

ECE-387 Final Project

Android Auto Head unit

Spring 2021

College of Engineering and Computing

Department of Electrical and Computer Engineering

ECE 387: Embedded Systems

Professor : Peter Jamieson

Lanutoshi Paul

April 30th, 2021

# Introduction

Most new cars today come with advanced infotainment systems that allow for features such as google maps, music applications like Spotify, call and message notifications and a host of other applications that can be downloaded on the go. I got to experience these new features for the first time when I rented a newer car on vacation. I just had to plug my phone into a USB port and voila all the essentials I needed while driving popped up on the infotainment screen. I appreciated how convenient the system was as well as the fact that I was less distracted while driving; I didn't need to touch my phone at all and I found that I was always looking ahead. Having enjoyed having these features while on the road I decided that I would try to implement this system in my 2001 Lexus RX300 for my Embedded systems final project. My car, being older, did not have any phone connectivity and the navigation system was very dated. I chose this project because I thought that as a senior computer engineer in college, I had the skill set to be able to build something that served an immediate need. Having worked on multiple projects and learned a lot from them, none of these were real world applications that I could use in my everyday life.

# Project Research

From the start, it was evident that I would have to spend quite a bit of money on this project. However, since this project would one day be permanently fitted in my car, I decided that it would be worth it. The components are as follows:

1. RaspberryPi 7inch LCD touch screen:

This was ordered from amazon and came with a ribbon cable to connect it to the Raspberrypi and a mount for the same. It cost around $70 on amazon. Here is a link for reference <https://www.amazon.com/Raspberry-Pi-7-Touchscreen-Display/dp/B0153R2A9I/ref=sxin_6?asc_contentid=amzn1.osa.ad2a7585-7957-47d4-8f17-291854f2725a.ATVPDKIKX0DER.en_US&asc_contenttype=article&ascsubtag=amzn1.osa.ad2a7585-7957-47d4-8f17-291854f2725a.ATVPDKIKX0DER.en_US&creativeASIN=B0153R2A9I&cv_ct_cx=raspberry+pi+touchscreen&cv_ct_id=amzn1.osa.ad2a7585-7957-47d4-8f17-291854f2725a.ATVPDKIKX0DER.en_US&cv_ct_pg=search&cv_ct_we=asin&cv_ct_wn=osp-single-source-earns-comm&dchild=1&keywords=raspberry+pi+touch+screen&linkCode=oas&pd_rd_i=B0153R2A9I&pd_rd_r=8a83140c-e982-4330-952d-f79f22ab29d6&pd_rd_w=i4jUt&pd_rd_wg=rv3Bm&pf_rd_p=c5aa77b7-f0ac-4b95-b207-5ef230c60c9b&pf_rd_r=CJP11KVPW1M7B5FY59Y0&qid=1621043470&sr=1-2-64f3a41a-73ca-403a-923c-8152c45485fe&tag=theradar-20>

1. RaspberryPi 3B+:

This is available on Amazon for about $50. Here is the link to Amazon <https://www.amazon.com/ELEMENT-Element14-Raspberry-Pi-Motherboard/dp/B07P4LSDYV/ref=sr_1_3?dchild=1&keywords=raspberry+pi+3B%2B&qid=1621043557&s=electronics&sr=1-3>

1. RaspberryPi camera:

These cost around $30 and are available here: <https://www.amazon.com/Raspberry-Pi-Camera-Module-Megapixel/dp/B01ER2SKFS/ref=sxin_6?asc_contentid=amzn1.osa.2eaef0e1-35bd-4135-b5b3-9bfeb0934cb5.ATVPDKIKX0DER.en_US&asc_contenttype=article&ascsubtag=amzn1.osa.2eaef0e1-35bd-4135-b5b3-9bfeb0934cb5.ATVPDKIKX0DER.en_US&creativeASIN=B01ER2SKFS&cv_ct_cx=raspberry+pi+camera&cv_ct_id=amzn1.osa.2eaef0e1-35bd-4135-b5b3-9bfeb0934cb5.ATVPDKIKX0DER.en_US&cv_ct_pg=search&cv_ct_we=asin&cv_ct_wn=osp-single-source-earns-comm&dchild=1&keywords=raspberry+pi+camera&linkCode=oas&pd_rd_i=B01ER2SKFS&pd_rd_r=f43847f0-08e3-4dd4-b127-db2f6976a4c2&pd_rd_w=XQFJK&pd_rd_wg=OFVdw&pf_rd_p=c5aa77b7-f0ac-4b95-b207-5ef230c60c9b&pf_rd_r=PRVNXXBN5RCRGKM6ZZ46&qid=1621043628&s=electronics&sr=1-1-64f3a41a-73ca-403a-923c-8152c45485fe&tag=theradar-20>

1. 6 foot ribbon cable

The ribbon cable cost around $10 and is available here:<https://www.amazon.com/gp/product/B072HVZYHF/ref=ppx_yo_dt_b_asin_title_o01_s00?ie=UTF8&psc=1>

