System Design I

Straight Outta COMPE

Functional Decomposition

**Authors:**

Zack Rauen

Juan Merchan

Jayesh Bokhiria

# Revision History

|  |  |  |
| --- | --- | --- |
| Date | Rev. Author | Revision |
| 12-Sept-2015 | J. Bokhiria | Initial layout of milestone 2. |
| 16-Sept-2015 | J. Bokhiria | Updated the layout and complete section 5. |
| 17-Sept-2015 | Z. Rauen | Reformat and complete section 1 & Complete section 2 with new diagrams. |
| 17-Sept-2015 | J. Merchan | Completed section 3. |
| 17-Sept-2015 | Z. Rauen | Cleaned up selected parts of section 3. |
| 19-Sept-2015 | J. Bohkiria | Updated section 4 and completed interfaces. |
| 19-Sept-2015 | Z. Rauen | Finalized formatting and merging documents. |
| 19-Sept-2015 | Z. Rauen & J. Bokhiria | Error checking and corrections |

Table of Contents

[Revision History II](#_Toc430448567)

[1. Introduction 1](#_Toc430448568)

[1.2. Purpose 1](#_Toc430448569)

[1.3. Scope 1](#_Toc430448570)

[1.4. Definitions, Acronyms and Abbreviations 1](#_Toc430448571)

[1.4.1. Definitions 1](#_Toc430448572)

[1.4.2. Acronyms 1](#_Toc430448573)

[1.4.3. Abbreviations 2](#_Toc430448574)

[1.5. References 2](#_Toc430448575)

[1.5.1. Documents 2](#_Toc430448576)

[1.5.2. Standards 3](#_Toc430448577)

[1.6. Overview 3](#_Toc430448578)

[2. System Operation 3](#_Toc430448579)

[2.1. System Description 3](#_Toc430448580)

[2.1.1. Line Tracker 4](#_Toc430448581)

[2.1.2. Marker Identifier 4](#_Toc430448582)

[2.1.3. User Interface 4](#_Toc430448583)

[2.2. Block Diagrams 4](#_Toc430448584)

[2.2.1. Overall 4](#_Toc430448585)

[2.2.2. Microcontroller 5](#_Toc430448586)

[3. Functional Descriptions 5](#_Toc430448587)

[3.1. Top Level 5](#_Toc430448588)

[3.1.1. Microcontroller (Freescale FRDM-K64F) 5](#_Toc430448589)

[3.1.2. LCD and User Interface 5](#_Toc430448590)

[3.1.3. Wheel Servos 5](#_Toc430448591)

[3.1.4. Blinkers 1 and 2 6](#_Toc430448592)

[3.1.5. Buzzer 6](#_Toc430448593)

[3.1.6. Line Camera 6](#_Toc430448594)

[3.1.7. Marker Camera 6](#_Toc430448595)

[3.1.8. Camera Servo 6](#_Toc430448596)

[3.1.9. Bluetooth Adapter 6](#_Toc430448597)

[3.2. Microcontroller 6](#_Toc430448598)

[3.2.1. I2C Controller 6](#_Toc430448599)

[3.2.2. LCD Driver 7](#_Toc430448600)

[3.2.3. Camera Driver 7](#_Toc430448601)

[3.2.4. Mode Controller 7](#_Toc430448602)

[3.2.5. Driving Control 7](#_Toc430448603)

[3.2.6. UART 7](#_Toc430448604)

[4. Specifications 8](#_Toc430448605)

[4.1. Interfaces 8](#_Toc430448606)

[4.1.1. System 8](#_Toc430448607)

[4.1.2. Microcontroller/Software 8](#_Toc430448608)

[4.2. Components (COTS) 9](#_Toc430448609)

[4.2.1. FRDM-K64G Board 9](#_Toc430448610)

[4.2.2. Car Chassis 13](#_Toc430448611)

[4.2.3. CMUCam5 Pixy Camera 14](#_Toc430448612)

[4.2.4. Line Scan Camera (128x1 sensor array) 18](#_Toc430448613)

[4.2.5. Servo 22](#_Toc430448614)

[4.2.6. Hall Effect Sensor (Speed Encoder) 23](#_Toc430448615)

[4.2.7. Battery 23](#_Toc430448616)

[4.2.8. Full-Bridge PWM Motor Driver 24](#_Toc430448617)

[4.3. Optional Components 26](#_Toc430448618)

[4.3.1. LCD 26](#_Toc430448619)

[4.3.2. Bluetooth 27](#_Toc430448620)

[4.3.3. LED 27](#_Toc430448621)

[4.3.4. Buzzer 28](#_Toc430448622)

1. Introduction
   1. Identification

This document shall be known as the System Design I and has an identifier of SOC-SysDes-1 to be referenced in future documents.

* 1. Purpose

The purpose of this document is to illustrate the design to the team in order to allow an efficient and quality driven implementation as well as having a sound design.

* 1. Scope

This documents scope is limited to the system design of the Intelligent Car Version 3. This means that implementation choices are avoided aside from the constraints already defined in *SOC-SysReq* including the use of a microcontroller. Implementations not defined there are thus avoided in this document.

* 1. Definitions, Acronyms and Abbreviations
     1. Definitions
        1. **Car:** The main product, the Intelligent Car Version 3.
        2. **Horn:** The buzzer used on the car.
        3. **Marker:** A cylindrical pipe that represents commands to the car.
        4. **Mode:** A constant state of the system, in this case of the car. This can be changed by input from the user (1.4.1.10).
        5. **Posts:** See Marker (1.4.1.3).
        6. **Product:** The entire system (1.4.1.9) as a finished design.
        7. **System:** This refers to the entire system in use, in this case the car as well as the personal computer that the car connects to in order to use data logging. Additionally this will include the optional mobile phone that acts as a remote as well.
        8. **User:** The person(s) that will be using or interacting with the product.
        9. **We:** The members of Straight Outta CompE.
     2. Acronyms
        1. **I/O:** Input/Output
        2. **COTS:** Commercial Off The Shelf.
        3. **I2C:** Optionally I2C. Inter-Integrated Circuit.
        4. **IEEE:** Institute of Electrical and Electronics Engineers.
        5. **PC:** Personal Computer.
        6. **SPI:** Serial Peripheral Interface Bus.
        7. **SOC:** Straight Outta CompE.
     3. Abbreviations
        1. **A:** ampere(s).
        2. **Bps:** bits per second.
        3. **C:** Celsius.
        4. **kg:** kilogram.
        5. **KHz:** kilohertz.
        6. **MHz:** megahertz.
        7. **mm:** metric millimeters.
        8. **ms:** milliseconds.
        9. **nm:** nanometer.
        10. **oz:** ounces.
        11. **V:** volts.
        12. **sec:** seconds.
        13. **in**: imperial inches.
        14. **px:** pixels.
        15. **ft:** feet.
        16. **cm:** metric centimeters.
        17. **°:** degree(s).
        18. **Mag:** magnitude.
  2. References
     1. Documents
        1. Lecture Material for EE416/464 Fall 2015, Clarkson University September 7 to September 17, 2015
        2. Product Concept Fall 2015, Abul Khondker
        3. System Requirements Specification, *SOC-SysReq*
        4. Car chassis

http://www.robotshop.com/en/dfrobot-4wd-arduino-mobile-platform.html

* + - 1. LiPo battery

http://www.amazon.com/gp/product/B002AK38TS?psc=1&redirect=true&ref\_=oh\_aui\_detailpage\_o06\_s00

