College of Engineering, Science and Environment

School of Engineering

Mechanical Engineering Project B

Semester 2 - 2024 ENGG4801 B

THE UNIVERSITY OF NEWCASTLE AUSTRALIA

FINAL YEAR PROJECT

PROJECT TITLE

NU Racing Powertrain Engineer

NAME & STUDENT NUMBER

Joshua Hayward C3356034

SUPERVISOR

Dr Dylan Cuskelly

ENGG4801B - NU Racing Powertrain Engineer

14/12/2024

Joshua Hayward [1]

[1] *Student of Mechatronics Engineering,*
The University of Newcastle, Callaghan, NSW 2308, AUSTRALIA
Student Number: C3356034
E-mail: joshua.hayward@uon.edu.au

Declaration

I declare that this thesis is my own work unless otherwise acknowledged and is in accordance with the University's academic integrity policy available from the Policy Library on the web at http://www.newcastle.edu.au/policylibrary/000608.html

I certify that this assessment item has not been submitted previously for academic credit in this or any other course.

I acknowledge that the School of Engineering may, for the purpose of assessing this thesis:

- Reproduce this thesis and provide a copy to another member of the Faculty; and/or
- Communicate a copy of this assessment item to a plagiarism checking service (which may then retain a copy of the item on its database for the purpose of future plagiarism checking).
- Submit the assessment item to other forms of plagiarism checking.
- Where appropriate, provide a copy of this thesis to other students where that student is working in/on a related project.

I further acknowledge that the School of Engineering may, for the purpose of sector wide Quality Assurance:

- Reproduce this thesis to provide a copy to a member of another engineering faculty.

Furthermore, I have checked that the electronic version is complete and intact on the submitted location.

Student name: Joshua Hayward

Signature: 

Date: 14/02/2024

I have not left a mess in the lab declaration page.

I Joshua Hayward, having completed the NU Racing Formula SAE-A Final Year Project, and using Tunra Annexe laboratory spaces, hereby submit that the area has been cleaned to level equal to, or better than when I commenced my work. All items I borrowed have been returned, and the area is now suitable for inspection and sing off by the appropriate laboratory manager.

Student name: Joshua Hayward

Signature: 

Date: 14/02/2024

Acknowledgements

I would like to thank all members of the NU Racing team who I have collaborated and bonded with over the past year. To my family, friends and relationships who have been extremely understanding, accommodating and supportive through what has been an extremely intensive year of study.

To the Mechanical Engineering department lead, Lachlan Fisher, and the Chief Engineer Josh Wenham for providing me with a beautiful race car through which I can spend countless hours developing my skills as an engineer.

To the Mechatronics Engineering department lead, Alec Chapman, for always pushing me to achieve the greatest that I can. For believing in me when it was looking like our year was over. I am gracious for the experience it has been working around you and hope I can continue to work with engineers like you.

To Tim Kerr and Marisa McLean, for orchestrating what has been such a great year of banter, motorsport and a life-long memory I will never forget.

To Dr Dylan Cuskelly, Dr Alex Gregg and Malcolm Sidney for sharing with us their knowledge and for enabling 20 students to go wild with quite literally hundreds of thousands of dollars of equipment to gather hands-on experience before entering the workforce.

I am appreciative for everybody I have met during this experience and I hope to carry the torch for NU Racing in my professional career.

Dot Point Summary

Acting as the Powertrain Engineer, a summary of contributions include:

- Commissioning of NU24 powertrain
  - Commission EMRAX 228
  - Calibration of Resolver
  - Diagnose CM200DZ
  - Commission CM200DX
  - Overhaul of system EEPROM
- Development of Regenerative Braking
  - Conduct research on implementation
  - Design regenerative braking strategy
  - Commission regenerative braking
  - Validate implementation
- Compliance of NU24
  - Fix APPS Plaus error
  - Fix inverter CAN time-out
  - Fix inverter enable reboot process
  - Fix BOTS error
  - Ensure compliance of regenerative braking
- Trackside Data Analysis
  - Configure MoTeC channels
  - Configure MoTeC display windows
  - Configure MoTeC warnings
  - Configure BMS CAN messages
  - Configure NU24.dbc
- FSAE-A Competition
  - Aid with EV Static technical inspection
  - Aid with EV Functional technical inspection
  - Strategic deployment of regenerative braking
  - BMS hard fault diagnosis
  - Ensure completion of endurance event

Abstract

This report entails the body of work completed by the author conducting their final-year-project as the Powertrain Engineer for NU Racing's 2024 Formula SAE-A campaign. The primary outcome of this project is the commissioning of a high-torque output motor and motor controller to build upon the successes of the 2023 competition EV. The upgraded powertrain resulted in a direct increase in performance for the vehicle at the 2024 competition, scoring valuable points as a result of the authors contribution.

Research, implementation and validation of a regenerative braking system was also conducted as part of the authors work with the NU Racing team. This year-long project allowed for the safe and effective addition of this feature, increasing the performance and efficiency of the competition EV. Strategic deployment ensured that the author was able to maximise the impact of this addition in scoring more points at the FSAE-A competition held in December of 2024.

Further work was completed ensuring the reliability of the EV as it was an area that the team had struggled with prior to the authors involvement. This was done through data-driven development, ensuring all modifications had an explored outcome and were carefully selected to only increase reliability. The result was a vehicle which performed consistently and delivered strong results in all dynamic events during competition.

Upon the conclusion of the final-year-project the author had achieved the following deliverables:

- Commissioning of a competitive electric drivetrain
- Commissioning of a replacement motor controller
- Integration of a functional regenerative braking system
- Development of documentation for troubleshooting powertrain components
- Development of reliable and competition-compliant mechatronics subsystems

Contents

List of Figures

List of Tables

List of Code Blocks

Glossary

NU23 - Newcastle University 2023 Competition Formula SAE Car
NU24 - Newcastle University 2024 Competition Formula SAE Car
EMI - Electromagnetic interference
CCL - Charge Current Limit
DCL - Discharge Current Limit
CAN - Controller Area Network
BMS - Battery Management System
PEN - Pedalbox Electronic Node
DEN - Dash Electronic Node
CEN - Central Electronic Node
LVD - Low Voltage Distribution Node
SOC - State of Charge
OKHS - OK High Signal
RTD - Ready to Drive
EEPROM - Electrically Erasable Programmable Read-Only Memory

1. Background

NU Racing

Newcastle University Racing is a team run by students, for students, to compete in the annual FSAE-A competition. In 2024 it comprised of 15 core-members and an additional roster of junior students and extracurricular team members. The author was involved in the Mechatronics department of the team working on their 2024 competition vehicle, NU24. Roles within the team are decided as part of the Final-Year Project induction week, where the author was assigned as the team's Powertrain Engineer. NU Racing employs an iterative development mentality, where NU24 aims to make small changes build upon the successes of Newcastle University's 2023 competition car, NU23.

Figure 1: NU Racing Final-Year Project Team at FSAE-A 2024

Formula SAE-A Competition

The FSAE-A Competition is the yearly Australasian regional Formula-Student competition and is the event at which NU Racing and many other University teams compete in Melbourne for a 4-day engineering challenge. It is an opportunity for students to become involved in a hands on, practical engineering environment and aims to create better, more experienced and employable engineers. In 2024 there were 22 Universities that competed across a range of static and dynamic events in which points were awarded based on the performance relative to all other teams.

FSAE-A 2024 teams participate in a design event, where they must justify the decisions made during the year when designing their competition vehicle. This includes preparing data analysis, validation and demonstrating overall understanding of key engineering principles. Additionally teams must cost their car as if it were to be mass manufactured and present this to the judges during the cost event. There is a business event which requires the team to conduct a business presentation where they are marked based on overall concept and delivery.

The teams must then complete several technical inspections in order to compete in various dynamic events. This is to ensure the safety and compliance of the competition vehicle to create an even basis for competition. The dynamic events include the skidpad and autocross events which are designed

to test the balance and handling capabilities of the FSAE vehicles. An acceleration event is held to measure the cars straight line acceleration over a 75 metre distance. Finally an endurance event is held which aims to evaluate the overall reliability and efficiency of the teams cars as they complete a 22 kilometre timed sprint around the autocross track.

These events are all weighted differently and their points tallied to award an overall winner of the FSAE-A competition. The primary goal of NU Racing's 2024 campaign was to maximise points scored at the competition held in December.

CAN Bus / NUCAN

NU Racing implements a nodal electronic design in which key subsystems of the car are given an electronic node with a Teensy 4.0 microprocessor to do localised calculations and broadcast and receive important information over the vehicle-wide CAN bus. It is crucial that the nodes are able communicate with one another effectively, as any singular node losing connection may cause the entire CAN bus to fail due to its error frame functionality.

NU Racing has an internally maintained CAN bus package that is able to used directly with the arduino-based processors without a deeper understanding in the public flexCAN_T4 libraries [1]. The team is able to modify the NU24.dbc file directly and when it is pushed to the online GitHub repository the respective NU24_CAN.cpp and NU24_CAN.h are automatically updated. This is a result of the work of previous team members Matthew Whitworth [2] and Jacob Bush [3].

NUCAN is the sole communication path for sending and receiving information from the Cascadia motor controller in NU24. Operating the motor controller in CAN mode allows the user to perform calculations and adjust the signals being sent without the need for complex analogue circuits. For example, introducing a scaling factor or creating a non-linear map[1] for the accelerator pedal input is able to be done onboard the PEN with a few lines of code.

PEN

The Pedalbox Electronic Node is the primary command unit for sending messages to the motor controller. It is responsible for reading the accelerator pedal position and performing internal calculations before sending a torque request over CAN where it is read by the motor controller. Throught 2024, additional functionality was added to the PEN to increase its utilisation to the point where it now acts as a small capability Vehicle Control Unit. The PEN handles certain BMS functions such as calculating and commanding a dynamic current limit to be used during torque calculations. Prior to 2024, the PEN was simply used to interpret and send the accelerator pedal position sensor data, as well as some minor analogue circuitry.

Regenerative Braking

Regenerative braking is the action of extracting kinetic energy from a moving vehicle by capturing it through the electric motor and charging the battery from the resulting current. This functionality can be varied and adjusted to not only increase system efficiency but also to increase braking performance by created a large resistive force used to slow down the vehicle. Regenerative braking works by syncing the internal switching mechanisms to allow back-emf generated by the rotating permanent magnet in the motor to result in a net current flow through the DC bus to the accumulator.

