Electrical Overview

Year: 2024 Semester: Spring Team: 05 Project: Dodgebot

Creation Date: ­February 1, 2024 Last Modified: February 1, 2024

Author: Ayman Motoda Email: aelnamr@purdue.edu

Assignment Evaluation: See Rubric on Brightspace Assignment

1.0 Electrical Overview

For this project, we are using a 32-bit microcontroller to communicate with an external computer to receive motor coordinates from the position calculation performed. We then take this value and feed it to the motor controller. To communicate with the computer, we are going to use UART at the maximum settable baud rate by the setup software to allow low latency between the controller and the computer. To communicate with the motor, we are going to use CANbus since this is the only natively supported communication protocol used by the microcontroller. The maximum baud rate of the motor interface is 1Mbit/s [1], so we are going to communicate with the motor at that rate also for lower latency.

2.0 Electrical Considerations

For our power delivery system, our input voltage is going to be at 12V and can be delivered at 5A. The required voltages for the different components around our board are going to be 5V and 3.3V. We will be using buck controllers to create these voltages. The STM32F2 chip that we are using has a typical operating voltage of 3.3V [2]. The NVIDIA Jetson computer that we are using has an operating voltage of 5V [3]. Any peripherals that are connected to both of these computers will be powered off by either of these chips.

The main essence of the operation of our robot is going to be speed. Our target is to dodge a punch that would occur in 100ms. We are going to allocate around 50ms to the computation and signal communication time. This means that having a higher clock speed will help in our baud rate in our communication systems. The maximum baud rate of UART on our STM32 chip is around 1Mbit/s [2]. This requires a clock rate of 16MHz for just the UART lines. The other communication system of CANbus will require the same baud rate and clock speed to run it. This means that at least a operating frequency of >32MHz is required for our system. For reliable operation, we will want to run at a frequency of >100MHz.

The total power that can be supplied to the board through the 12V input is 60W. The total output power that can be supplied by the switching regulators is 5V at 5A or 25W, and 3.3V at 5A or 16.5W[4]. The total power output by the regulators is 41.5W. Since we are using switching regulators that are at least 82% efficient, we only have a total power usage of 51W which is within our input supply power demands.

|  |  |  |  |
| --- | --- | --- | --- |
| Device | Max Current Draw [A] (5V) | Device | Max Current Draw [A] (3.3V) |
| NVIDIA Jetson [3] | 4 | STM32F2 [2] | 0.120 |

*Table 1: Electrical Max Current Component Requirements*

Any peripherals connected to these devices are within the power draw specifications provided as they are powered off these computers/microcontrollers, so no additional power draw should occur.

3.0 Interface Considerations

The two utilized interfaces in this project are UART and CANbus. UART is used for communicating with the main computer. The main computer has the motor commands we need to send to the motor controllers. The STM32 and the Jetson are going to be set up for UART so that we can transmit this data. We chose this methodology because the distance is going to be short between both controllers, so using a synchronous protocol was not a concern since it will require less lines as well as making it much easier to integrate with the Jetson. The baud rate is also very high, so we can get very low latency between both controllers. A 1Mbit baud rate allows transmission times of way below 1ms [2]. The second communication protocol that is being used is CANbus. This is because this is the only way we can communicate with the motor controllers. This has the same baud rate as UART, so there is no bottle neck for each other except for the speed of the microcontroller to take the data and create the new packet frame for the other communication interface [5]. The peripheral interface that we will focus on is USB, which is built into the Jetson, but it is to receive the camera feed.

4.0 Sources Cited:

[1] “CAN-Bus Interface connector X13/X14,” *Kollmorgen.com*, 2023. https://webhelp.kollmorgen.com/akd2g/english/Content/AKD2G-S%20Install/e/AKD2S\_install\_09\_21\_00\_CANbus%20interface.htm?TocPath=AKD2G%20Installation%7CElectrical%20Installation%7C\_\_\_\_\_13 (accessed Feb. 01, 2024).