* + - 1. CMUCam5 Pixy Port Pinouts

http://cmucam.org/projects/cmucam5/wiki/Port\_Pinouts

* + - 1. CMUCam5 Pixy, Using Color Codes

http://cmucam.org/projects/cmucam5/wiki/Using\_Color\_Codes

* + - 1. CMUCam5 Pixy, Porting Guide

http://cmucam.org/projects/cmucam5/wiki/Porting\_Guide

* + - 1. CMUCam5 Pixy, Powering Pixy

http://cmucam.org/projects/cmucam5/wiki/Powering\_Pixy

* + - 1. Camera mount w/ two servo based pan-tilt mechanisms

http://www.robotshop.com/en/lynxmotion-pan-and-tilt-kit-aluminium2.html#What

* + - 1. HS-422 Servo Motor

http://www.robotshop.com/en/hitec-hs422-servo-motor.html

* + - 1. Line Scan Camera with TAOS linear sensor array

https://community.freescale.com/docs/DOC-1030

* + - 1. H-bridge specs – Full-Bridge PWM Motor Driver

moodle.clarkson.edu/pluginfile.php/321618/course/section/66133/A4973-Datasheet.pdf

* + - 1. SainSmart I2C IIC Interface RGB LED Screen LCD 1602 + Keypad For Raspberry Pi

http://www.amazon.com/SainSmart-Interface-Screen-Keypad-raspberry/dp/B00SMHSTL0/ref=cm\_cr\_pr\_product\_top?ie=UTF8

* + - 1. Hall Effect sensor (speed encoder)

http://www.robotshop.com/en/dfrobot-wheel-encoders-dfrobot-3pa-4wd-rovers-2pk.html#Specifications

* + - 1. Diffused Green 10mm LED

http://www.adafruit.com/products/844

* + - 1. Piezo Buzzer – PS1240

http://www.adafruit.com/products/160

* + - 1. SparkFun Bluetooth Modem - BlueSMiRF Silver

https://www.sparkfun.com/products/12577

* + 1. Standards
       1. I2C
       2. SPI
       3. IEEE 802.15
       4. UART
  1. Overview

The rest of this document discusses the operation of the system at an overall level in section 2. Also within section two are the block diagrams, one for the overall system and another for the software that shall be implemented on the microcontroller. Functional descriptions of each individual block can be found in section 3 and are separated by scope (system versus software). The last section, section 4, holds the specifications for the interfaces between the blocks as well as the specifications for COTS blocks. The COTS found in section 4 will correspond to those defined in *SOC-SysReq*.

1. System Operation

This section briefly describes the functionality of the system at the highest level. For slightly more detailed information, check section 3.

* 1. System Description

The system to be designed is a third generation "intelligent” car, the system should be able to follow a black line against a white background. The system should also be able to navigate autonomously around a track and also be able to find the track if placed outside the track. The system should use two vision sensors to process information and make decisions based on its location in the track. One sensor can be used to detect the markers along the track. These markers provide instructions to the car system to turn left, right, pause, stop or continue when it comes to an intersection on the track.

The car system must be able to perform three basic data processing functions. These functions not only satisfy the project conditions but also allow the user to interact with the car system for better performance.

* + 1. Line Tracker

The car system must be able to identify its location by using the camera sensor to detect a black line in the track. Once the car system sees that Line it must process that in data coming in and make the necessary adjustments to follow the line.

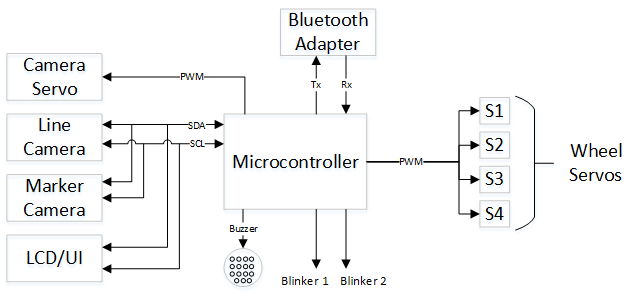
* + 1. Marker Identifier

The second vision detector camera must be able to identify the color in the markers placed along the track. Then the car system must be able to process the data and detect the color patter in the maker. The car system must use the color pattern to pick an action/ direction to follow.

* + 1. User Interface

The car system user interface must allow the user to easily control the car. Using this interface the user must be able to pick a challenge mode and also obtain basic information about the status of the car.

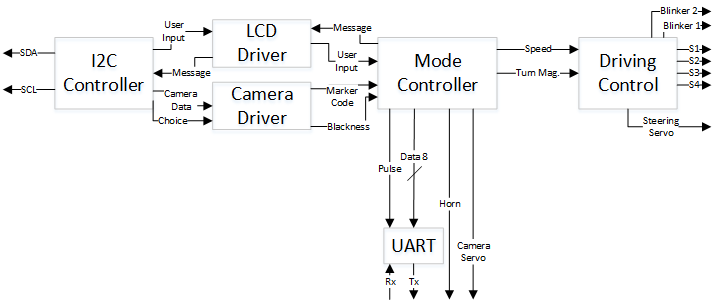
* 1. Block Diagrams
     1. Overall



**Figure 2.2.1.1:** System level block diagram.

Figure 2.2.1.1 shows the overall system as a set of logical blocks. The main design decisions can be understood from here. This includes the decision for the use of the I2C standard as well as using the microcontroller as the main control unit. The blocks are described in more detail in section 3.1

* + 1. Microcontroller



**Figure 2.2.1.2:** Block diagram of the functionality within the microcontroller.

Figure 2.2.1.2 is a representation of the software blocks that should be modeled within the microcontroller. Again, the main design decisions can be seen from the diagram such as having the horn/buzzer directly related to the mode as described in section 3.1.5. More information on the blocks can be found in section 3.2.

1. Functional Descriptions
   1. Top Level
      1. Microcontroller (Freescale FRDM-K64F)

The microcontroller from Freescale FRDM-K64F will handle user and environmental inputs from the different block components to then determine a series of actions to be outputted to the car system. More information on that can be found in section 3.2.

* + 1. LCD and User Interface

The user interface is composed of a series of buttons and an LCD to interact with the user smoother when picking an operation mode. The LCD will display the current operational mode then using the buttons the user will be able to change between modes. The buttons will also allow further interaction with the user by allowing the user to customize options specific to the mode. The LCD and buttons will communicate back and forth with the Microcontroller using the I2C communication protocol.

* + 1. Wheel Servos

The movement of the car will be handled using 4 servo motors (S1-S4) each driven by a PWM signal generated by the Microcontroller. The microcontroller will determine the direction the car must follow (Straight, left, right, backwards) and then power the correct series of motors to perform this movement.

