLEMENTATION USING CAN BUS & FREE RTOS

Imranullah Khan. Md
PG Student,Department of ECE
Jerusalem college of Engineering,
Anna University
Chennai,India
ik4081@gmail.com

Tamil Selvi
Professor,Department of ECE
Jerusalem college of Engineering,
Anna University
Chennai,India
tamilselvi@jerusalemengg.ac.in

Sheeja V Francis
Professor& HOD,Department of ECE
Jerusalem college of Engineering,
Anna University
Chennai,India
sheejavfrancis@jerusalemengg.ac.in

***Abstract*—** An electrical device used in vehicles Right in front of the driver sits the dashboard, a control panel that houses all the necessary instruments and buttons for controlling the vehicle. These days, it's common for cars to have many digital dashboards, all of which work together to make driving easier and safer.In the end, this project aims to create a dashboard interface for an ECU that can monitor a variety of parameters, including fuel level, seatbelt alert warning, safety system with airbag management, and more. The project will be carried out using a 32-bit controller that is based on the ARM7 architecture.The NXP LPC2148 is a 32-bit RISC microcontroller with Thumb extensions based on the ARM7TDMI-S architecture, built by Philips's offshoot firm. The device has a lot of features, including 32 KB of RAM, 512 KB of on-chip flash memory, a vectored interrupt controller, two 10-bit ADCs with fourteen channels, a USB 2.0 full-speed device controller, two UARTs, two serial interfaces (I2C and SPI), and a lot more.

Keywords— Low Power control, Controller Area Network, Serial Peripheral Interface.

I. INTRODUCTION

One of the many benefits of automotive electronic modules is their low cost, simplicity of deployment, and integration into a peer-to-peer network that provides improved transfer rates of 1Mbits, robust error checking, and so on. The building blocks of a peer-to-peer network are interconnected machines. These nodes can monitor different parameters and notify the CCU when something changes. It is mostly used in vehicles that operate in hazardous situations because to its reliability.

The first node, an LPC2148 microcontroller based on ARM7, is where we are attaching sensors. Vehicles are fundamental to human existence on a daily basis.Traditional electrical systems improve ride quality, handling, and fuel efficiency, but they also increase the likelihood of accidents. In order to address issues like reliability, complexity of body wires, and space limitations, protocols are essential for car networking [1].

Many different kinds of networks and protocols are presently in use by different automakers.One of several bus protocols, CAN was developed in the 1980s by German inventor Robert Bosch for use in serial communication between network nodes.The automotive industry embraced it for use in automobiles after its consistent execution transformed it into the globally recognized standard known as the ISO 11898 model [2].

Roughly one hundred Electronic Control Units (ECUs) regulate the electrical systems of modern cars, improving the driver's convenience and security while driving. The majority of the vehicle's systems are

managed by electronic control units (ECUs), such as the anti-lock braking system, airbag deployment, and safety-critical engine management. The reliability of an ECU's communication is crucial to the driver's safety while driving.

The most common method of internal vehicle communication is the Control Area Network (CAN). Its well-known benefits—including its ability to self-diagnose and rectify problems, its simplicity of wiring, and its high tolerance to electrical interference—make CAN bus an excellent match for the automotive sector.

RELEATED WORKS

Almost all fatalities each year are the result of automobile accidents. A road accident claims the life of an Indian citizen every four minutes.Quick expansion is being seen in the automotive sector. Carhartworking technology places a premium on safe vehicle operation.Apt for use with embedded microprocessors or microcontrollers in real-time applications. There are usually both hard and soft real-time requirements for applications like this. We argue that a soft real-time need exists when a deadline is given, yet the system can still operate properly if there is a delay. To illustrate the point, a system may seem unresponsive without really being rendered worthless if it is too slow to respond to keystrokes.

We argue that a need is hard real-time when there is a deadline that the system cannot afford to miss. For example, if a driver's airbag responds too slowly to information from collision sensors, it might do more harm than benefit.Free RTOS is a real-time kernel (or real-time scheduler) that may be used to build embedded programs that meet their rigorous real-time requirements. It makes it possible to organize programs as a collection of independent execution threads.

A single-core CPU can only handle the processing of a single thread at a time.The application's designer gives each thread a priority so the kernel knows which one to execute. At its most fundamental level, the designer of an application may prioritize threads that execute hard real-time demands above threads that execute soft real-time requirements.