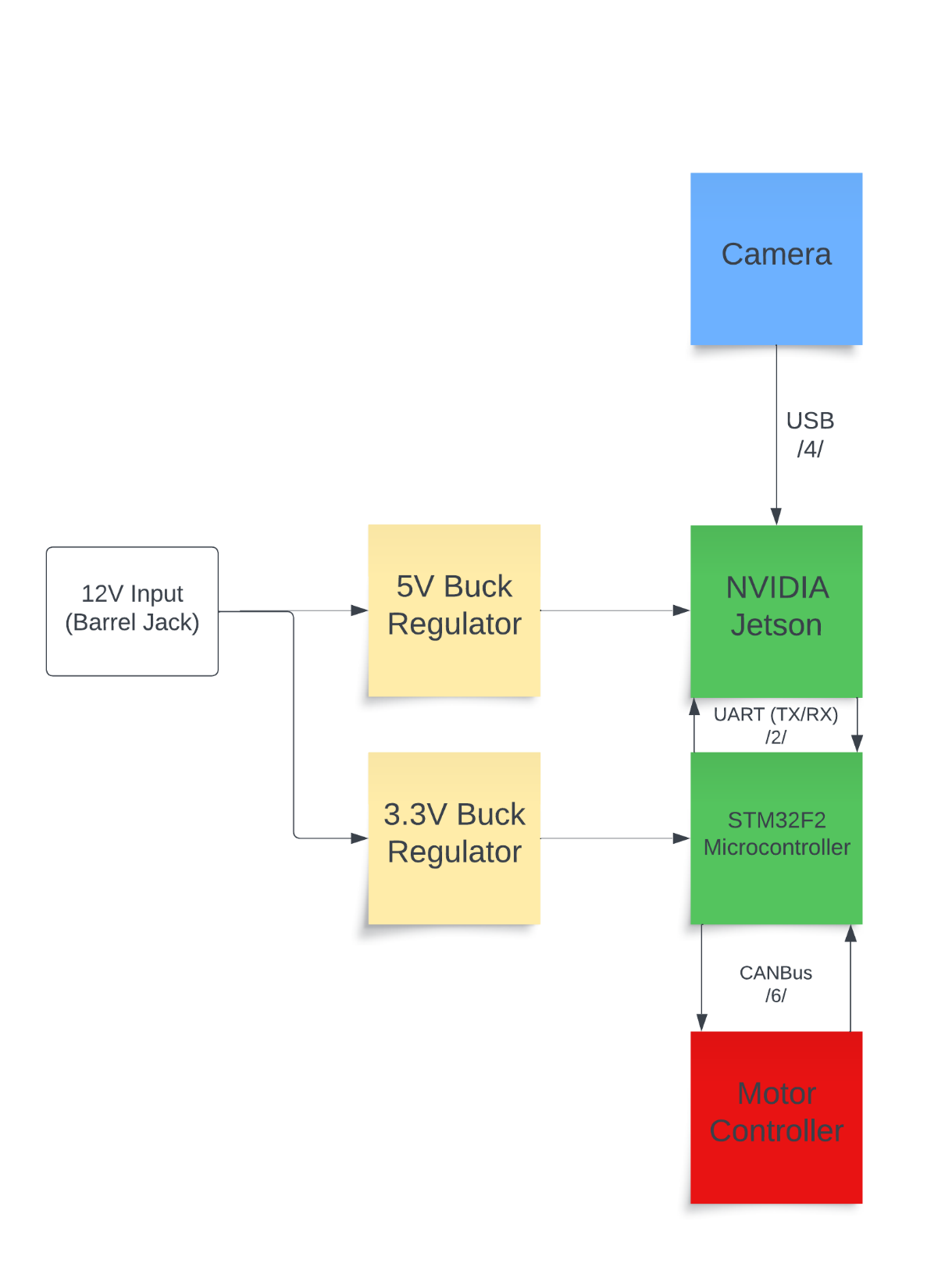
[2] STMicroelectronics, “Arm®-based 32-bit MCU, 150 DMIPs, up to 1 MB Flash/128+4KB RAM, USB OTG HS/FS, Ethernet, 17 TIMs, 3 ADCs, 15 comm. interfaces and camer,” *STMicroelectronics STM32F205xx STM32F207xx*, Jul. 2020. https://www.st.com/resource/en/datasheet/stm32f207vg.pdf

[3] NVIDIA, “DATA SHEET NVIDIA Jetson Nano System-on-Module Maxwell GPU + ARM Cortex-A57 + 4GB LPDDR4 + 16GB eMMC,” *NVIDIA Developer*, 2014. https://developer.download.nvidia.com/assets/embedded/secure/jetson/Nano/docs/JetsonNano\_DataSheet\_DS09366001v1.1.pdf?HZQ9BeogomVeemquiSOvhM7IlASVgbYrcnerDqbamhgFT5S6OkQQGpJ8msR59j7HNav0uJMUWPGre5Ww15J7JNqRyx1A39U4VKjOtE3ajsUDzBhR9sl7suRkyi5jqqD5gsoPpMFkgLJSGAWYMUiBRHfKH8gg4qnbs5LaO2P1gqPnN8wqiXcqopTTfS6w== (accessed Jan. 20, 2024).

[4] Texas Instruments, “LM2678 LM2678 SIMPLE SWITCHER ® High Efficiency 5-A Step-Down Voltage Regulator,” 2000. Accessed: Feb. 01, 2024. [Online]. Available: https://www.ti.com/lit/ds/symlink/lm2678.pdf?ts=1706759876304&ref\_url=https%253A%252F%252Fwww.google.com%252F

[5] Kollmorgen, “CAN-Bus Interface Connector X13/X14,” *Kollmorgen.com*, 2023. https://webhelp.kollmorgen.com/akd2g/english/Content/AKD2G-S%20Install/e/AKD2S\_install\_09\_21\_00\_CANbus%20interface.htm?TocPath=AKD2G%20Installation%7CElectrical%20Installation%7C\_\_\_\_\_13

Appendix 1: System Block Diagram

**