* + 1. Blinkers 1 and 2

The blinker lights (LED lights) will be operated by the microcontroller using an I/O signal. The microcontroller will use data from the cameras to determine if the car must turn left or right and then select which blinker to turn on.

* + 1. Buzzer

The buzzer will be operated using an I/O signal from the microcontroller. The tune of the buzzer will variate depending on the operation mode of the car system or if there is an action that requires an action from the user. For instance, during discovery mode, the buzzer will beep slowly when the car has not discovered the track and will become quicker as the car thinks it has detected the track.

* + 1. Line Camera

The line camera will process and transmit data to the microcontroller using the I2C communication Standard. The Microcontroller will then process the data and determine were the track/Line is in relation with the car. The camera will be mounted in the front center of car to make following the line easier.

* + 1. Marker Camera

The marker camera will process and transmit data to the microcontroller using I2C communication Standards. The camera will detect different color marker along the track and then transmit this information to the microcontroller to determine a further action for the car system. The markers will dictate the direction the car must follow. This camera will also be placed on a servo as demonstrated in 3.1.8.

* + 1. Camera Servo

The Marker camera will be mounted on a servo, this will improve the cars ability to find markers along the track or to find the track itself when placed outside the track area for the discovery mode. The servo will be operated using a PWM signal generated by the microcontroller, data from the camera will determine the direction (rotational angle) of the servo.

* + 1. Bluetooth Adapter

The Bluetooth Adapter will be operated using transmit/receive (TX/RX) communication protocols between the car system, a PC and a remote control. This Bluetooth adapter will be used to transmit image data to a local computer and also to operate the car system using a remote control (phone Application).

* 1. Microcontroller
     1. I2C Controller

The I2C Controller uses two input/output signal (SDA and SCL) to transmit data from the cameras to the Camera Driver and from the user (bottom input) to the LCD driver.

The camera data is transmitted from the I2C Controller to the Camera driver using a Camera Data signal and a Choice (Camera selection) signal. Using two signal helps determine which camera data is been transmitted (Marker Camera or Line Camera).

The I2C Controller outputs a User Input signal containing the operation mode selection to the LCD Driver, then I2C Controller takes an input signal Message which contains the message to be displayed on the LCD.

* + 1. LCD Driver

The LCD Driver forwards the desired action signal to the Mode Controller, then using an input signal from the Mode Controller (Message) determines the message to be displayed on the LCD and output the Message signal. The driver, rather than just forward the user input directly, interprets the user input and gives a simple command to the mode controller in order to simplify and modularize.

* + 1. Camera Driver

The camera driver uses two input signals Camera data and Choice to determine which camera the data belongs to and output a Marker Code signal or Blackness reading signal to the Mode Controller. The Marker Code signal will indicate to the mode controller if there are any markers indicating how to turn at an intersection. The blackness signal will tell the mode controller dark the image is in terms of how similar it is to what it thinks is the track.

* + 1. Mode Controller

The Mode controller will be the main data processing block in the software design. The mode controller will take multiple data inputs from the Camera Driver and the LCD driver (User logic) to perform output actions in the car system.

The mode controller will use the User input signal from the LCD driver to determine an operation mode, then output the Message signal containing the LCD display selection back to the LCD driver.

The Marker and Blackness signals from the Camera Driver are user used by the Mode selector to determine a speed and direction for the car system. After processing the data the Mode Controller will output two signals Speed and Turn Magnitude to the Driving controller. Using the same data, the Mode Controller will determine the Tune of the Buzzer, and the direction that the Marker camera must face using the Camera Servo output signal.

The Mode Controller will also generate a pulse and an 8-bit data package (Data 8) which will then be sent to the UART.

* + 1. Driving Control

The Driving controller uses two input signals from the Mode Controller a Speed and Turn Magnitude. Using this two signals the Driving controller breaks down the speed of the each servo motor (S1-S4). After this the Driving Controller determines the direction the car must turn and output the signal of Steering Servo. Using the same direction data the Driving Controller will pick a Blinker 1 or 2 (Left or Right) and output the corresponding signals.

* + 1. UART

The UART controller takes the Pulse and Data Signals from the Mode Controller and then sends the data to the Bluetooth Adapter using a Tx signal. The Bluetooth adapter will generate an Rx signal which will be used to drive the car using a remote control device.

1. Specifications
   1. Interfaces
      1. System

Microcontroller is the center unit in terms of the computer for the car. All the input and output signals comes from the microcontroller. A general overview of top level interfaces is as follows:

* + - 1. An output signal of PWM from microcontroller is an input signal of camera servo (line camera). It controls the camera’s position (horizontal and vertical) by controlling the servo motors of camera mount pan-tit.
      2. A second output signal of PWM from microcontroller is an input signal of wheel servos. It will regulate the speed and the direction of the wheels.
      3. An output buzzer signal from microcontroller is an input of the Buzzer. It will buzz or beep to alert when the track is detected during the discovery challenge and completion of the course.
      4. A two signals blinker 1 and blinker 2 from microcontroller is an input to the LED lights. When the car come to intersection a blinker will go which indicate the direction the car will drive.
      5. A transmit (Tx) signal from microcontroller to the bluetooth adapter. An output signal, receiver (Rx) from bluetooth adapter is an input of the microcontroller. This is a bluetooth communication where data is sent and received by microcontroller.
      6. Two signals serial data (SDA) and serial clock (SCL) is an input and output of between four components microcontroller, line camera, marker camera and LCD/user interface (UI). This method of communication is called Inter-Integrated Circuit (I2C).
    1. Microcontroller/Software
       1. Driving control

A two output signals Speed and Turn mag. from the controller are input of driving controller. Driving Control has a seven output signals, a Steering Servo, four signals to wheel servo and two signals to blinkers. This is a description of driving mechanism of the car in terms of interface of various components communicating together.

* + - 1. Buzzer

A Buzzer signal is an output of the controller, see **4.1.1.3** for detailed behavior.

* + - 1. Camera Servo

A Camera servo signal is an output of the controller, see **4.1.1.1** for detailed behavior.

* + - 1. UART

A two output signals Pulse and Data (8 bits) from the controller are input of UART. UART has one input receive (Rx) and one output transmit (Tx), see **4.1.1.5** & **3.2.6** for detailed behavior.

* + - 1. I2C Controller

There are intermediate components between controller and I2C Controller.

* + - 1. LCD Driver

LCD driver takes in the Message signal which the output signal of the controller and sent it to the I2C controller. It also takes an output signal User Input from the I2C then sends it to the controller. Other words, a message from the controller goes through the LCD driver to I2C controller. Similarly, a user input from I2C controller goes through LCD driver to the controller.

* + - 1. Camera Driver

A camera driver takes in two outputs Camera data and Choice from the I2C controller. It will out output two signals Marker code and Blackness to the mode controller. In other words, the camera driver acts like a decoder which takes in the output from the I2C controller, convert it, and send it to the mode controller in a format that is more useful. More specifically, the Marker Code simply tells which known marker combination was detected. The blackness signal is a representation of how black the image is, which allows the mode controller to decide how far the center of the car is from the track.