[1] Such as that in Figure 37

2. State of the Art

EV3

To understand the development of NU Racing's FSAE cars it is important to acknowledge the previous design iterations. EV3 was the 2022 competition vehicle and featured a dual-motor configuration. This utilised two EMRAX 188 motors and required the use of two motor controllers. In addition to this, a maximum tractive system voltage of 403.2V was used. EV3 had an aerodynamics package consisting of a front and rear wing, while the overall chassis design is similar to the cars which followed it. This car introduced the nodal electronic system design and was a testing platform for low-voltage systems and the new CAN bus.

EV3 had a successful campaign in 2022 when it finished fourth overall in the FSAE-A competition. This was one of the most successful teams in NU Racing's EV division and served as a great platform to begin development for NU23.

Figure 2: NU Racing 2022 FSAE Competition Car EV3

NU23

The 2023 NU Racing FSAE car was a development on the successful EV3 legacy platform. There were some major changes to the EV tractive system including an increase to maximum system voltage of 453.6V. NU23 also saw the change to a single motor, continuing to use the EMRAX 188 and Bamocar Unitek D3 motor controller. Late during the year this motor controller was replaced with a Cascadia CM200DZ due to constant errors with the existing Bamocar unit. In addition to this, over 50kg of weight was dropped due to simplification of the bodykit design and reduction of chassis elements. NU23 was a design shift which prioritised aligning bottlenecks and a simplified design methodology.

The 2023 FSAE-A competition saw NU23 excel in dynamic events, placing 2nd in both autocross and skidpad. This validated the effectiveness of performance upgrades made to NU23 through the year. Unfortunately the team was unable to complete the endurance event due to a hard fault triggered as a result of a BMS error condition. This resulted in an overall placing of seventh, but it provided the 2024 team with an excellent basis from which to design a podium placing car through small improvements.

Figure 3: NU Racing 2023 FSAE Competition Car NU23

NU24

The development of Newcastle University's FSAE 2024 car builds off of the successes of NU23, maintaining the same design philosophy. It was chosen to increase the torque delivery by purchasing an EMRAX 228 MV motor. Additionally, the wheelbase was shortened due to a new powerbox design and the overall weight of the vehicle was reduced by a further 20kg. The primary goal for NU24 was to reduce overall part count and complexity.

Figure 4: NU Racing 2024 FSAE Competition Car NU24

3. Commissioning of EMRAX 228 Medium Voltage Motor

3.1. Introduction

The EMRAX 228 is a compact axial flux permanent magnet synchronous electric motor [4]. It is similar in design to the EMRAX 188 used in previous NU Racing FSAE cars including NU23. It was chosen by engineers of the 2023 team during their design phase and as such, had been purchased to be integrated into NU24. This motor is a great replacement as it offers a 130% increase in torque and a 106% increase in peak power, while only adding 5.6kg in weight.[2] The motor is also slightly larger in diameter and has a lower maximum speed which means a new gear ratio was selected and the powerbox had to be manufactured to accommodate.

3.2. Powerbox Upgrade and Gear-Ratio Change

Figure 5 shows the old powerbox design which had the EMRAX 188 positioned in front of the rear axle between the spool and the accumulator. To accomodate for the EMRAX 228's larger diameter, Figure 6 shows the new motor positioned vertically above the spool which also allowed for shortening of the rear of the chassis. The decision to shift the motor upwards was made by Jye Hollier as part of the 2023 design phase as Chief Engineer [5].

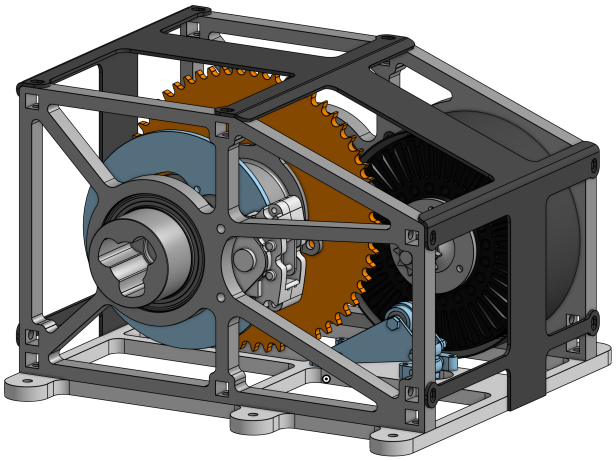


Figure 5: NU23 Powerbox

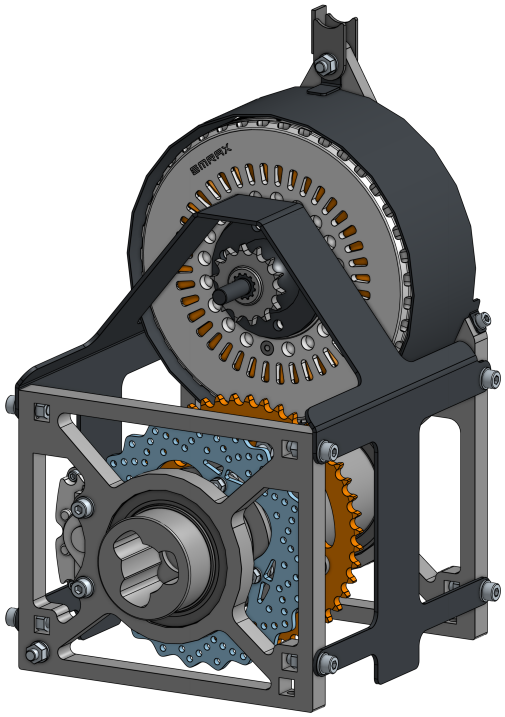


Figure 6: NU24 Powerbox

The overall drive ratio was changed from 60:11 to 46:14, where the motor will spin now at a rate 3.29 times greater than the spool. This gear change is significant as it means that with 230Nm of motor torque, the EMRAX 228 will result in a peak wheel torque of 755Nm. This is an increase of 210Nm from NU23's drivetrain.

[2]The EMRAX datasheets for the 188 and 228 motors can be found in Appendix B and Appendix C respectively

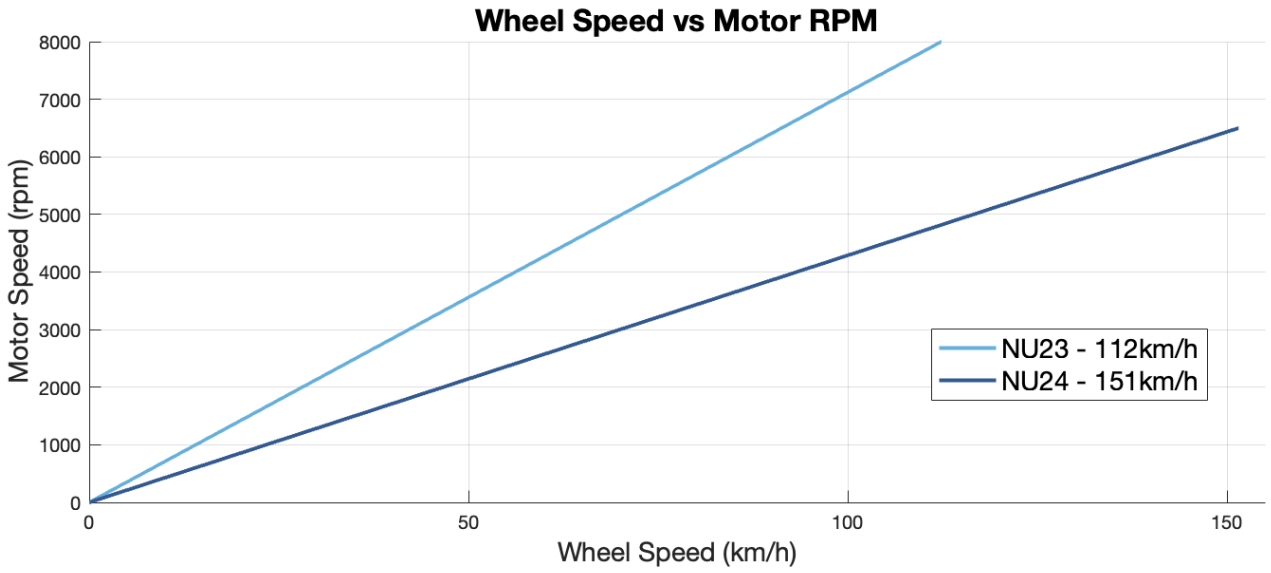


Figure 7: Wheel Speed vs Motor RPM for NU23 and NU24 drivetrain configurations

Figure 7 shows the decreased motor speed as a function of wheel speed. It can be seen that NU24 is able to achieve speeds of up to 151km/h which is important given that NU23 was regularly able to achieve its top speed.

$$\text{Torque} = \frac{\text{Current} \times \text{Voltage}}{\text{Motor Speed}} \tag{1}$$

Equation 1 shows the torque relationship for an electric motor. For a given maximum torque, maximum current draw and fixed voltage, the torque output function will resemble Equation 2.

$$\text{Torque} = \min\left(\frac{Current_{Max} \times Voltage}{\text{Motor Speed}}, Torque_{Max}\right) \tag{2}$$