In theory, this would ensure that hard real-time threads would execute before soft real-time threads, but in practice, prioritizing decisions aren't binary.The first publication's writers accomplished remarkable performance and CANbus integration with the help of the ARMController, which is the principal controller unit of a vehicle. Control networks may enhance their mutual performance via the use of may, which allows for high-speed communication and data sharing among station nodes.

II. TECHNICAL APPROACH

Figure 1 shows how the CAN protocol is used in car electronics. The CAN modules used in automotive control systems are used in this project.For data transmission, there are two microcontrollers on each side, and for encoder and decoder, there are two CAN modules used as CAN Trans receivers.

The microcontroller is connected to the sensors and power supply unit on the encoder side, and the CAN transceiver is attached to it. A CAN transceiver and an integrated LCD display microcontroller comprise the decoder side setup.

Use of real-time operating systems is recommended for applications involving microcontrollers or small microprocessors embedded deep inside a system. There are usually both hard and soft real-time requirements for

applications like this. The system would continue to operate correctly even if the deadline was missed, as indicated in soft real-time requirements.

To illustrate the point, a system may seem unresponsive without really being rendered worthless if it is too slow to respond to keystrokes.We argue that a need is hard real-time when there is a deadline that the system cannot afford to miss. For example, if a driver's airbag responds too slowly to information from collision sensors, it might do more harm than benefit.

In an embedded system, the two most important parts are the memory and the CPU or microcontroller. Some of them have a serial port or may connect to a network. Usually, they can't read, write, or access files saved on drives.

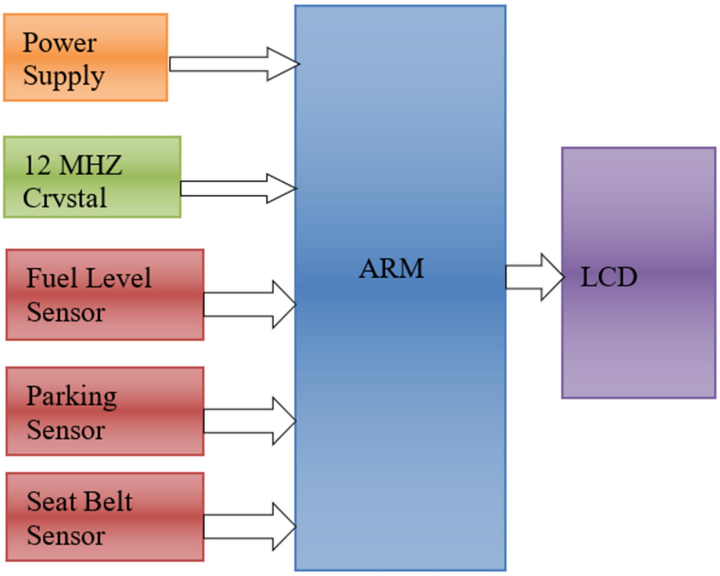


Fig. 1 .Electronic Control Unit Node 1

One of the many benefits of automotive electronic modules is their low cost, simplicity of deployment, and integration into a peer-to-peer network that provides improved transfer rates of 1Mbits, robust error checking, and so on. The building blocks of a peer-to-peer network are interconnected machines. These nodes can monitor different parameters and notify the CCU when something changes. It is mostly used in vehicles that operate in hazardous situations because to its reliability. The first node, an LPC2148 microcontroller based on ARM7, is where we are attaching sensors.

The MCP 2551 is well-known as a rapid CAN transceiver, capable of operating at rates as high as 1 MB/s. A CAN device mostly links to the CAN controller and the physical bus. The may protocol controller makes use of differential signals, which the CAN module may send and receive in a way that is compliant with the ISO-11898 standard.Display and monitor for safety belts:

Reminder to fasten seatbelt The presence of an outside object or the motion of an inside machine part may trigger a limit switch. In addition to controlling and interlocking machinery, a limit switch also acts as a safety feature.