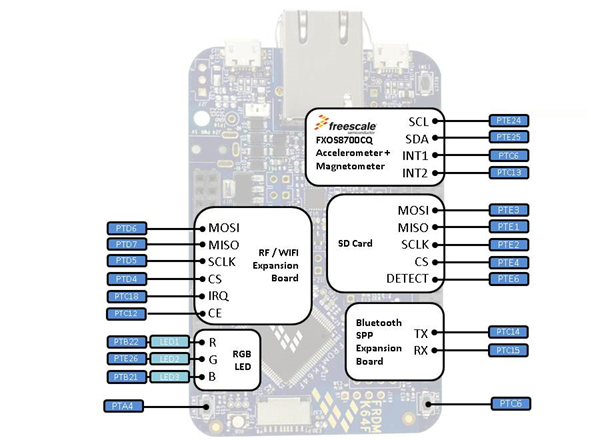
* 1. Components (COTS)
     1. FRDM-K64G Board

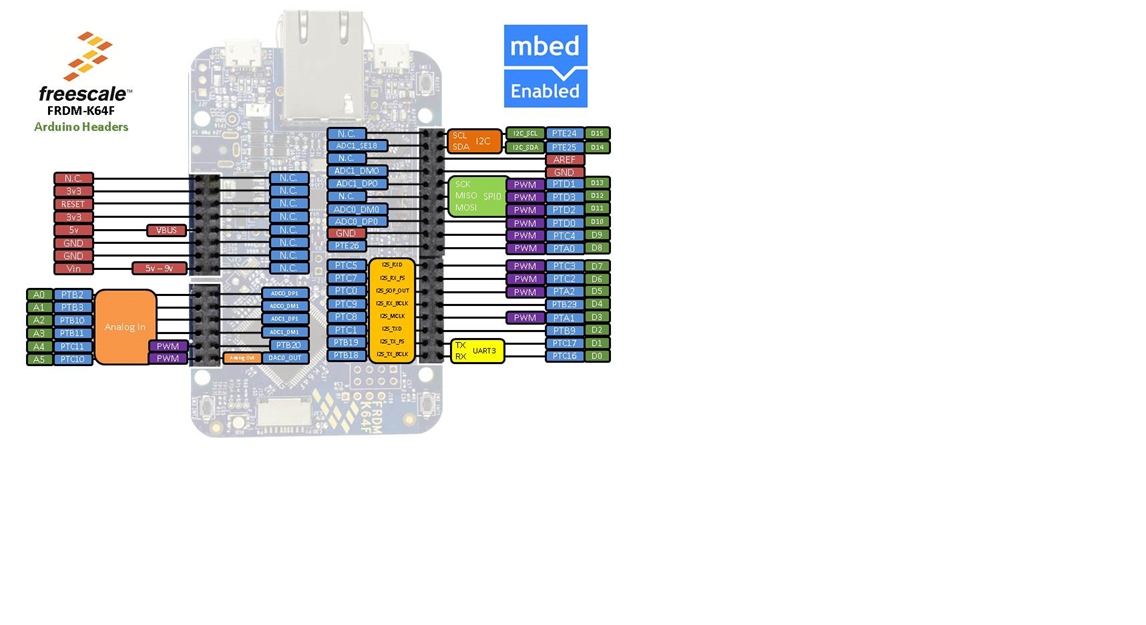
The features that include in FRDM-K64G Board are shown below in table 1.

|  |  |
| --- | --- |
| Technical Values |  |
| MK64FN1M0VLL12 MCU | 120 MHz, 1 MB flash memory, 256 KB RAM, low-power, crystal-less USB, and 100 Low profile Quad Flat Package (LQFP) |
| Dual role USB interface with micro-B USB connector |  |
| LED | RGB |
| Accelerometer and magnetometer | FXOS8700CQ |
| 2 push buttons |  |
| power supply | OpenSDAv2 USB, Kinetis K64 USB, and external source |
| MCU input/output through Arduino™ R3 compatible I/O (max IOs 66) connectors |  |
| Programmable OpenSDAv2 debug circuit supporting the CMSIS-DAP Interface software that provides: | Mass storage device (MSD) flash programming interface  CMSIS-DAP debug interface over a driver-less USB HID connection providing run-control debugging and compatibility with IDE tools  Virtual serial port interface  Open source CMSIS-DAP software project |
| Ethernet |  |
| Secure Digital High Capacity (SDHC) |  |
| Add-on RF module: nRF24L01+ Nordic 2.4GHz Radio |  |
| Add-on Bluetooth module: JY-MCU BT board V1.05 BT |  |

**Table 4.2.1.1:** FRDM-K64G specifications [1]

Below figure a shows the pin diagram of microcontroller.

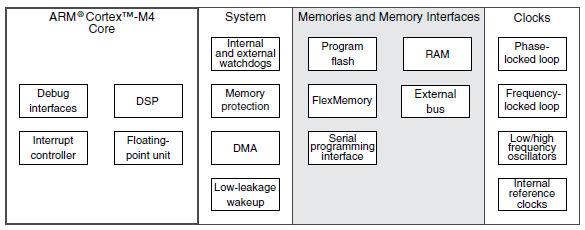
****

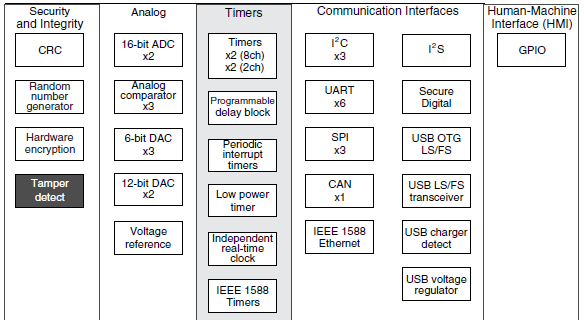
****

**Figure a:** FRDM-K64F pin diagram

**K64F Family Guide**

Below figure b shows the diagram of K64F core module and peripherals





**Figure b:** K64F Family Guide

**K64F Datasheet**

This datasheet contains electrical specifications which includes timing, voltages, currents, and peripheral specifications, this is shown below in table 2.

|  |  |
| --- | --- |
| Technical Values |  |
| Clock | |  |  | | --- | --- | | Description |  | | System and core clock | 120MHz | | Bus clock | 60MHz | | FlexBus clock | 50MHz | | Flash clock | 25MHz | | USB FS clock | 48MHz | | I2C master clock | 25MHz | |
| Electrical Characteristics | |  |  |  | | --- | --- | --- | | Description w/ units | Min. | Max. | | Digital supply voltage () (V) | -0.3 | 3.8 | | Digital supply current (mA) | -- | 185 | | Digital input voltage (V) | -0.3 | 5.5 | | Analog input voltage (V) | -0.3 | +0.3 | | Analog supply voltage (V) | -0.3 | -0.3 | | RTC battery supply voltage (V) | -0.3 | 3.8 | | Supply voltage (V) | 1.71 | 3.6 | | Analog supply voltage (V) | 1.71 | 3.6 | |

**Table 4.2.2.1:** FRDM-K64G specifications [1]

* + - 1. Analog to Digital Converter (16-bit ADC SAR)

The K64 has two modules, ADC0 and ADC1 and below is list of main features.