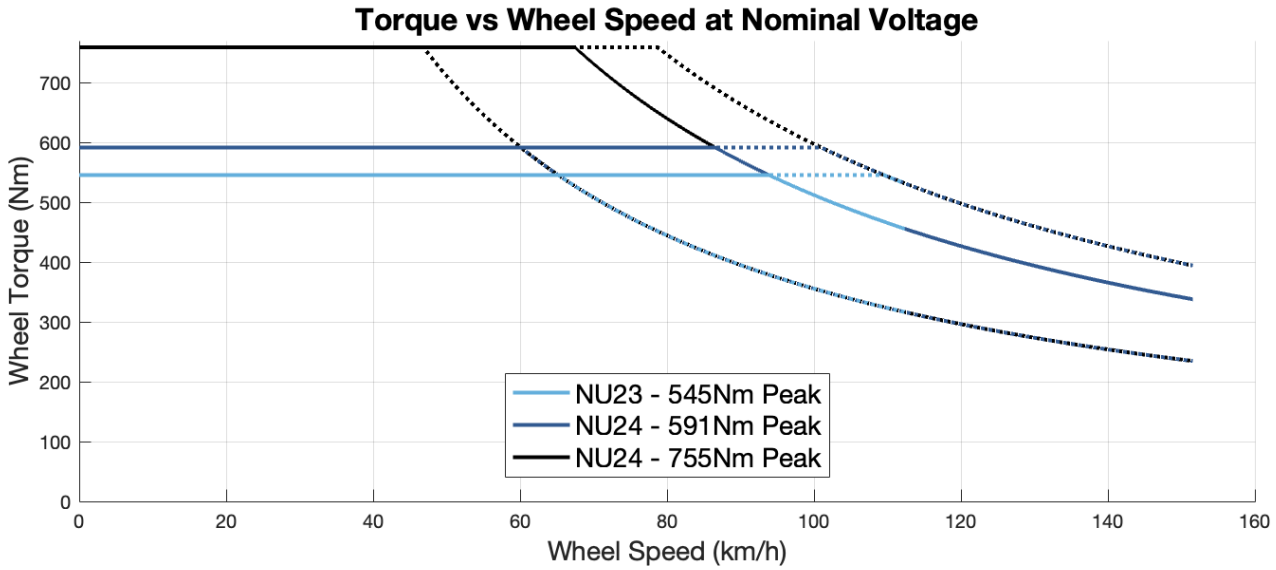
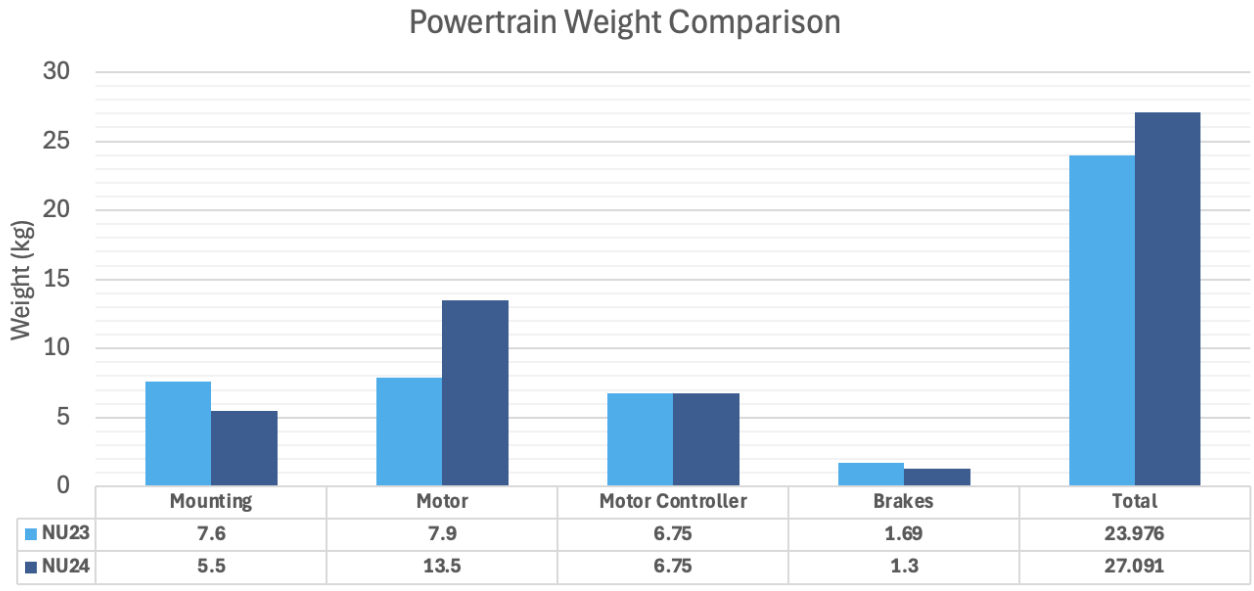


Figure 8: Wheel Torque vs Wheel Speed for NU23 and NU24 drivetrain configurations

It is important to note that the two previous figures contain both a solid and dashed line for each drivetrain configuration. This is to represent the torque at the system nominal voltage of 388.8V as well as the torque at minimum and maximum voltage as described in Appendix D. This indicates the range of expected torque values during the FSAE Competition. Additionally, there are two values for NU24 shown in the figures which represent the 231Nm of motor torque which the EMRAX 228 is capable of, and the 180Nm of motor torque that were deemed to be the physical limit of the powerbox. This was determined during testing and validation of the NU24 powerbox, and can be further read upon in Lachlan Fisher's report.[6]

Figure 8 shows the relationship between motor torque and motor speed which is determined by three key factors; maximum motor torque, gear-ratio and maxmium current. Since both cars utilise the same pack configuration, it can be seen that once the curve enters the current-limiting section (seen by the inverse proportionality to speed) all configurations follow the same trajectory. The figure accounts for the change in gearing between NU23 and NU24 which will affect both wheel torque and wheel speed, hence the current limiting will have a fixed relationship to wheel speed.

The drivetrain for NU24 is able to achieve a greater wheel torque than NU23 for all speeds less than 93km/h where it is then current-limited due to the rated output of the pack. Combining this with the greater top-speed capability, the EMRAX 228 with the new gearing is able to deliver greater performance in all aspects when compared the previously used EMRAX 188.



	Mounting	Motor	Motor Controller	Brakes	Total
NU23	7.6	7.9	6.75	1.69	23.976
NU24	5.5	13.5	6.75	1.3	27.091

Figure 9: Powertrain Weight Comparison between NU23 and NU24

This is achieved whilst maintaining a minimal increase in overall weight due to work done by Lachlan Fisher [6] and Rishi Mathuria [7]. Figure 9 shows that the larger motor weighs 5.6kg greater, but the total weight increase of the powertrain amounts to 3.1kg. In the scope of NU24's competition weight[3], this is a 1.3% increase, whilst the wheel torque was increased by 8.4%. Should the powerbox be upgraded during the development of NU25 to unlock the full potential of the EMRAX 228, the wheel torque available will increase by a further 28%.

[3]The official competition weight of NU24 was 242.5kg

3.3. EEPROM Changes

The Cascadia Motion CM200 series inverters require specific programming to be integrated with a list of known, supported motors. This is documented within the user manuals and due to the change of EMRAX motors, several EEPROM parameters must be changed to suit.

EEPROM Description	EMRAX 188 Value	EMRAX 228 Value
Motor_Type_EEPROM	198	128
Motor_Torque_Limit_EEPROM_(Nm)_x_10	1320	2310
Motor_Overspeed_EEPROM_(RPM)	7000	6500
Max_Speed_EEPROM_(RPM)	6500	6000
Break_Speed_EEPROM_(RPM)	6500	5500

Table 1: Comparison of EMRAX 188 and EMRAX 228 EEPROM Values

Table 1 shows the EEPROM's that had to be changed in order to configure the motor controller to work with the EMRAX 228 MV.[4] Cascadia automatically adjusts the maximum torque, quadrature and direct current limits based on the Motor Type parameter. 198 corresponds to the EMRAX 228 MV motor classification and was changed during commissioning. In addition to this, the motor speed related EEPROM's were modified to ensure that the larger motor did not exceed it's manufacturer recommended maximum speed specifications

3.4. PEN Code Modification

At the time when the EMRAX 228 was commissioned into NU24, the PEN code did not have any regenerative braking functionality, nor did it have any non-linear throttle mapping. The transition in software between the EMRAX 188 and 228 motors was achieved by modifying the maximum torque output.

```
#define MAX_MOTOR_TORQUE 150        // Defined by motor parameters (Nm)
```

This singular line of code is all that is needed to be changed thanks to the work of Jacob Bush [3] and Alec Chapman. [8] This can also be credited to the simplicity of CAN-based operation provided by the Cascadia CM200 range of inverters.

3.5. Resolver Calibration

Part of the installation of new motors is the resolver calibration process. The resolver is a rotary position device which the inverter uses to determine the rotation of the motor. Improper calibration will result in a motor that runs out of phase and in most scenarios will not rotate in an effective manner. The accuracy of the resolver calibration will dictate the efficiency of the motor and inverter, hence it is very important to do this correctly.

The correct resolver for the EMRAX 228 MV as described by Cascadia Motion is the Tamagawa TS2620N1095E161 5-Pole resolver. This has two components, one of which is installed on the rotor shaft of the motor as seen in Figure 10. The rotating component must be installed onto the rotor

[4] A full list of EEPROM changes can be found in Appendix E

shaft first. It should be noted that this does not have to go on in a certain axial orientation, but once installed it is crucial it is not able to rotate in relation to the motor shaft. This is important to note as this rotation had occurred on the EMRAX 188 during testing for NU23 in previous years. As a preventative measure, it was decided to install this component with a small amount of bearing retaining compound to limit this possibility. This has been referred to Alec Chapman, Lead Engineer for NU25, as an improper attempt at removal could lead to damage to either the resolver or motor.

The stator component of the resolver must be mounted on the provided mounting plate by using four grub-screws to seat it around the rotor. This can be clocked in any way, so long as a new calibration is performed when it has moved relative to the motor plate. It rests on a machined shoulder, and when seated correctly should align vertically with the other installed component of the resolver.

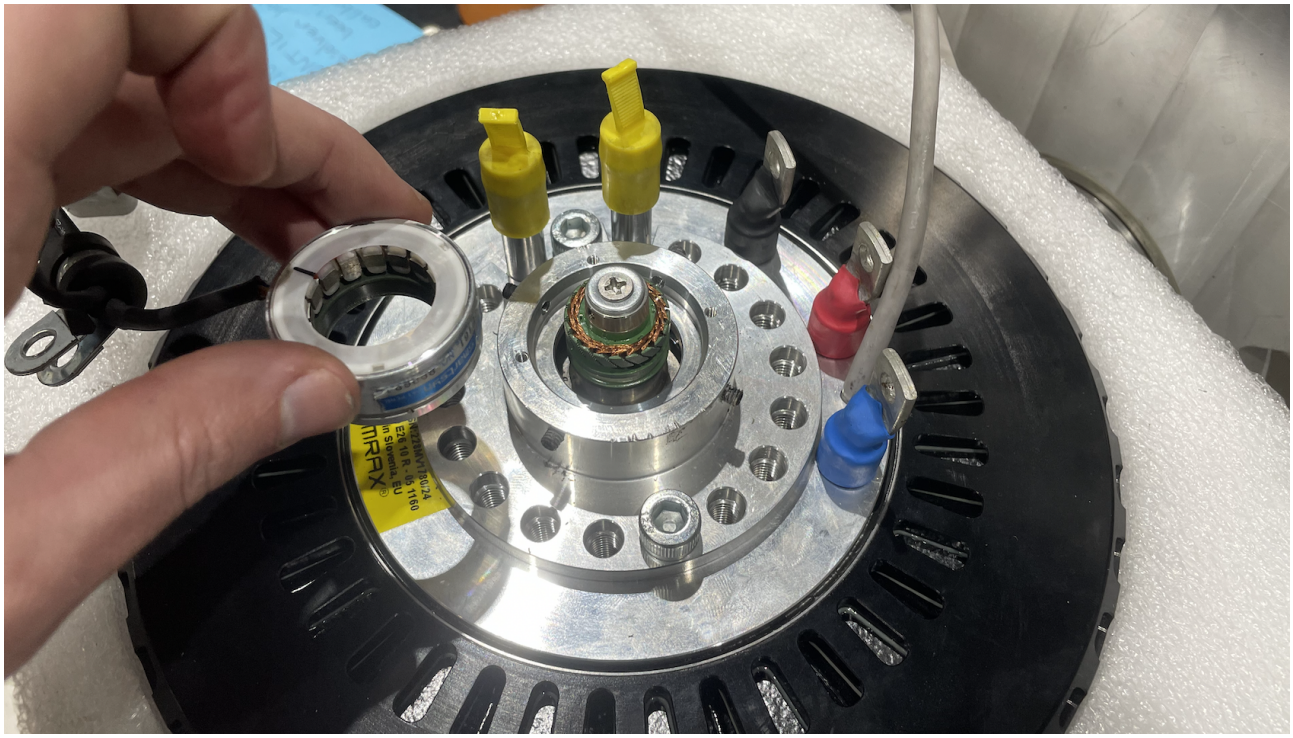


Figure 10: Tamagawa Resolver Installed on the EMRAX 228 MV

Once installed, the resolver calibration process must be undertaken as described in the Cascadia Motion Resolver Calibration Process document. [9] For future reference, the process is outlined below.