Simple Limit Switch

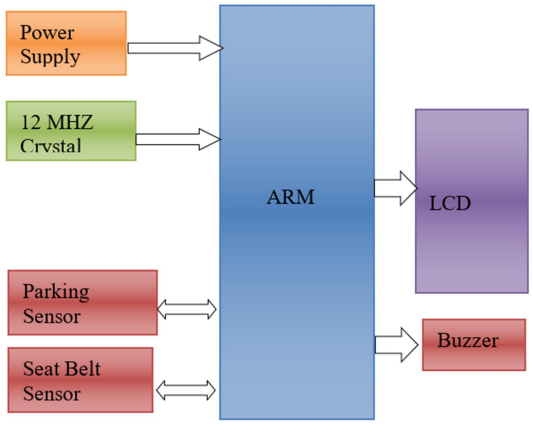


Fig. *2.Electronic Control Unit Node 2*

Conventionally, a device that converts the digital signals generated by a CAN controller into signals suitable for bus communication (differential output) is required at every node. Similarly, it acts as a buffer between the CAN MCP 2551 controller and any external sources that may generate high-voltage spikes on the CAN bus. In this setup, the MCP 2551 serves as both a transmitter and a receiver. Two states are shown on the transmitter CAN bus: recessive and dominant. Each condition is determined by the current value of the voltage difference between the CAN High and CAN Low lines. When it's below 1.2 V, the state is dominant; when it's beyond, the condition is recessive. In the Open Systems Interconnection (OSI) model of a communication system, the MCP2551 operates at the physical layer, which is layer 1.Dashboard sensors often make use of liquid crystal displays, or LCDs, which combine the best features of liquids and crystals. Molecularly organized into a crystal-like structure, they exhibit almost liquid-like mobility within a narrow temperature range.

'Smart LCD' displays are increasingly being used by microcontroller devices to illustrate data visually. What follows is an explanation of how to link an LPC2148 microcontroller to a Hitachi LCD screen. Affordable, user-friendly, and capable of producing a readout with its 8 × 80 pixels, LCD displays built on Hitachi's LCD HD44780 module are a great choice. Hitachi LCD screens have not just the conventional ASCII set of characters but also Greek, mathematical, and Japanese symbols.

A +5V supply and eleven I/O lines are required for an 8-bit data bus display. Just the supply lines and seven more lines are needed for a 4-bit data bus. If the LCD screen is off, the data lines are tri-state, meaning they have a high impedance (as if they were disconnected).Additionally, the microcontroller needs three "control" lines to power the LCD.

In order to help drivers avoid collisions, parking sensors are installed in vehicles. A controller and ultrasonic sensors that release sound waves constitute the bulk of these systems.

while traveling, collecting data on the time it takes for signals to reflect off of various surfaces.

Fig. 1 .Electronic Control Unit Node 1

In these systems, the CAN transceiver 2551 plays a crucial role in both the encoding and decoding of sensor inputs and outputs to and from the microcontroller IC, as shown on the LCD screen.The fuel gauges in your car may tell you how much gas is in the tank by monitoring the voltage across a sensing system's variable resistor.

Fig. 3 .Electronic Control Unit N

III. EXPERIMENTAL EVALUATION

It displays the system's workflow,First, the ARM7 microprocessor and CAN transceiver are initialized. Then, the sensor is started and its data is scanned. If the data is Analog, it is changed and sent to the microcontroller. If it is Digital, it is retrieved by the CAN module.

IV. REAL TIME INTEGRATION

Finally, the model we developed is integrated into the program for playing music. It monitors the environment in real time, increases or decreases the volume as needed, and keeps the user informed at all times.Because it serves as the job's stack and contains task information in the task control block (TCB), random access memory (RAM) is crucial for every work. When you create a new task using xTaskCreate(), it will trigger the required.

Allocating heap memory is done automatically by FreeRTOS. If you use xTaskCreateStatic() to create the task, you may statically allocate RAM at build time. However, this technique brings two more parameters to the function as the application writer is required to give the RAM. When a task is created, it is first set to the Ready state. On the other hand, pvTaskCode will immediately go into the Running state if they exist. Since tasks in C are essentially simply functions that never exit, an infinite loop is a typical approach to design them. All the pvTaskCode parameter really is is a pointer to the function (basically simply the name of the function) that runs the task. User ID for Computer A job having a name that describes it. Sending this to

xTaskGetHandle() will obtain a handle to a task; however, it is mostly used for debugging. One application-specific parameter, configMAX_TASK_NAME_LEN, specifies in characters the maximum allowable name length (including the NULL terminator).