|  |  |
| --- | --- |
| Technical Values |  |
| Modules supports | 4 differential pair analog inputs  24 single-ended analog inputs |
| Triggers | Software and hardware |
| Single or continuous conversion, and automatic return to idle after single conversion |  |
| Configurable sample time and conversion speed/power |  |
| ADC on any of the software selectable channels |  |
| All modes perform conversion by successive approximation algorithm |  |
| Conversion complete |  |
| Input clock selectable from up to four sources |  |
| Hardware average function |  |
| Self-calibration mode |  |
| Registers | Three groups of ADC status and control registers: ADCx\_SC1n, ADCx\_SC2, and ADCx\_SC3  Two configuration Register: ADCx\_CFG1, and ADCx\_CFG2  Data result register (ADCx\_Rn)  Compare value registers (ADCx\_CVn)  Offset correction register (ADCx\_OFS)  Gain Registers |
| Selectable voltage reference | External or alternate |

**Table 3:** FRDM-K64G ADC specifications [1]

This is a generic description of the K64F and ADC in terms of specifications and its requirements, for a complete datasheet see section 1.5.1 reference 1.

* + 1. Car Chassis

The DFRobot 4WD Arduino-compatible Platform with Encoders as known as the car chassis is built of high-strength aluminum alloy body material. It includes a 4x driver motors, 4x wheels, and a complete chassis with counting hardware. It also includes a second level which allows to add more electronics and has a space for a standard servo motor. The visual car chassis is shown below in figure a.



**Figure a:** DFRobot 4WD Arduino-compatible Platform with Encoders [4]

The following components will be placed on the car chassis a microcontroller, LEDs, bluetooth transmitter, 2 cameras, LCD display, and a horn. Here is list of physical and electrical properties of car chassis

|  |  |
| --- | --- |
| Technical Values |  |
| Compatible with Hitec 311, 422 servo and Pan/Tile Kit |  |
| Infrared sensors switch installed |  |
| Voltage range | +4.5 to 6V |
| Current load | 1,200mA |
| Motor type | 130 |
| Rotational speed of | 10,000 revolutions/minute |
| Gearbox reduction ratio | 1:120 |
| Maximum speed | 68cm/s |
| Wheel diameter | 65mm |
| Dimensions | length 230mm, width 185mm, and high 110mm |
| Weight | 614 grams (without batteries) |
| Maximum load | 800g |

**Table 4.2.2.1:** Car Chassis specifications [4]

* + 1. CMUCam5 Pixy Camera

Here is list of technical specification for CMUCam5:

|  |  |
| --- | --- |
| Technical Values |  |
| Processor | NXP LPC4330, 204 MHz, dual core |
| Image sensor | Omnivision OV9715, 1/4", 1280x800 |
| Lens field-of-view | 75 degrees horizontal, 47 degrees vertical |
| Lens type | standard M12 (several different types available) |
| Power consumption | 140 mA typical |
| Power input | USB input (5V) or unregulated input (6V to 10V) |
| RAM | 264K bytes |
| Flash | 1M bytes |
| Available data outputs\* | UART serial, SPI, I2C, USB, digital, analog |
| Dimensions | 2.1" x 2.0" x 1.4 |
| Weight | 27 grams |
| Processor | NXP LPC4330, 204 MHz, dual core |
| Image sensor | Omnivision OV9715, 1/4", 1280x800 |
| Lens field-of-view | 75 degrees horizontal, 47 degrees vertical |
| Lens type | standard M12 (several different types available) |
| Power consumption | 140 mA typical |
| Power input | USB input (5V) or unregulated input (6V to 10V) |
| RAM | 264K bytes |

**Table 4.2.3.1:** CMUCam5 specifications [6-9]

\* Use I2C for output data type

* + - 1. Pixy Serial Protocol

The pixy camera will output the detected object’s data using I2C serial communication. It will automatically updates or sent new data every 20ms. I2C has a multi-drop 2-wire port (pins 5 I2C SCL, and I2C SDA of the I/O connector, see figure a and b) that allows a single master to communicate with up to 127 slaves (up to 127 Pixys). The I2C interface operates as an I2C slave and requires polling. There are weak 4.7k pullups to 5V on data (SDA) and clock (SCL) signal, via R14 and R15. These signals are 5V tolerant.

|  |  |
| --- | --- |
| **Figure a:** Reverse side of Pixy [6-9] | **Figure b:** Input/output ports [6-9] |

* + - 1. Analog and digital output

Pixy has a single analog (DAC) output, so there are two modes for analog/digital output. Mode 4 outputs the x value of the center of the biggest detected object to pin 3 of the I/O connector. The analog output is 0V if object is on the far left of the image and 3.3V is object on the far right. Mode 5 outputs the y value of the biggest detected object to pin 3 of the I/O connector. Pin 1 goes high (3.3V) when an object is detected, and low (0V) when no object is detected. The analog output is 0V if object is on the bottom of the image and 3.3V is object on the top. Pixy's digital output is 0 to 3.3V logic and can source/sink 5 mA. Pixy's analog (DAC) output ranges between 0 to 3.3V with roughly 200 ohm impedance. [6-9]

* + - 1. Using color codes

Pixy will be used to detect a color coded (two color tags) posts, then it will decode it to determine if it’s a post or not. Once a post’s color is captured the will store it in the memory feature which will eliminate false detection from other color detected and the car will not follow the command from that color. The post’s angle estimate, decoded color codes, position, and sizes are processes at 50 frames per second. Other words, the pixy processes an entire 640x400 image frame every 1/50th of a second (20ms). Below table 4.2.3.3.1 shows the color code objects are returned as special object blocks.

|  |  |  |
| --- | --- | --- |
| **Bytes** | 16-bit words | Description |
| 0, 1 | 0 | sync: 0xaa55=normal object, 0xaa56=color code object |
| 2, 3 | 1 | checksum (sum of all 16-bit words 2-6) |
| 4, 5 | 2 | color code number |
| 6, 7 | 3 | x center of object |
| 8, 9 | 4 | y center of object |
| 10, 11 | 5 | width of object |
| 12, 13 | 6 | height of object |
| 14, 15 | 7 | angle of object (only applies to color code objects |

**Table 4.2.3.3.1:** Color code object block [6-9]

According to the above table the sync code is different (0xaa56 instead of 0xaa55) and instead of the signature number (word 1) returning a single number between 1 and 7, color code object blocks return a an octal (base 8) number with each digit being the signature number of the detected tag (16 bits can represent up to 5 octal digits). The four color code mode parameter for the pixy are following disabled, enabled, color code only, and mixed.