It should be noted that conducting the resolver calibration requires spinning the motor with a drill which will generate high voltages at both the phase terminals of the motor and inverter, as well as the DC terminals of the inverter. Proper safety precautions must be taken to reduce the risk of electrocution, the operator must:

- Ensure phase leads are connected securely between the inverter and motor, checking for damage in the insulation or connectors. If there is any damage to the phase leads, do not proceed with calibration and contact the NU Racing Electrical Safety Officer.
- Ensure DC high-voltage leads are connected securely to the inverter. The other end of these leads must be connected to the HIP, or high-voltage distribution board, which must be connected to the accumulator. The accumulator must have low-voltage connections which will enable the

user to complete the pre-charge and discharge sequences. If there is any damage to any DC high-voltage connections, or the pre-charge/discharge is not functional, do not proceed with calibration and contact the NU Racing Electrical Safety Officer.

- Ensure that all components, if not mounted to the chassis, are appropriately grounded to each other including the inverter, motor, accumulator and HIP. This is important for the motor to allow appropriate discharge of static voltages induced during the calibration process.
- Ensure there is no risk of wires becoming tangled or strained by clamping the powerbox down to a secure surface. Create an exclusion zone around the calibration area to ensure other people are not able to come into contact with high-voltage components.

Only after ensuring the safety of all high-voltage connections can the user proceed with the resolver calibration process.

Figure 11: Resolver Calibration setup with NU23 Powerbox

The resolver calibration apparatus that was used during testing is shown in Figure 11, where a Makita 18V drill was used as the external source to spin the EMRAX motor. Additionally, a tachometer was used to verify the motor speed as reported by the RMS GUI software.

The software used for resolver calibration can be found at https://www.cascadiamotion.com/documents under Tools, and RMS GUI.

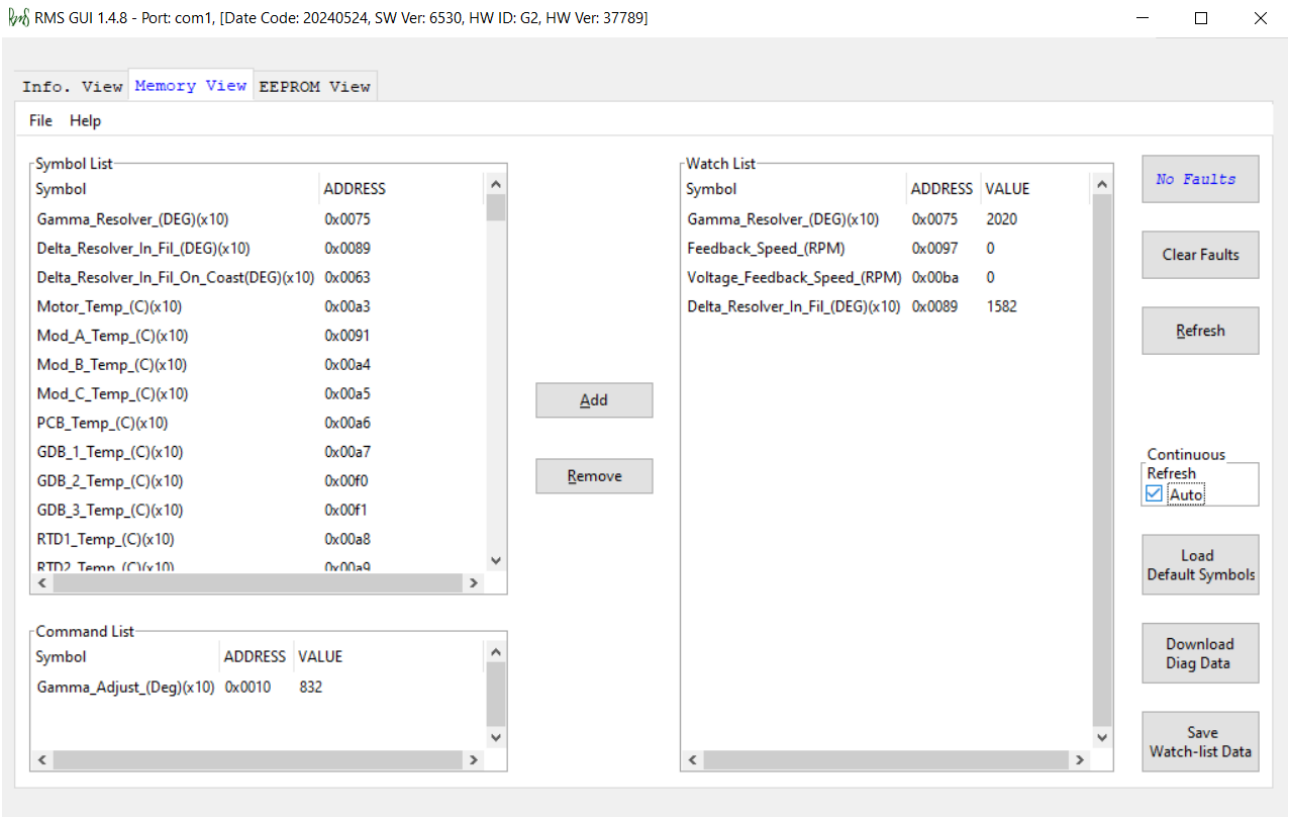


Figure 12: Example View of the RMS GUI Display Window.

Several settings must be viewed using RMS GUI during calibration of the resolver. Figure 12 shows an example window containing variables which are important during the calibration process. These variables are accessed by clicking on the Symbol List on the left of the window and hitting the 'space' key on the computer. This will open a prompt window where the user can type the name of the variable they wish to monitor. The variable can then be added to the Watch List by pressing the Add button in the center of the GUI window. When adding the Gamma_Adjust symbol, it will appear in the Command List on the bottom left of the window where its value can be modified by double clicking on the value and typing.

Figure 13 and Figure 14 show the RMS GUI EEPROM window view and CM200 bench harness. These figures will be useful to the reader should they require performing the resolver calibration process. The following resolver calibration guide has been adapted from Cascadia Motion to suit NU Racing's application.

Info. View | Memory View | EEPROM View

EEPROM List

Symbol	ADDRESS	VALUE
Serial_Number_EEPROM	0x0113	2858
PWM_Nominal_Freq_EEPROM_(kHz)	0x0150	12
PWM_Minimum_Variable_Freq_EEPROM_(kHz)	0x01be	12
PWM_Maximum_Variable_Freq_EEPROM_(kHz)	0x01bf	17
PWM_Stall_Freq_EEPROM_(kHz)	0x01bd	12
PWM_Var_Freq_Mode_EEPROM(0=NOM_1=VARS_2=VAR)	0x0179	0
PWM_Var_Dwell_Rate_EEPROM_(3ms)	0x01c0	10
Motor_Type_EEPROM	0x0119	128
Veh_Flux_EEPROM_(Wb)_x_1000	0x0100	34
Angle_Advance_Factor_EEPROM_x_100	0x0149	0
Gamma_Adjust_EEPROM_(Deg)_x_10	0x011a	-1218
Run_Mode_EEPROM(Trq=0_Spd=1)	0x0116	0
Inv_Cmd_Mode_EEPROM(CAN=0_VSM=1)	0x011b	0
Key_Switch_Mode_EEPROM	0x012b	0
Precharge_Bypassed_EEPROM_(0=N_1=Y)	0x0115	1
Relay_Output_State_EEPROM_(0=OFF_1=ON)	0x012c	0x000c
Discharge_Enable_EEPROM	0x016d	0
CAN_ID_Offset_EEPROM	0x011d	0x00a0
CAN_Extended_Msg_ID_EEPROM(0=N_1=Y)	0x0131	0
CAN_J1939_Option_Active_EEPROM	0x0132	0
CAN_OBD2_Enable_EEPROM	0x0152	0

Refresh

Load EEPROM Values

Program EEPROM Values

Save to File

Figure 13: RMS GUI EEPROM Window View.

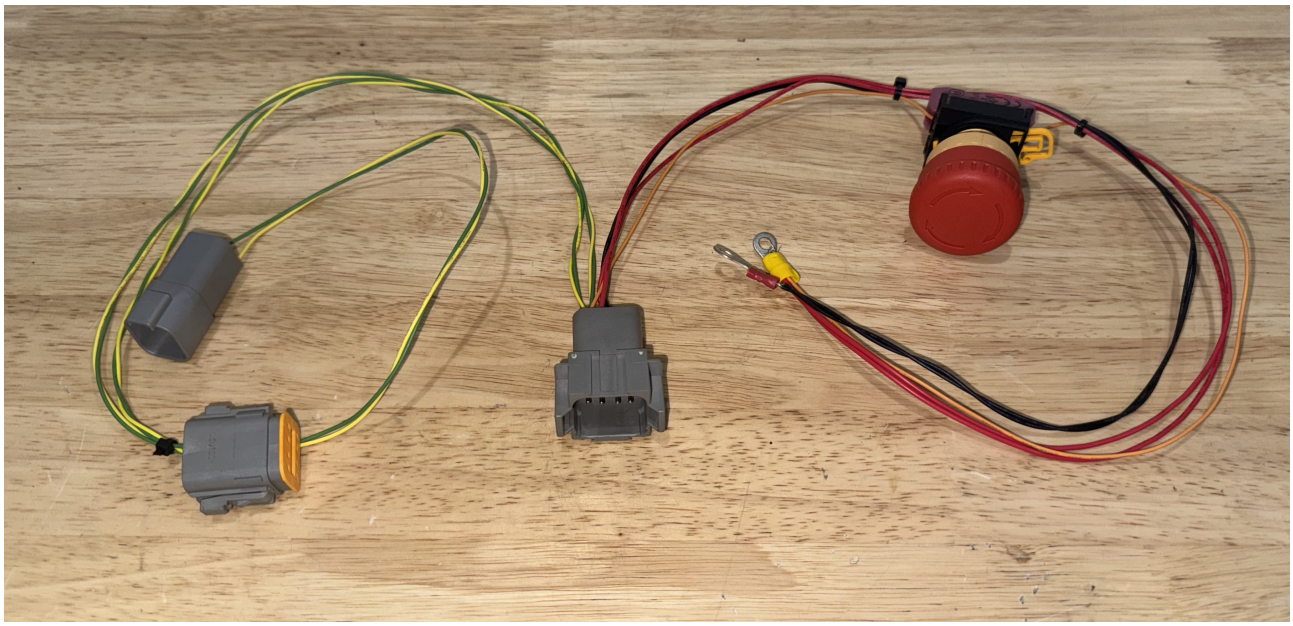


Figure 14: Bench Testing Harness made for testing with the CM200 Inverters.