Beyond this length, whatever string you provide will be Because it serves as the job's stack and contains task information in the task control block (TCB), random access memory (RAM) is crucial for every work. The required RAM is allocated from the FreeRTOS heap during the xTaskCreate() job creation process. Developers may statically allocate RAM at build time when jobs are created using xTaskCreateStatic(), albeit it adds two more parameters to the function.

As soon as they are created, tasks are initially set to the Ready state. Nevertheless, they will immediately go into Running mode if no higher-priority activities can perform. Before or after launching the scheduler, you may choose to create tasks.Invokes vTaskDelay() to place the caller task in the Blocked state for a set number of tick interruptions. When a delay time of zero ticks is given, the calling task will not enter the Blocked state. However, it will surrender to any Ready state tasks that share its priority. Simply using vTaskDelay(0) will invoke taskYIELD().

Important Differences Between vTaskDelay() and vTaskDelayUntil(), the calling task enters the Blocked state and remains in that condition for the provided number of ticks starting from the time vTaskDelay() was called. The invocation of vTaskDelay() determines the relative timing of the task's departure from the Blocked state. The currently running task will enter the Blocked state and remain there until you specify a certain amount of time using vTaskDelayUntil(). The job that caused the delay will exit the Blocked state at the exact instant that is supplied, regardless of when vTaskDelayUntil() was invoked.Releases an instance of a task that was first created using the xTaskCreate() or xTaskCreateStatic() functions.

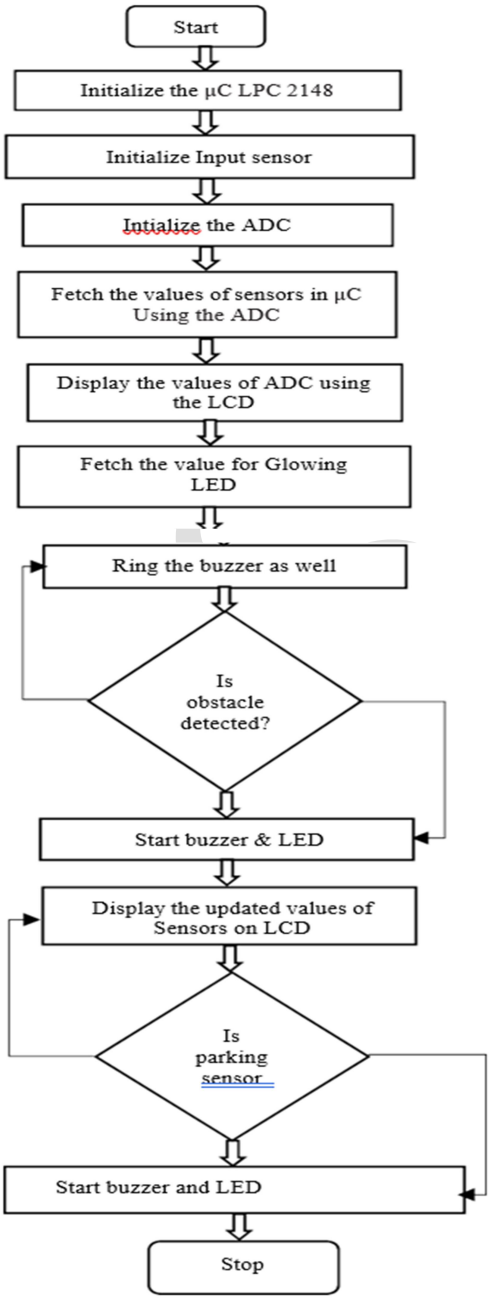
Any tasks that have been removed will no longer have the ability to enter the Running state. Once you delete a task, you will no longer be able to access it by its handle.After a task is withdrawn from execution, it is the responsibility of the presently running task to free up the memory that held the stack and data structures (task control block) of the deleted process.

Accordingly, programs that use the vTaskDelete() API function must further guarantee that the idle task does not go unprocessed. While the job is in the Running stage, it needs time to idle.Removing a task just releases the memory that it has been allocated by the kernel. It is the responsibility of the application, not the kernel, to free up any memory or other resources assigned to a job when the application deletes it.A new software timer is created and returned by this method.

Each software timer requires a little amount of RAM to keep track of the timer's current state. When you use xTimerCreate() to create a software timer, FreeRTOS will automatically allocate this RAM from the heap. An additional input is required for the program developer to statically allocate a software timer at compilation time using the xTimerCreateStatic() method, however this method is still preferable.