1. 32 GB SD card

The SD card can be bought at your local grocery store and pharmacy. They are cheaper to buy on Amazon and cost around $10. Here is a link: <https://www.amazon.com/SanDisk-Ultra-UHS-I-Memory-SDSDUNR-032G-GN6IN/dp/B07YFG7T8F/ref=sxin_6?asc_contentid=amzn1.osa.e1285160-c6f6-47fa-96ed-4de251efbdaa.ATVPDKIKX0DER.en_US&asc_contenttype=article&ascsubtag=amzn1.osa.e1285160-c6f6-47fa-96ed-4de251efbdaa.ATVPDKIKX0DER.en_US&creativeASIN=B07YFG7T8F&crid=2L9M8EPV5AUGJ&cv_ct_cx=32gb+sd+card&cv_ct_id=amzn1.osa.e1285160-c6f6-47fa-96ed-4de251efbdaa.ATVPDKIKX0DER.en_US&cv_ct_pg=search&cv_ct_we=asin&cv_ct_wn=osp-single-source-earns-comm&dchild=1&keywords=32gb+sd+card&linkCode=oas&pd_rd_i=B07YFG7T8F&pd_rd_r=4091198c-11de-42bd-b5da-bb94de76c1a1&pd_rd_w=tH2vl&pd_rd_wg=5H6qF&pf_rd_p=c5aa77b7-f0ac-4b95-b207-5ef230c60c9b&pf_rd_r=G6TSADVRD6RXZSFJ9JJ8&qid=1621043913&s=electronics&sprefix=32+gb+sd%2Celectronics%2C180&sr=1-3-64f3a41a-73ca-403a-923c-8152c45485fe&tag=tgl0a3-20>

1. USB to USB type-C cable

These can be found at the local grocery store or pharmacy. Here is a link to Amazon: <https://www.amazon.com/Charger-Braided-Charging-Compatible-Samsung/dp/B0794M53HQ/ref=sr_1_3?dchild=1&keywords=usb+type+c&qid=1621044034&s=electronics&sr=1-3>

1. 5V power outlet

This can be bought from Amazon as well: <https://www.amazon.com/Sockets-Cigarette-Splitter-Separate-Compatible/dp/B07P7TMQCL/ref=sr_1_5?dchild=1&keywords=5v+power+supply+cigarette+lighter&qid=1621044101&s=electronics&sr=1-5>

The only components that I had to purchase were the touch screen and the 6 foot ribbon cable for the camera. The others were borrowed from my Senior capstone advisor. Having ordered the required hardware, it was now time to look at the software that would be run. From my research, I found that there were two open source versions of Android Auto that could be used. One of them was Crankshaft and the other was OpenAuto. These could be run within the Raspbian OS or could be flashed onto the SD card as a stand alone OS. I also learnt that since these were open source, there would be a lot of configuration tweaks that would need to be done in order for them to function properly.

# Implementation Process

Once I had received all the hardware components, it was now time to move on to implementing the software. After reading reviews on the customizability of both Crankshaft and Open Auto, I learned that developers had good things to say about Open Auto and hence I decided to go with it. I also intended to install Open Auto within the Raspbian OS as this would allow for an easier UI experience especially in the developing stages. This was done in the following manner:

1. Raspbian was downloaded and flashed onto the SD card.
2. An empty text file titled “sh” was created in the boot folder of the SD card to enable remote access to the Pi.
3. Since I did not have a keyboard, monitor or mouse to interface with the Raspberrypi, I connected to it remotely using Putty and VNC viewer.
4. On accessing Raspbian, commands were entered into the terminal window to download Open Auto. There were as follows:
5. Downloading the script:

$ git clone https://github.com/novaspirit/androidauto\_rpi\_install

1. Switching to the download directory:

$ cd androidauto\_rpi\_install

1. Changing permission in order to execute:

$ chmod +x install.sh<br>

1. Final installation:

$ ./install.sh

Everything ran perfectly until step d. The RaspberryPi would crash and the installation would not work. I looked at various forums to come up with a fix but was unable to find one. I also tried manually installing the various dependencies but encountered the same error. I assumed that the processing was too heavy for the Pi since I was running two heavy operating systems so I decided to try flashing Open Auto directly onto the SD card. On boot up it was looking promising but then I was asked to enter a user name and id before it fully booted up. I learnt that there were tweaks to be made in the configuration settings. Having tried all the suggested changes, I was still not able to get past the GUI asking for user name and password.

The next option was to try and install Crankshaft instead. This came with its challenges as I was not able to find an OS image that would work within Raspbian. Without Raspbian it would be a lot more difficult to interface with the Pi. Everything would have to be done through the terminal that is connected virtually. I went with installing it as a stand alone OS and it ran just fine. The GUI popped up and on connection to my phone, the screen was showing all the necessary information.

The next step was working on the rear view camera. Crankshaft comes with built in libraries that allow camera functionality. One of these commands to test the camera is “crankshaft rearcam show” which displays a live video stream. Initial testing failed and all I saw was a blank screen. I learnt that this was because there was an issue with the “cam\_overlay”. One of the solutions was to check the video format using “v4l2-ctl --all” and then tweakng cam\_overlay but I was not able to fix it. I then tried using basic RaspberryPi terminal commands and it worked just fine.

After this I had to develop a program for the rearview camera from the terminal. I decided to use python to accomplish this task. There were issues with installing python as some of the files on the disk were read only. This was fixed by using “csmt boot unlock” which allowed for some files to be accessed in the boot folder. The python program that was developed for testing was very straightforward to implement.

# Future Work

This is a project that is still under development and I plan to make a mount for the LCD screen on the dashboard to start off. The ribbon cable that was purchased for the camera was too short so it was implemented as a dash cam. I plan to get an HDMI adapter and run the wiring to the rear camera through the plastic cladding in the cabin. Overall this project taught me a lot about the nitty gritty of producing a real world embedded system. There are a lot of costs that one might not consider and parts take a while to reach you. While using open source software a lot of unexpected bugs come up that eat away at your time. However, this is a project that I hope to see in its full implementation over the summer.

Here are the pertinent links:

Github: <https://github.com/paull-oh/Android-Auto-Head-Unit>

Youtube: <https://youtu.be/hLWBfuTDK08>