Calibration Setup

1. Disconnect the motor controller wiring loom from the CEN and connect it to the bench testing harness shown in Figure 14 [INSERT IMAGE ON PREVIOUS PAGE].
2. Connect the positive ring terminal, indicated by red and orange wires, to a DC power supply positive connection.
3. Connect the negative ring terminal, indicated by black wires, to the DC power supply ground.
4. Turn the low-voltage master switch to the on position.
5. Turn the tractive-system master switch to the on position.
6. Observe the MoTeC display and ensure the precharge sequence has completed successfully. This will be indicated by the HV Present displaying a '1' on the display.
7. Ensure the water pumps are running and that there is sufficient water flow through the inverter.
8. Set the DC power supply to 12 volts, and turn it on.
9. Confirm that settings described in Section 3.3 have been set correctly for the appropriate motor and resolver combination.

Resolver Verification and Motor Direction

10. Configure the RMS GUI window to display the following parameters and enable continuous refresh mode.
    - Gamma_Resolver_(DEG)_x_10
    - Feedback_Speed_(RPM)
    - Voltage_Feedback_Speed_(RPM)
11. Rotate the motor slowly, at less than one revolution per second, in the direction that would correspond to forward motion of the vehicle and observe the values displayed by Gamma_Resolver_(DEG)_x_10. Take note of the values increasing or decreasing as this will indicate the convention of rotation for the motor.
12. Tighten an M12 nyloc nut onto the output shaft of the EMRAX motor.
13. Turn the drill to its high-speed drill function and attach the 1/2 inch driver bit with the 19mm socket and socket extension.
14. Spin the motor in a clockwise direction using the drill, observing the magnitude and polarity of the Feedback_Speed_(RPM) and Voltage_Feedback_Speed_(RPM) values. These must have equal magnitude and matching polarity, which must also match the convention described in Step 11.
15. Stop spinning the motor.

Resolver Angle Offset Adjustment

16. Configure the RMS GUI window to display the following parameters and enable continuous refresh mode.
    - Gamma_Adjust_EEPROM_(Deg)_x_10

- Gamma_Adjust_(Deg)_x_10
- Delta_Resolver_In_Fil_(DEG)_x_10
- Feedback_Speed_(RPM)

17. Spin the motor in a clockwise direction using the drill, observing the Feedback_Speed_(RPM) and Delta_Resolver_In_Fil_(DEG)_x_10.
18. Record the value corresponding to Delta_Resolver_In_Fil_(DEG)_x_10 at a time when the magnitude of the Feedback_Speed_(RPM) is greater than 1000rpm.
19. Stop spinning the motor.
20. Perform the resolver gamma adjustment calculation:
    - **Positive Feedback_Speed_(RPM)** -

$$\theta_{Adjustment} = 900 - \Delta_{Resolver} \tag{3.1}$$

$$\Gamma_{Adjust} = \Gamma_{EEPROM} - \theta_{Adjustment} \tag{3.2}$$

    - **Negative Feedback_Speed_(RPM)** -

$$\theta_{Adjustment} = 900 + \Delta_{Resolver} \tag{4.1}$$

$$\Gamma_{Adjust} = \Gamma_{EEPROM} + \theta_{Adjustment} \tag{4.2}$$

21. Enter the Γ_{Adjust} value into Gamma_Adjust_(Deg)_x_10.
22. Repeat Steps 17–21 until the Delta_Resolver_In_Fil_(DEG)_x_10 value reads 900 for positive rotation or -900 for negative rotation.
23. Set the value for Gamma_Adjust_EEPROM_(Deg)_x_10 to the final value of Gamma_Adjust_(Deg)_x_10.
24. Power cycle inverter power by turning off the DC power supply and turning it back on.
25. Repeat Steps 17-19 to ensure correct calibration by confirming Delta_Resolver_In_Fil_(DEG)_x_10 aligns to Step 21.

Safe Pack Up

26. Turn off the DC power supply
27. Turn the tractive-system master switch to the off position.
28. Observe the MoTeC display and ensure the discharge sequence has completed successfully. This will be indicated by the HV Present displaying a '0' on the display.
29. Turn the low-voltage master switch to the off position.
30. Disconnect the bench testing harness from the motor controller and DC power supply.
31. Connect the motor controller harness back to the CEN.

3.6. Results

After resolver calibration the car was returned to its race-ready condition and the motor was spun with the vehicle on stands to confirm the correct operation of the EMRAX 228 MV. The motor spun smoothly and in the correct direction, which is an indication of correct resolver calibration. After this, the team organised a track day on University campus at the ICT carpark to slowly increase the torque output of the motor to ensure the new powerbox would be able to tolerate the increased stress. The day of testing had been structured around carefully validating the safety systems of the car while increasing the maximum torque available. The following outlines the sequence of testing procedures:

System	**Testing Method**	**Result**
1. Shutdown Circuit	Set the maximum torque to 20Nm and verify the functionality of BSPD, BOTS, APPS Plausibility, and shutdown circuit E-stops.	BSPD initially not functioning due to current sensor, circuit repaired, and all other systems working correctly.
2. Motor Speed Limit	Increase maximum torque to 90Nm and set the motor speed limit to 1400rpm.	Torque noticeably increased and car stops accelerating when it reaches the programmed speed limit.
3. Field Weakening	Increase D-axis current limit, ID_Limit_EEPROM_(Amps)_x_10, to 221 Amps. Increase maximum torque to 160Nm and motor speed limit to 6500rpm.	Torque greatly increased and motor noticeably enters field weakening.
4. BMS Current Limit	Set BMS discharge current limit to 75 Amps.	Current is limited by the inverter to 75 Amps with a small amount of overshoot.[5]

Table 2: Methodology for Commissioning of NU24 with EMRAX 228 MV

When the torque was increased to a maximum of 160Nm it was hypothesised that the powerbox may not be able to handle the full torque output of the motor and it was decided not continue increasing the torque output.

Based on driver feedback from Alec Chapman, the power delivery felt no different to NU23, with the high-speed being a bit slower due to the current limiting. This was considered a major success for the initial shakedown testing day as it allowed the team to proceed with development and testing for other low-voltage systems that had to be made rules compliant.

[5]This is visible in the DC bus current shown in Figure 15.

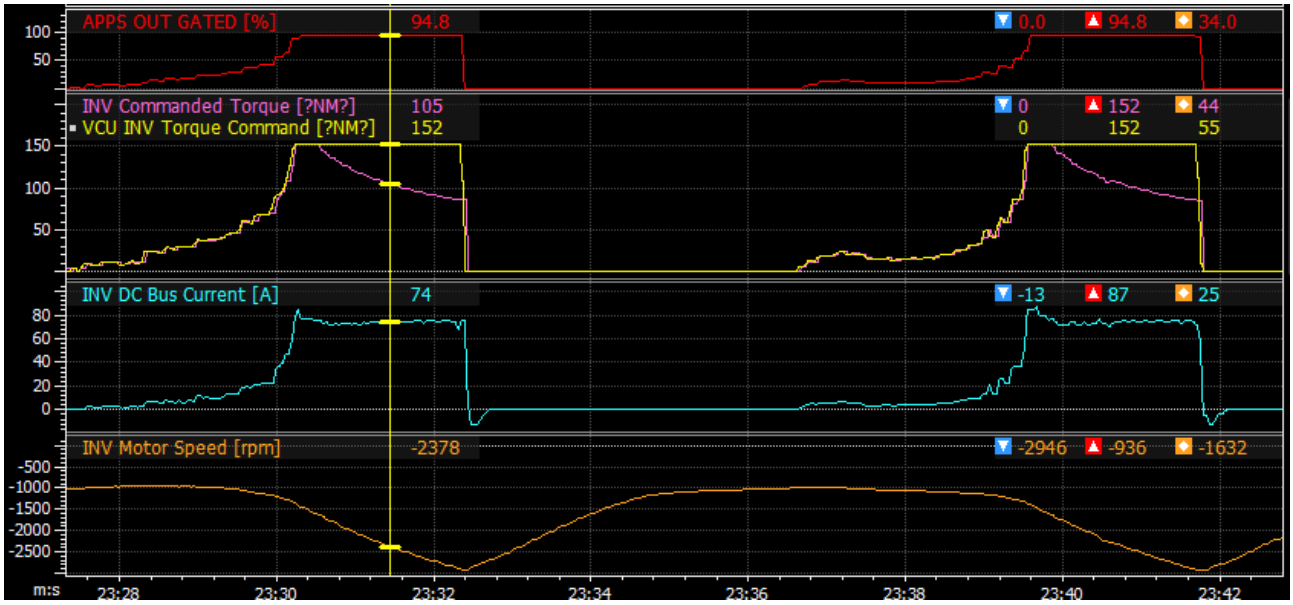


Figure 15: NU24 First Data using EMRAX 228 MV Motor.

During this testing, static BMS current limits were implemented on the Orion BMS for the first time. Figure 15 shows the DC current drawn by the motor and inverter is limiting to 75 amps. This is accomplished by the inverter saturating the CAN-based torque command, resulting in the reduced inverter commanded torque. The VCU INV Torque Commadn (Yellow line) shows the requested torque from the PEN whilst the INV Commanded Torque (Pink line) shows actual torque allowed due to the current-limiting PID controller. This follows the theoretical torque mapping observed in Figure 8, as the motor speed increases, the maximum torque available to satisfy the current limit begins to reduce with inverse proportionality.

4. Malfunction of CM200DZ Inverter

4.1. Introduction

After the commissioning of the NU24 powertrain, the mechatronics team began making small changes to the low-voltage systems to comply with FSAE-A technical regulations. This involved changing the precharge circuit, tuning the BSPD current sensor and repairing the APPS plausibility analogue circuitry. None of these subsystems should directly affect the operation of the motor controller. It should be noted that at this point in the year, regenerative braking had not yet been implemented and all testing was conducted with only a drive torque.