When many timers use the same callback function, you can tell which one has expired by looking at the timer identifier in the callback code. The timer identifier also allows for the storage of values between calls to the callback function of the timer.Typically, the scheduler is not started until the main() function or one it calls is already executing. The scheduler will never do anything other than execute tasks and interrupts once it is started.The scheduler enters the Running state when the highest-priority task that was produced during the Initialization stage is executed.Automatic generation of the Idle task occurs upon scheduler startup. Assuming

there is enough heap RAM in FreeRTOS to build the Idle task, vTaskStartScheduler() should return. On an ARM7 or ARM9 microcontroller, the CPU has to be in Supervisor mode to use vTaskStartScheduler().



V. CONCLUSION

Free RTOS is an affordable, lightweight, and powerful operating system option. Using microcontroller, the controller establishes connections to several kinds of sensors. So, we achieved our goals and the "CAN protocol In future can be used and its used in Automation Electronics" was a success. To further improve the system's

ability to detect and avert accidents, it may be enhanced by adding additional sensors such as vibration, infrared, and proximity sensors.

REFERENCES

[1] Jufang Hu , Chunru Xiong, "Study on the Embedded CAN Bus Con- troll System in the Vehicle", International Conference on Com- puter Science and Electronics Engineering, IEEE, pp. 440-442, 2012.

[2] Saif Al-Sultan, Ali H. Al-Bayatti, and Hussein Zedan, "Context Aware Driver Behaviour Detection System in Intelligent Transportation Systems" IEEE transactions on vehicular technology, vol. 62, no. 9, pp.4264-4275, November 2013.

[3] Wang dong, Cheng quan cheng, Li Kai, Fang Bao-hua, "The automat- ic control system of anti-drunk-driving" in IEEE 978-1-4577-0321- 8/11, pp. 523-526, 2011.

[4] Ji Hyun Yang, Zhi-Hong Mao, Louis Tijerina, Tom Pilutti, Joseph F. Coughlin, and Eric Feron, "Detection of Driver Fatigue Caused by Sleep Deprivation", IEEE Transactions On Systems, Man, And Cy-bernetics—Part A: Systems And Humans, vol. 39, no. 4, pp. 694-703, July 2009.

[5] B.Gmbh, "CAN specification" vol 1 Version 2.0, 1991

[6] Pazul, "Controller Area Network (CAN) Basics" Microchip technology Inc., AN713, May 1999.

[7] Ashwini S. Shinde, Prof. vidhyadhar and B. Dharmadhikari,"Controller Area Network for Vehicle automation", International Journal of Emerging Technology and Advanced Engineering, Vol. 2, Issue 2,pp.12-17, 2012.

[8] P. R. Burje, K. J. Karande, and A. B. Jagadale, "Embedded On- Board Diagnostics system using CAN protocol", pp. 734–737,2014.

[9] R. Ranjan and K. S. Chari, "Design of Controller Area Network Based Automated Safety System for Vehicle", pp. 7, 2017

[10] Maria Pinto, et. al., "Influence of front light configuration on the visual conspicuity of motorcycles" Elsevier-International Journal of Accident Analysis and Prevention, vol. 62, pp. 230-237,2013.

[11] Dr. K. Shashidhar, B Benhur, Khaja Moinuddin, Rizwan Ahmad, Design And Implementation Of Iot Based Smart Health Care Monitoring System, International Journal of Multidisciplinary Engineering in Current Research - IJMEC Volume 8, Issue 1, January-2023, http://ijmec.com/, ISSN: 2456-4265.

[12] Mrs. Sharea takreem, syed junaid, Syed hasan noori, syed junaid**,** IOT based speed control of the AC motor control and security using OTP, International Journal of Multidisciplinary Engineering in Current Research - IJMEC Volume 8, Issue 1, January-2023, http://ijmec.com/, ISSN: 2456-4265.