* + - 1. Control data sent to Pixy

The objects in each frame are sorted by size, the largest objects are sent first using I2C serial communication. I2C operates in slave mode and rely on pulling to receive updates. In case, when pixy does not detect the posts it will sends zeros. Each data is sent in an object block, see above table 4.2.3.3.1. All the values for the posts are 16-bit words, sent least-significant byte first (little endian). For example, when sending the sync word 0xaa55, Pixy sends 0x55 (first byte) then 0xaa (second byte). There will be control data sent to pixy to control the pan/tit servos, adjust the camera brightness and set the LED color. Below tables shows the pan/tilt servo, camera brightness, and LED control.

|  |  |  |
| --- | --- | --- |
| **Bytes** | 16-bit words | Description |
| 0, 1 | y | servo sync (0xff00) |
| 2, 3 | y | servo 0 (pan) position, between 0 and 1000 |
| 4, 5 | y | servo 1 (tilt) position, between 0 and 1000 |

**Table 4.2.3.4.1a:** Pan/tilt servo control [6-9]

|  |  |  |
| --- | --- | --- |
| **Bytes** | 16-bit words | Description |
| 0, 1 | y | camera brightness sync (0xfe00) |
| 2, 3 | n | brightness value |

**Table 4.2.3.4.1b:** Camera brightness (exposure) control [6-9]

|  |  |  |
| --- | --- | --- |
| **Bytes** | 16-bit words | Description |
| 0, 1 | y | LED sync (0xfd00) |
| 2 | n | Red value |
| 3 | n | Green value |
| 4 | n | Blue value |

**Table 4.2.3.4.1c:** LED control [6-9]

* + - 1. Powering Pixy

The pixy will be powered through I/O (see, pin 2 of figure b) connector. An unregulated power input will be used since we will be pan/tilt unit with Pixy and an Arduino. **Pixy is sourcing the power** and powering the Arduino through the Arduino cable, because Pixy can source up to 1.5A of current, which is plenty for itself, the servos and the Arduino. If you attempt to have the **Arduino source the power** and power Pixy and the pan/tilt through the Arduino cable, either the Arduino's power regulator will be overwhelmed, or you'll lose a lot of power through the cable, both of which will mean **the servos won't function** (they'll probably move to one end of their limits and buzz). [6-9]

Below figures c and d shows the servo and unregulated power-in port

|  |  |
| --- | --- |
| **Figure c:** Unregulated power-in port [6-9] | **Figure d:** Servo port [6-9] |

* + 1. Line Scan Camera (128x1 sensor array)

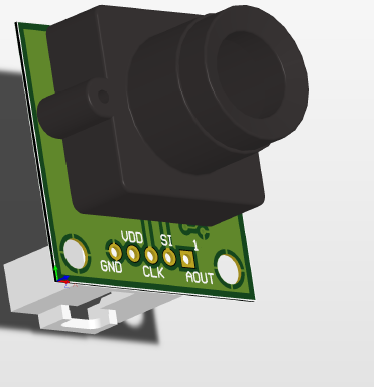
The line following camera use for this car is a daughterboard that provides a TAOS TSL1401R 128-pixel linear array sensor and an adjustable lens. Below table represents the electrical characteristics and specification.

|  |  |
| --- | --- |
| Technical Values |  |
| Supply voltage | 3V to 5.5V |
| Max. supply current | 4.5 mA |
| High level input voltage | 2V to VDD |
| Low level input voltage | 0V to 0.8V |
| Max. high level input current | 1 |
| Max. low level input current | 1 |
| Analog output volate (white) | 1.6V to 2.4V |
| Analog output volate (dark) | 0V to 0.2V |
| Lens | 7.9mm focal length, f2.4 fixed aperture, manual focus, 12mm x 0.5mm thread |
| Exposure Time | 267µS to 68mS |
| Resolution | 128 pixels |

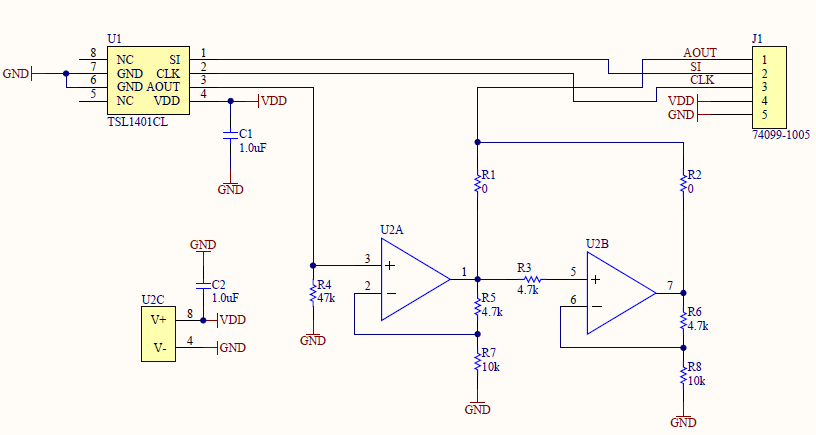
**Table 4.2.4.1:** Line scan camera’s electrical characteristics

Other specifications are following,

1. Focusable imaging lens, 5-pin (ground, power, SI, CLK, AO) physical interface on PCB on .100" grid (see below figures a and figure b)
2. Three-pin MCU interface with analog pixel output
3. Built-in amplifier stage to improve white/black differentiation



**Figure a:** Line camera’s pin diagram [12]



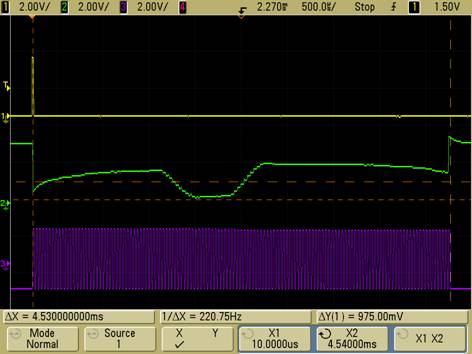
**Figure b:** Line camera’s schematic [12]

* + - 1. Signals

The following signals are produced and processed for the camera

* **A clock (CK) which** latches serial input (SI) and clocks pixels out (low to high) continuous signal
* **SI (serial input to sensor)** begins a scan / exposure discrete pulses, pulse must go low before rising edge of next clock pulse
* **Analog output** (AO) which is analog pixel input from the sensor (0-Vdd) or or tri-stated

The clock and serial input signals are simple ON/OFF signals which can be produce using a GPIO Pin, setting the pin high and low corresponding to the desired exposure time of the camera. Below figure c shows an actual camera output image. In the below figure, the yellow line is serial input (SI), green line is Camera Signal, and purple is the clock.