Three weeks after the initial NU24 shakedown testing day, the car was returned to race-ready condition and a motor spin-test was conducted the night prior to a testing day. Upon arrival in the morning, a second motor spin-test was conducted where the motor stopped responding to pedal applications. The author connected to the RMS GUI software and repeated the motor spin-test. Upon conducting this it was noticed the inverter had faulted and was reporting an internal diagnostic troubleshooting code.

4.2. Hardware Over-current Fault

Each time the motor loses power the Cascadia inverter reports a run-time fault reading Hardware Over-current Fault. This is defined by the manufacturer under, "This fault occurs when any of the current sensors detect an over-current condition which could be positive or negative. All six over-current faults are ORed together to cause the HW over-current fault."[10] This data is broadcast over the CAN-bus and as such can be recorded by the MoTeC datalogger. Figure 16 shows at the observed time[6] the Run Fault Lo word displays the value 512. This correlates to the hex value 0x200 which refers to the Hardware Over-current Fault.

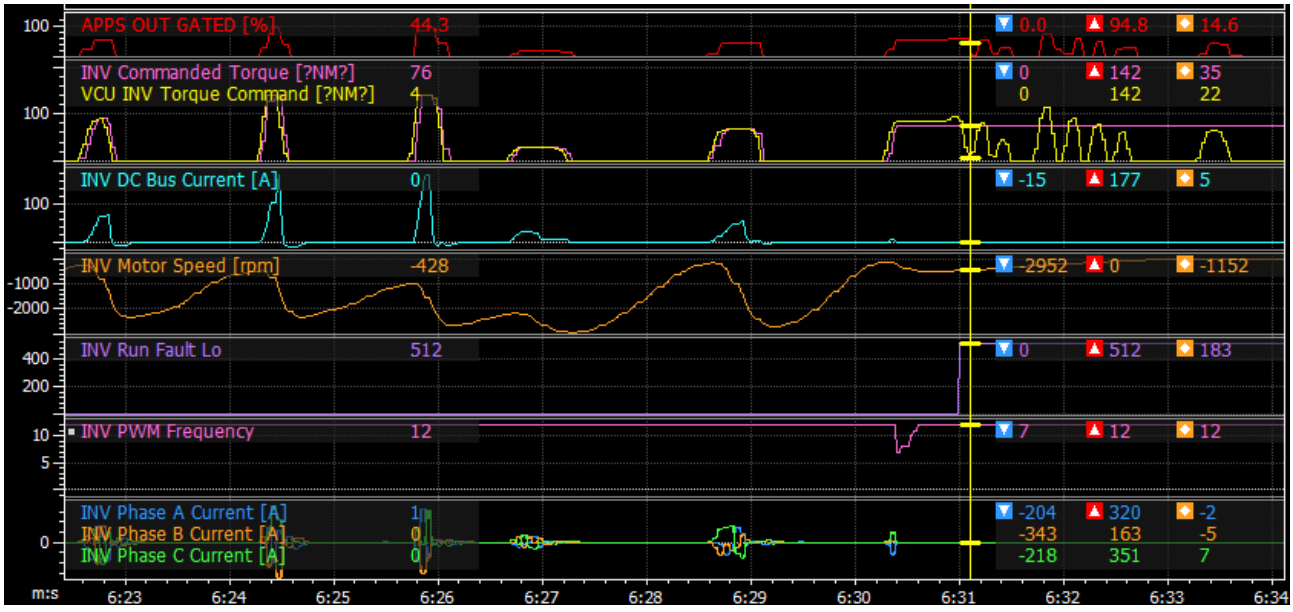


Figure 16: MoTeC Data during an Overcurrent Fault of the CM200DZ.

[6]Illustrated by the vertical yellow line

It can also be seen that prior to the inverter entering a fault state, the PWM Frequency drops to 7kHz. This does not happen on the other pedal applications in the view window of Figure 16. This was a detail that was consistent during all repeated testing that involved the inverter faulting due to a Hardware Over-current Fault.

4.3. Troubleshooting

Cascadia Motion was able to offer a small amount of guidance through an email chain between themselves and the author.[7] It was suggested that the firmware was updated to ensure any potential bugs may be resolved. Additionally, they suggested a review of the resolver wiring harness and whether it had been damaged or routed differently since the inverter had stopped operating. They had confirmed that the EEPROM values appeared to be correct for the implementation of the EMRAX 228 MV with the Tamagawa 5-Pole resolver.

The firmware for the inverter was updated as the documentation mentions improvements to the performance of PWM and stall burst models. The firmware was updated from ID 6526 to ID 6530, however this did not fix the issue and resulted in the motor no longer reacting to any torque requests.

It was then decided to recalibrate the resolver for the EMRAX 228. Upon attempt to calibrate the resolver it was noticed that the magnitude of the voltage feedback speed was double that of the magnitude of the resolver feedback speed. At this point it is not recommended to calibrate the resolver and instead check for correct phase cable installation and motor type EEPROM. The latter was confirmed and it appeared to be a hardware issue within the inverter and the calibration process was attempted. It was noticed however, that there was no number of Gamma Adjust that would result in a correctly calibrated resolver. The inverter would continuously report a value of Delta Resolver equal to zero, regardless of the value of Gamma Adjust.

The author decided to reinstall the EMRAX 188 motor and attempt resolver calibration in the event that the EMRAX 228 may be damaged. Similar behaviour was observed were it was not possible to complete the resolver calibration process. This resulted in no torque response from the motor and the team was not able to spin the motor at all. Attempts were made to run the inverter in speed mode, however no success was had spinning the motor.

The firmware was returned to ID 6526 and the last previous working EEPROM file was uploaded to the inverter and there was no response from the motor. At this stage it appeared that no matter what configuration of firmware, motor or EEPROM settings, the inverter was no longer able to generate any movement out of either of the motors.

An additional side-effect that was noticed was that when the inverter was enabled via the ready-to-drive sequence, a static DC voltage of 290V was measured between the motor phase cables and vehicle ground. This static voltage remained until the BMS would fault and cause the voltage to discharge and disappear.

Alec Chapman, the NU Racing ESO, was contacted it was decided the in order to avoid risking any injury to the team working on the EV Tractive System that the inverter and relevant high-voltage wiring be disconnected.

[7] This email chain is available in Appendix F

4.4. Stealth EV support

After contacting the manufacturer, Cascadia Motion, the author was told that support must be provided by the company that NU Racing purchased the CM200DZ inverter from. Stealth EV offered limited support, asking for a system overview of information such as the battery specs, configuration and system voltage. They had also suggested that the team attempt returning to using the EMRAX 188 motor which had already been confirmed not to be the solution. Additionally they had suggested the author to inspect the high-voltage phase cables for damage and ensure shielding to both the motor and inverter chassis.

This was something that the team had not been implementing as the phase cables are shielded via connection to the inverter inside the provided Rosenberger HPK connectors. Due to the phase connections at the motor being cable lugs bolted to the motor phases, no additional connection for the shielding had been made to the motor chassis. In fear of damaging the existing cables it was decided not to connect this shielding as it was determined it would not likely be the solution to the problem.

It was requested with Stealth EV if the team could have the inverter sent back to the dealer in order to have it assessed for damage under a warranty claim. NU Racing was not able to clearly diagnose what caused the inverter to suddenly fail, however it is believed it was in connected to the change in PWM frequency caused the by logic inside of the inverter.

At the time of writing this report, the CM200DZ inverter is with Stealth EV being assessed for a warranty claim. Alec Chapman and the author have not heard of any results of this assessment, and have not received any update from Stealth EV since September.

4.5. Diagnosis

As a result of communication through Cascadia Motion following the purchase of the CM200DX inverter, the likely failure of the inverter was described to be a loss of current regulation in stall. This is a result of the variable frequency PWM functionality offered by Cascadia Motion that has not been finely tuned to suit EMRAX motors. During the commissioning of the CM200DX replacement motor it was suggested to the team to disable this feature whilst continuing to use the EMRAX motors.

Stealth EV has not been able to comment on the damage that may or may not have been caused to the inverter as a result of this, and is something that will be followed up in the NU25 competition cycle by the future Chief Engineer, Alec Chapman.

5. Commissioning of CM200DX Inverter

5.1. Introduction

With the malfunction of the previous CM200DZ inverter, it was decided that if the team were to compete at the upcoming FSAE-A competition a new inverter needed to be purchased. Due to complications with receiving technical support from Stealth EV, it was decided to purchase a replacement inverter directly from Cascadia Motion while awaiting assessment on the CM200DZ. Thanks to feedback from the manufacturer, the Mechatronics team leader Alec Chapman purchased a new CM200DX inverter to be used in NU24.

The DX model inverter has a lower operating voltage of 50-480VDC compared to the DZ model of 200-840VDC. This is within the tractive system voltage of NU24 so would not effect its operation with the current accumulator configuration. Additionally, the DX model inverter has a higher peak and continuous motor current rating. This is beneficial in the event that the over-current fault being experience is a genuine limitation of the inverter and not a system hardware malfunction. Details on the difference between inverter specifications can be found in Appendix A.

5.2. CAN Bus Troubleshooting

BMS OKHS Drop-out

Upon reviewing CAN logs from MoTeC i2, it was discovered that the BMS OKHS would drop out periodically as the car entered Ready-to-Drive (RTD). This signal was traced back to the Orion BMS2 where it has two inputs. Both come from the BMS 26 pin connector as multi-purpose outputs (MPO's) 1 and 2. Of this, MPO2 is a CAN controlled output with active-low which represents the state of TEMP OKHS and MPO1 is the active-high BMS error signal output. These signals in parallel results in the following logic: **(NOT TEMP_OKHS) OR BMS_Error_Signal**

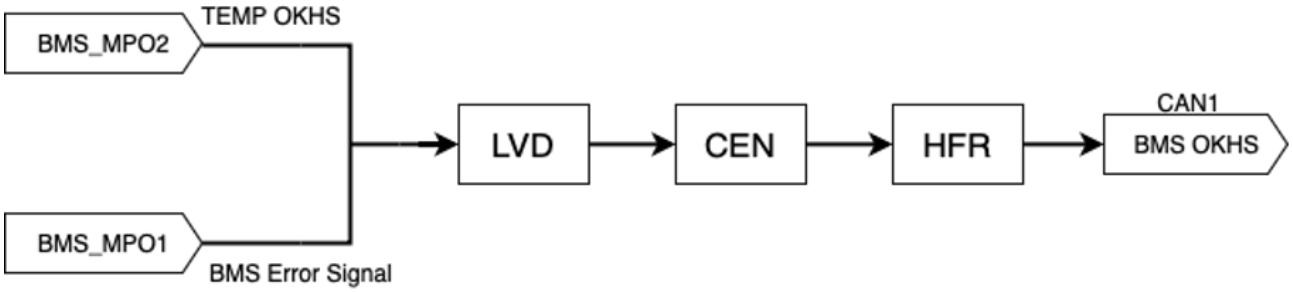


Figure 17: Block Diagram of the BMS sent CAN Messages.