[13] http://www.keil.com/dd/docs/datashts/philips/lpc2119_2129.pdf (Last accessed 16-Oct-2020)

[14] http://ww1.microchip.com/downloads/en/DeviceDoc/21667pdf (Last accessed 16-Oct-2020)

[15] http://www.ti.com/product/LM35 (Last accessed 16-Oct-2020)

[16] http://www.akcp.com/products/intelligent-sensors/fuellevel-sensor/ (Last accessed 16-Oct-2020)

[17] L. Marchegiani and P. Newman, "Listening for Sirens: Locating and Classifying Acoustic Alarms in City Scenes," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 10, pp. 17087-17096, Oct. 2022, doi: 10.1109/TITS.2022.3158076

[18] A. Meghanani, A. C. S. and A. G. Ramakrishnan, "An Exploration of Log-Mel Spectrogram and MFCC Features for Alzheimer's Dementia Recognition from Spontaneous Speech," 2021 IEEE Spoken Language Technology Workshop (SLT), Shenzhen, China, 2021, pp. 670-677, doi: 10.1109/SLT48900.2021.9383491.

[19] Spoorthy, V., & Koolagudi, S. G. (2023). Bi-level Acoustic Scene Classification Using Lightweight Deep Learning Model. Circuits, Systems, and Signal Processing, 1-20.

[20] Raj Kumar D bhure, K. Saisrikar, Ch. Devayani, D. Shivareddy, Energy Efficiency Routing For Manet Using Residual Energy, International Journal of Multidisciplinary Engineering in Current Research - IJMEC Volume 8, Issue 4, April-2023, http://ijmec.com/, ISSN: 2456-4265.

[21] Mesaros, Annamaria & Heittola, Toni & Diment, Aleksandr & Elizalde, Benjamin & Shah, Ankit & Vincent, Emmanuel & Raj, Bhiksha & Virtanen, Tuomas. (2017). DCASE 2017 CHALLENGE SETUP: TASKS, DATASETS AND BASELINE SYSTEM.

[22] X. Zhao and D. Wang, "Analyzing noise robustness of MFCC and GFCC features in speaker identification," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 2013, pp. 7204-7208, doi: 10.1109/ICASSP.2013.6639061.

[23] Kipyatkova, I. LSTM-Based Language Models for Very Large Vocabulary Continuous Russian Speech Recognition System. In Proceedings of the SPECOM 2019: Speech and Computer, Istanbul, Turkey, 20–25 August 2019; Volume 11658, pp. 219–226.

[24] Basu, S.; Chakraborty, J.; Aftabuddin, M. Emotion recognition from speech using convolutional neural network with recurrent neural network architecture. In Proceedings of the 2017 2nd International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 19–20 October 2017; pp. 333–336.

[25] Jo, A.-H.; Kwak, K.-C. Speech Emotion Recognition Based on Two-Stream Deep Learning Model Using Korean Audio Information. *Appl. Sci.* 2023, *13*, 2167. https://doi.org/10.3390/app13042167

[26] Piczak, K.J. ESC-50: Dataset for Environmental Sound Classification. Available online: https://github.com/karolpiczak/ESC-50

[27] Gemmeke, J.F., Ellis, D.P.W., et al. (2017). "Audio Set: An Ontology and Human-Labeled Dataset for Audio Events." In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 776-780). DOI: 10.1109/ICASSP.2017.7952261.

[28] AudioSet.Availableonline: https://research.google.com/audioset/download.html

[29] Chu, S.; Narayanan, S.; Kuo, C.C.J. Environmental sound recognition with time-frequency audio features. IEEE Trans. Audio Speech Lang. Process. 2009, 17, 1142–1158.

[30] Rawat, A.; Mishra, P.K. Emotion Recognition through Speech Using Neural Network. Int. J. Adv. Res. Comput. Sci. Softw. Eng. 2015, 5, 422–428.

[31] Liu, J.M.; You, M.; Li, G.Z.; Wang, Z.; Xu, X.; Qiu, Z.; Xie, W.; An, C.; Chen, S. Cough signal recognition with Gammatone Cepstral Coefficients. In Proceedings of the 2013 IEEE China Summit and International Conference on Signal and Information Processing, Beijing, China, 6–10 July 2013; pp. 160–164.

[32] I. Toshio, "An optimal auditory filter," in Applications of Signal Processing to Audio and Acoustics, IEEE ASSP Workshop, 1995, pp. 198

[33] B. R. Glasberg and B. C. Moore, "Derivation of auditory filter shapes from notched-noise data," Hearing Res., vol. 47, no. 1, pp. 103–138, 1990.

[34] Hochreiter, S.; Schmidhuber, J. Long Short-term Memory. Neural Comput. 1997, 9, 1735–1780.

[35] Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv 2017, arXiv:1704.04861