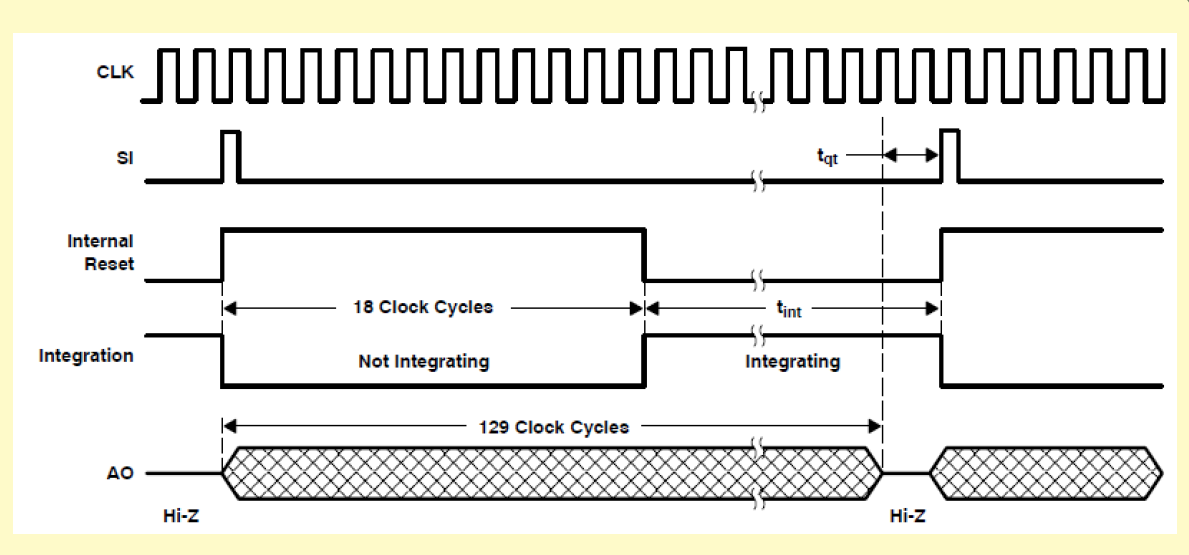
[](https://community.freescale.com/servlet/JiveServlet/showImage/102-1030-14-3992/camera+sample.jpg)

**Figure c:** Actual camera output image [12]

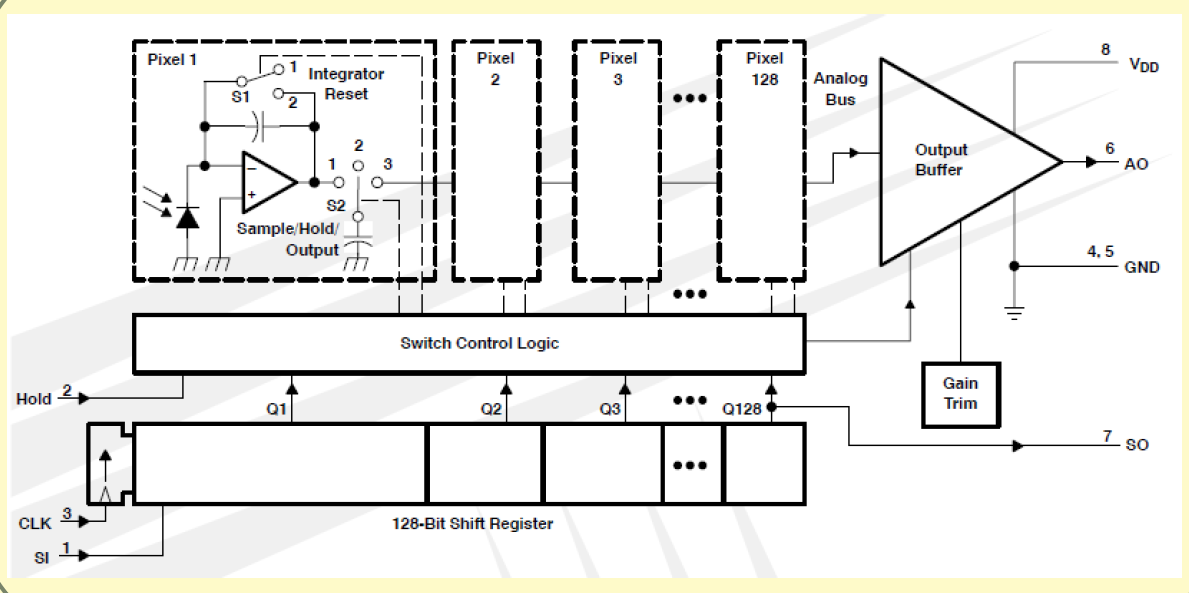
* + - 1. Timing

At the falling edge of the clock there is an SI pulse, an internal reset is high and camera is not integrating. The SI pulse goes low before rising edge of next clock pulse. During this 18 clock cycle there is no integrating which means there is no sampling. From the 19th clock cycle until the next SI pulse there is integration which means sampling. During these 129 clock cycles there is an analog output (AO) data is accumulating. After the 129th clock cycle the data is send and before the next SI pulse the data will be sent to microcontroller in order to determine the location of the line. This cycle continuously repeats during the course of car traveling on the track.

As below figure d shows this timing for creation and read of signals.



**Figure d:** Timing diagram of the line scan camera [12]



**Figure e:** Functional block diagram of the line scan camera [12]

* + - 1. Analog Read and Read/Write

The Analog Output (AO) signal from the camera need to be processed and read by the microcontroller’s Analog to Digital Converter (ADC). The microcontroller’s ADC will convert a continuous signal into a discrete number which is proportional to the signal voltage. This sampled analog signal is between 0 to 5 volts, where a 0 volts is 0 in digital discrete and 5 volts is 255 in digital discrete since an 8 bit ADC has 256 () discrete levels. The maximum signal simple rate is limited by the microcontroller which is 2 MHz. In write mode, the GPIO pin can be set, cleared, or toggled via software initiated register settings.

* + - 1. Camera Limitations

According to the line camera’s datasheet, the sensor consists of 128 photodiodes arranged in a linear array. Light energy impinging on a photodiode generates photocurrent, which is integrated by the active integration circuitry associated with that pixel. During the integration period, a sampling capacitor connects to the output of the integrator through an analog switch. The amount of charge accumulated at each pixel is directly proportional to the light intensity and the integration time. The integration time, T can be calculated using below formula, where n is the number of pixels.

The minimum integration time is 33.75us and the maximum integration time is capacitors will saturate if exceeding 100ms. The frequency range between 5 KHz to 8 MHz, where 8 MHz is in the equation above.

The integration time occurs between the 19th clock cycle and the next SI pulse. The clock frequency itself has little to do with the integration time. One each rising edge, the clock outputs one of the previously sampled intensity values. This means that integration time should be set by varying the time between SI pulses, not changing the clock frequency. This makes the CLK frequency high, and has as much time as needed between the two SI pulses to obtain the desired intensity value. [15]

* + 1. Servo

The **Hitec HS-422 Servo Motor** is one of the most durable and reliable servos Hitec. Its dual iron-oilite bushings, high impact resin gear train, high performance circuitry, excellent centering, and resolution. A figure of servo motor is shown below in figure a. Below, table 1 shows the physical and technical description of servo motor.

|  |  |
| --- | --- |
| Technical Values |  |
| Control system | +Pulse width control 1500 NEUTRAL |
| Operating voltage range | 4.8V to 6.0V |
| Operating temperature range | -20 to +60C |
| Test voltage | At 4.8V At 6.0V |
| Operating speed | 0.21sec/60 at NO Load 0.16sec/60 at NO Load |
| Stall Torque | 3.3kg.cm(45.82oz.in) 4.1kg.cm(56.93oz.in) |
| Operating angle | 45/ one side pulse traveling 400 |
| Direction | Clock wise/pulse traveling 1500 to 1900 |
| Current drain | 8mA/IDLE and 150mA/No load running |
| Dead band width | 8 |
| Connector wire length | 300mm (11.81 in.) |
| Dimensions | 40.6x19.8x36.6mm (1.59x0.77x1.44in) |
| Weight | 45.5g (1.6oz) |

**Table 1: Hitec HS-422 Servo Motor** spec. [11]

|  |  |
| --- | --- |
| **Figure a:** HS-422 Servo Motor [11] | **Figure b:** Lynxmotion Pan and Tilt Kit / Aluminium [10] |

Camera mount w/ two servo based pan-tilt mechanisms assembly for horizontal surface mount and it uses two Lynxmotion servo brackets. It includes two HS-422 servo motor. Above figure b shows pan-tilt kit.