To validate the BMS error signal, the TEMP OKHS output was removed and the BMS OKHS message was logged. This resulted in the BMS OKHS no longer changing to zero under the same conditions, as it no longer relies on other components of the CAN-Bus. It was noticed however, that a lot of the 'heartbeat' signals from the different electronic nodes on the vehicle were not being updated over CAN. These heartbeat signals are sent from each electronic node as an indication that the on-board Teensy is still communicating over the CAN-Bus. For these signals to stop being updated it indicates that the CAN transceiver is broken, the Teensy is no longer powered, or there is a problem with the CAN communication wires.

CAN-Bus Overload

The CAN-Bus was further logged over MoTeC, where it was noticed that when entering RTD (as indicated by both RTD Button and RTD State) the entire CAN-Bus is experiencing an outage of messages for several seconds. The Kvaser CanKing was connected to monitor the live traffic of the bus. It was noticed that the network maintains between 20-30% utilisation under normal operation. Then, when entering RTD, it has hundreds of error frames and the utilisation increases to greater than 95%. The CAN output window in Figure 18 shows that there are hundreds of error frames sent over the bus at the instant the RTD button is pressed. These error frames are defined by Kvaser as:

> "Error frames indicate a problem with the network topology/configuration.
> You need to make sure:
> 1. The CAN bus is properly terminated. You should have 120 Ohm termination at the furthest points of the CAN network between CAN_H and CAN_L. This would mean you should measure approximately 60 Ohms between CAN_H and CAN_L."[11]

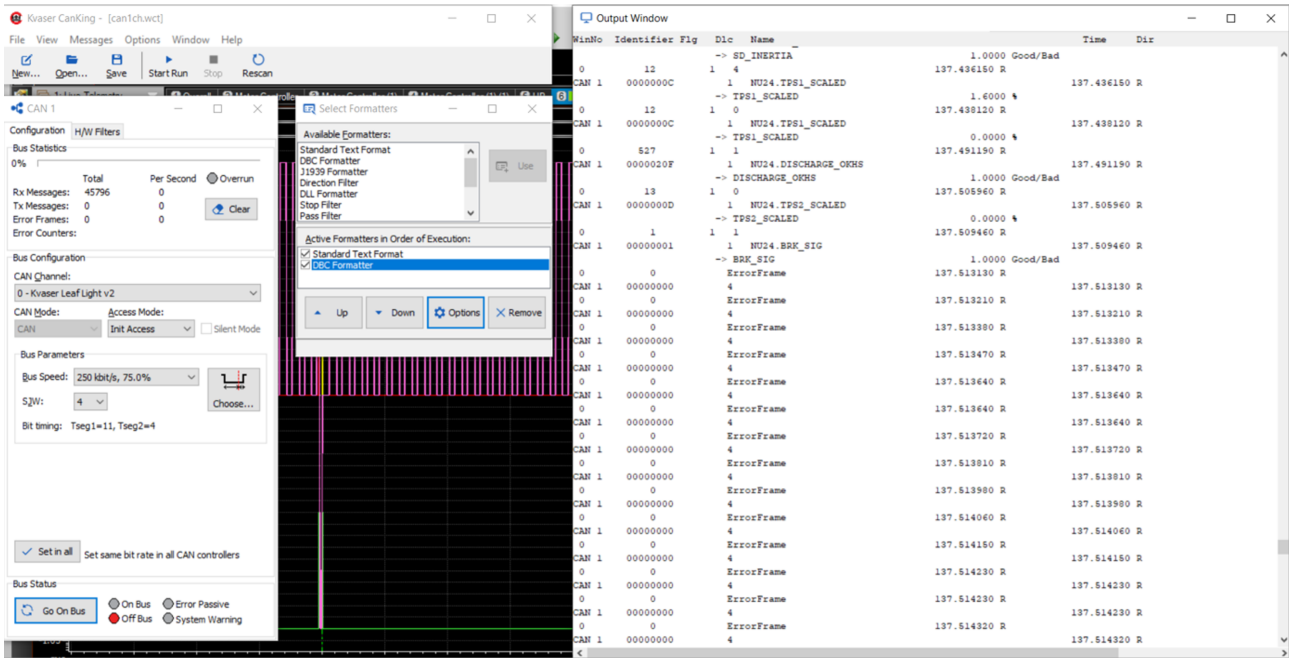


Figure 18: Error Frames being Generated and Observed over Kvaser CanKing.

The CAN-Bus was then tested for resistance between the CAN High and CAN Low wires where it was noticed that there was a 40 Ω total resistance. This was then tested with different nodes disconnected to discover which contained termination resistors. This resulted in finding 120 Ω termination resistors in the PEN, LVD and BMS. This was solved by disconnecting the LVD termination resistor through the PCB-mounted termination switch. This did not resolve the CAN error frames and was decided it was not the root cause of the issue.

Inverter CAN Command Message

It should be noted that the RTD state sent over CAN is used in the logic for the Cascadia CM200DX inverter enable command. This logic takes place in the PEN and is shown below.

```
inverterEnable = (int)rtdState * (int)sdBots;
```

The variable sdBots represents the state of the shutdown circuit on NU24 after the Brake Over-Travel Switch (BOTS).

It was decided to try and inject an **rtdState = 1** message onto the CAN bus at the relevant ID 519 to have the PEN transmit an **inverterEnable = 1** message for the inverter. This caused the sounder, which typically indicated RTD initiating, to make noise for a short moment, however the car remained out of RTD. The team then tried to hold the RTD Button for 5 seconds and noticed that the CAN outage was longer than when having a momentary press. The DEN code was investigated as this is where the **rtdState** message is sent from, however no fault was found in the implementation. The PEN code, specifically the CAN command message, was reviewed. This implementation was also correct, so it was decided to look at the Cascadia Motion supplied CAN database file.

There were minor differences to the implemented DBC.[8] The Cascadia database was copied over to NU24.dbc. The inverter's low-voltage connection was then unplugged from the CEN, and attempted to switch the car into RTD. This time there was no CAN overload, no error frames, and the car successfully entered RTD. The low-voltage connections were reconnected and the inverters high-voltage DC bus connection was removed. This had the same result as prior, and the car was successfully able to enter RTD. It was noticed however, that the inverter had a low voltage fault on the DC bus which is expected without high-voltage present.

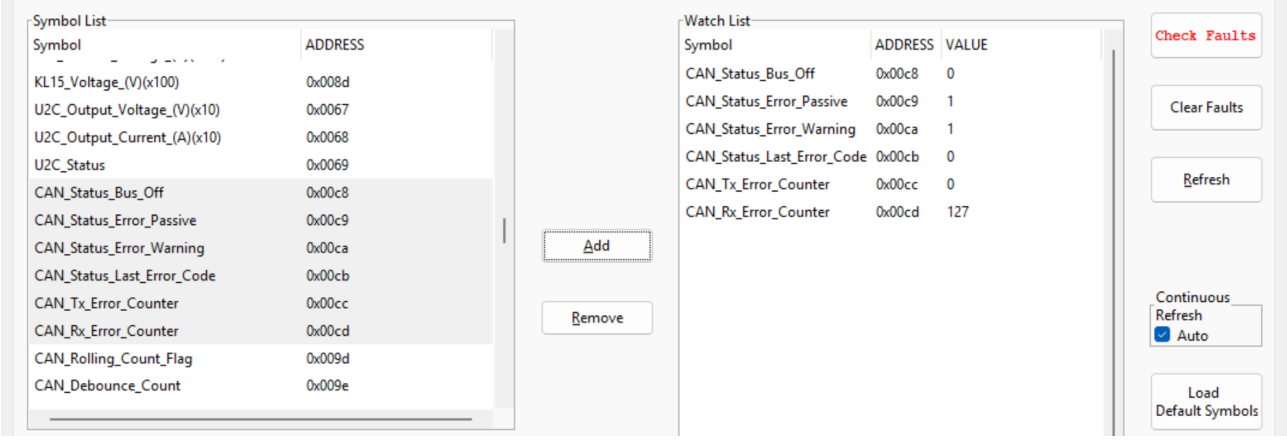


Figure 19: CAN Status Messages being shown on RMS GUI.

The RMS GUI application was used to monitor the CAN status values during the standard RTD process with all inverter connections plugged in. Figure 19 shows the idle values displayed by the GUI while outside of RTD. The inverter then enters a CAN time-out fault after the EEPROM designated duration for CAN timeout. After exhausting most other paths, it was decided to revert the firmware to 6526 from 6530. This showed no change in operation. EEPROM values from the last operational track day in July were uploaded to the inverter, however this did not change the operation. All code on the PEN was reverted to previous working code and the issue persisted.

[8] It should be noted that the CM200DZ previously used, was running an older 6526 firmware version during the operation for NU23 and NU24 shakedown testing.