* + 1. Hall Effect Sensor (Speed Encoder)

**The DFRobot Wheel Encoders for DFRobot 3PA and 4WD Rovers are used to get the rotation degree of the wheel. Its best fit with micro DC geared motor.** Below table 5.1.6 shows the technical specifications of speed encoder.

|  |  |
| --- | --- |
| Technical Values |  |
| Voltage | +5V |
| Current | <20mA |
| Resolution | 20PPR |
| Weight | 20g |

**Table 4.2.6.1:** **Speed encoder** spec. [15]

* + 1. Battery

The lithium (ion) polymer (LiPo) battery is the power source for the car. It has a high-capacity and high-discharge rate which will make it feasible to add optional feature without over draining the current. The below figure is representation of LiPo battery.



**Figure 4.2.7.1:** LiPo battery [5]

The table below (4.2.7.1) shows the specifications of LiPo battery

|  |  |
| --- | --- |
| Technical Values |  |
| C Rate | 20C |
| Volts | 7.4V |
| Capacity | 2000mAh |
| Cell Count | 2S |
| Cell Configuration | 2S1P |
| Continuous Discharge | 20C (40A) |
| Max Burst Rate | 30C (60A) |
| Max Volts per Cell | 4.2V |
| Max Volts per Pack | 8.4V |
| Min Volts per Pack | 6V |
| Charge Rate | 1C (40A) |
| Wire Gauge | 14 AWG Soft and Flexible Low Resistance Silicone Wire. |
| Plug Type | Venom UNI Plug. Compatible with Traxxas Plug, Tamiya Plug, Deans Plug & EC3 Plug. |

**Table 4.2.7.1: LiPo battery** spec. [5]

* + 1. Full-Bridge PWM Motor Driver

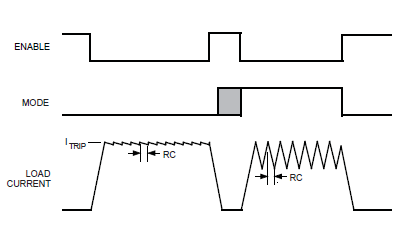
It’s designed for bidirectional pulse width modulated (PWM), current control of inductive load. The feature of this is shown below in the table and a pin diagram is shown below in the figure.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Technical Values |  | | continuous output currents | A | | Operating voltage | 50V | | Logic supply voltage | 3 to 5 5V | | Maximum PWM frequency | 40 KHZ | | Max. leakage current | 50 | | Motor supply current | 700 | | Reference input voltage range | 0 to 1 V | | Internal thermal shutdown circuitry |  | | Crossover current and UVLO protection |  |   **Table 4.2.8.1a:** H-bridge spec. [16] | **Figure 4.2.8.1b:** H-Bridge pin diagram [16] |

Below formula is used to calculate the peak value of the current limiting (), where is reference input voltage and is current sensing resistor.

In forward or reverse mode, when the current reaches , the comparator reset a latch that turn off the selected source driver or selected sink and source driver pair depending on whether the device is operating in slow or fast current-decay mode.

In slow current-decay mode, the selected source driver is disabled and both sinks are turned on. In fast current-decay mode, the selected sink and source driver pair are disabled, then the opposite pair is turned on. Below figure 4.2.8.2 shows this fast and slow current decay waveforms.



**Figure 4.2.8.2:** Current decay waveform [13]

During braking, need to ensure that the motor’s current does not exceed the rating of the device. Below formula is used to calcite this, where is back EMF voltage, and resistive load.

RC fixed off-time is an internal PWM current-control circuitry which uses a one shot to control the time of driver(s) remain(s) off. Below formula is used to calculate one-short time, fixed off-time (), where is external resistor, and is capacitor.

This is a generic description of the motor, for a complete datasheet see section 1.5.1 reference 13.

* 1. Optional Components
     1. LCD

The LCD used on the car is called SainSmart I2C IIC Interface RGB LED Screen LCD 1602 + Keypad For Raspberry Pi. The feature of this LCD is following:

* LCD1602 LCD monitors
* RGB LED
* Five key buttons (Up / Down / Left / Right / OK to select)
* Displays two lines of 16 letters,
* Five key buttons with LCD1602, provides input interface, which can reach the selected menu function.
* RGB LED with different input level, can display seven kinds of colors

The below figure 4.3.1.1 shows this LCD.



**Figure 4.3.1.1:** LCD [14]

* + 1. Bluetooth

The bluetooth used is called BlueSMiRF by Sparkfun. It uses the RN-442 module, and works as a serial (RX/TX) pipe. It’s a small, low power, and highly economic bluetooth radio. A figure is shown below 4.3.2.1. The feature and electrical characteristics of this is shown in below table 4.3.2.1. On the receiver end of the PC, bluetooth class 1 (bm08c1 c08 v1) used to get the data from the BlueSMiRF.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Technical Values |  | | Radio | 0.15x0.6x1.9 in | | Low power | Sleep: 26  Connected: 3  Transmit: 30 | | Transmission distance | 18m | | Frequency | 2.402~2.480 GHz | | Operating Voltage | 3.3V-6V | | Serial communications | 2400-115200bps | | Operating Temperature | -40 ~ +70C | | Supply Voltage (DC) | 3.3V | | Built-in antenna |  |   **Table 4.3.2.1:** BlueSMiRF spec. [18] | **Figure 4.3.2.1:** BlueSMiRF [18] |

* + 1. LED

The LED used on the car is called Diffused Green 10mm LED, figure is shown below 4.3.3.1. The feature and electrical characteristics of this LED is shown in below table 4.3.3.1.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Technical Values |  | | Diameter | 10mm | | Wavelength | 565nm | | Forward Voltage | 2.2-2.5V @ 20mA current | | Reverse Voltage | 5V | | Forward Current | 20mA | | Max. reverse current | 10 | | Peak Forward current | 120mA | | Operation temperature | -35 ~ 80 |   **Table 4.3.3.1: LED** spec. [16] | **Figure 4.3.3.1: LED** [16] |

* + 1. Buzzer

The horn used on the car is called Piezo Buzzer – PS1240, figure is shown below 4.3.4.1. It makes beeps, tones and alerts. The technical details of the horn shown in below table 4.3.4.1

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Technical Values |  | | Diameter | 11.90mm | | Height | 6.53mm | | Weight | 0.70g | | Peak-to-peak voltage | 3.3V | | Loudest tone frequency | 4 KHz |   **Table 4.3.4.1: Horn** spec. [17] | **Figure 4.3.4.1: Horn** [17] |