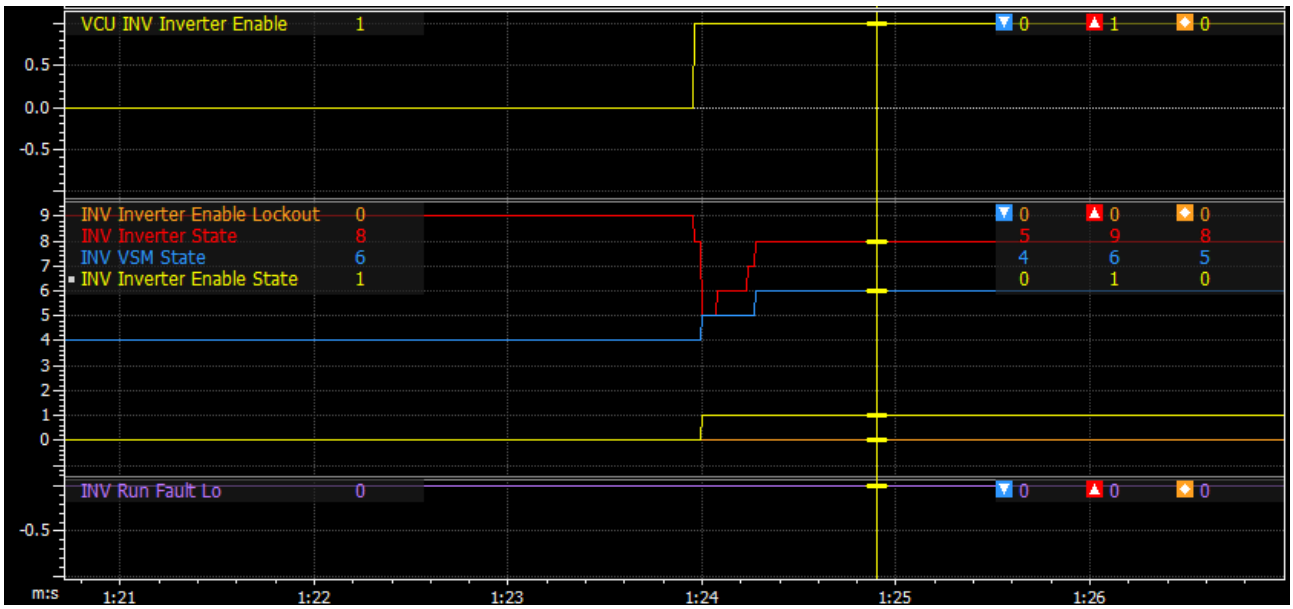


Figure 20: MoTeC Data of Successful Inverter Enable Sequence

Several RTD attempts were data-logged through MoTeC to compare to a previous example of a successful inverter enable. Figure 20 illustrates a successful sequence from NU23 which results in the inverter being enabled. It can be seen the the Inverter State and VSM State both go through a sequence of stages before coming to rest at their enabled values.

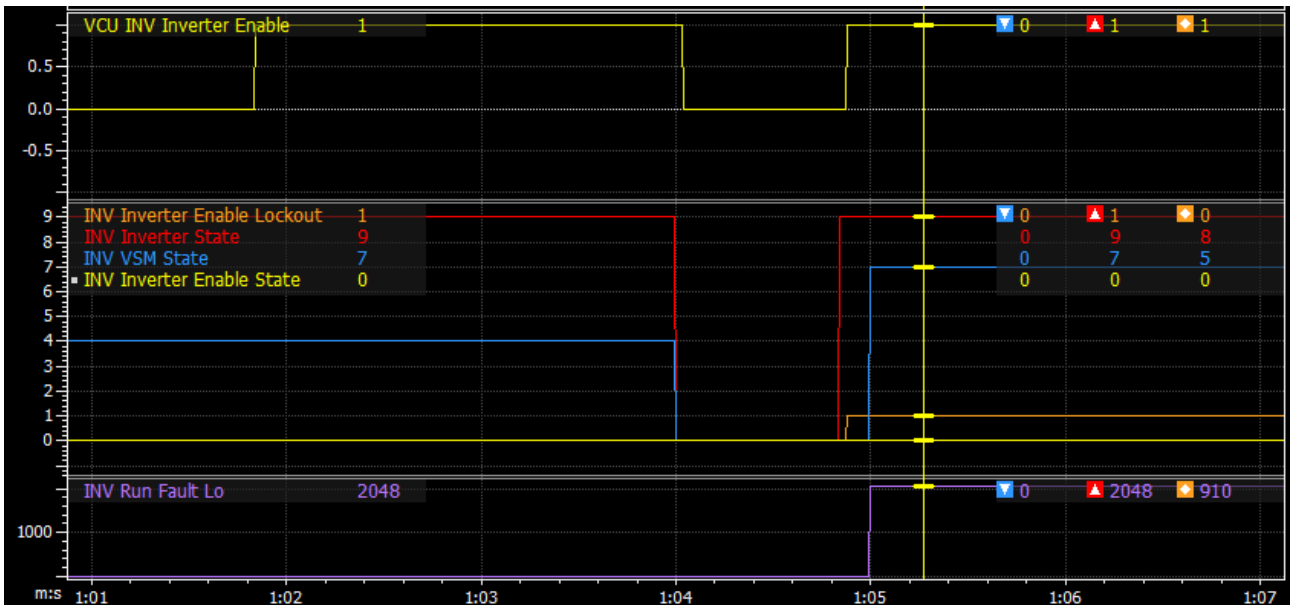


Figure 21: MoTeC Data of Failed Inverter Enable Sequence

Alternatively, Figure 21 shows the recorded RTD initiation during testing with the new CM200DX inverter with the CAN drop-out visible at the timestamp 1:04. This causes the inverter to enter a fault state indicated by the INV Run Fault Lo. This value of 2048 corresponds to the CAN Message Timeout Fault. After this occurs, the inverter enters an Inverter Enable Lockout state where it is no

longer able to be enabled until it has been power cycled. It appeared that the fault that was causing the systems CAN-Bus to become overwhelmed and drop out is related to the CAN communication of the CM200DX, specifically the messages it is trying to read from the CAN-Bus. This is indicated by the CAN Rx Error Counter shown by the inverter's GUI. This is reinforced by the fact that when the motor controller is either disconnected, unpowered, or in an error state, the CAN-Bus does not suffer from error frames. The transmission messages working correctly and some received messages working correctly led the team to believe that the onboard transceiver was not damaged.

Inverter Rolling Counter

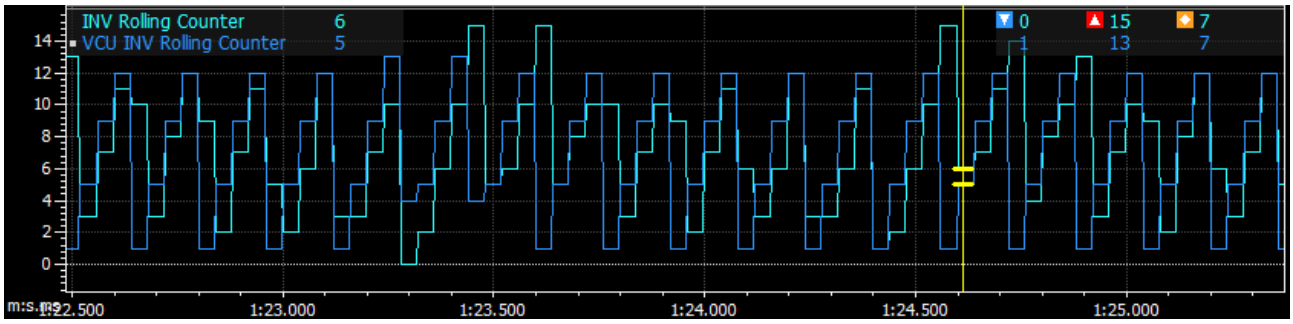


Figure 22: MoTeC Data illustrating Rolling Counter Implementation

The addition of a rolling counter in the CAN command message helps when diagnosing the CM200DX interacting with the other CAN nodes on NU24. When sending a command message from the PEN, it will include an unsigned 4-bit rolling counter integer (between 0 and 15). This rolling counter is then compared to the stored value in the inverter, which is then increased by 1 if it is a match, in expectation of the next CAN command message. This cycle repeats with the inverter rolling counter number trailing the command message rolling counter by 1 if all messages are being received in sequence. If the rolling counter numbers do not match at the time of comparison, the inverter stored number is not increased. This behaviour is shown in Figure 22 where it was working successfully at FSAE-A 2024 Competition.

Figure 23 illustrates the rolling counter and all other CAN signals failing entirely when the Inverter Enable sequence begins. This persists until a CAN time-out error takes place, shutting off the inverter and restoring the CAN-Bus. It was noticed that the connection between the DCDC negative terminal (which is representative of the chassis common ground node) and the Powerbox was 44.1kΩ. This was of concern due to the potential for transients resulting from poor grounding. It was decided to incorporate grounding wires between several components to create low-resistance paths to ground. This was done using 8-gauge wire and sanding a portion of powder coating off of the chassis to ensure a good connection.

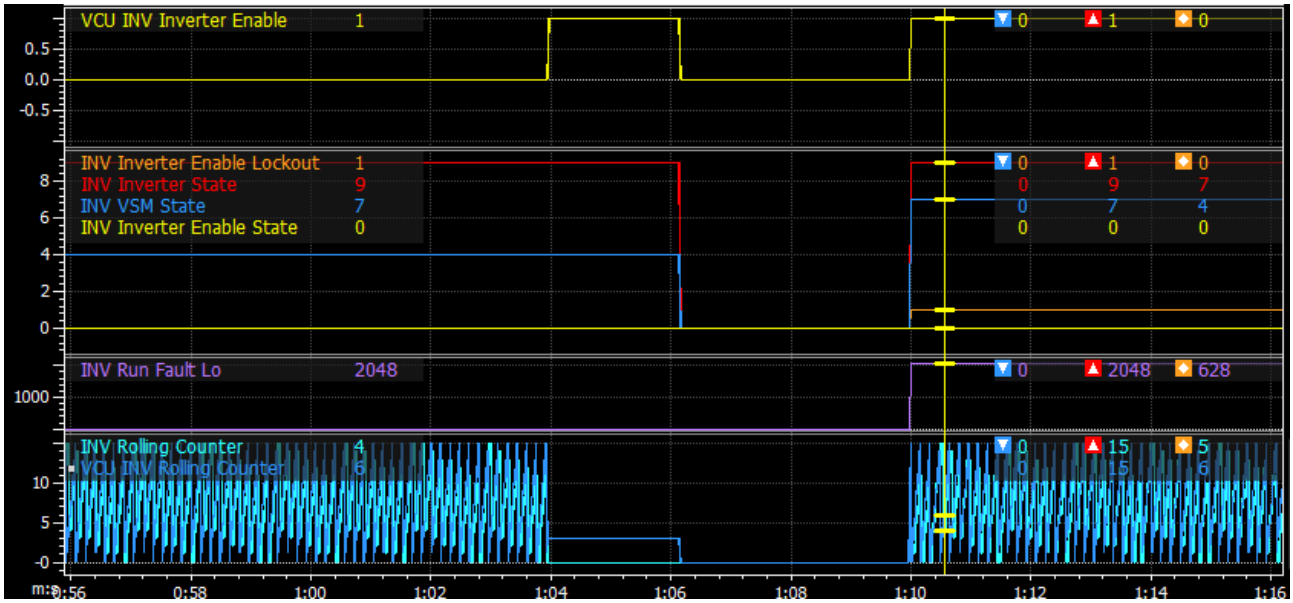


Figure 23: MoTeC Data of Rolling Counter during a Failed RTD Sequence

Following adding grounding wires, the CAN signals now remain after the inverter enable state changes. The INV Rolling Counter data is heavily disrupted and not following the uniform pattern evident prior to the inverter enable message. This is indicative that the noise has been reduced but there is still some persistent electromagnetic interference on the CAN network that needs to be addressed.

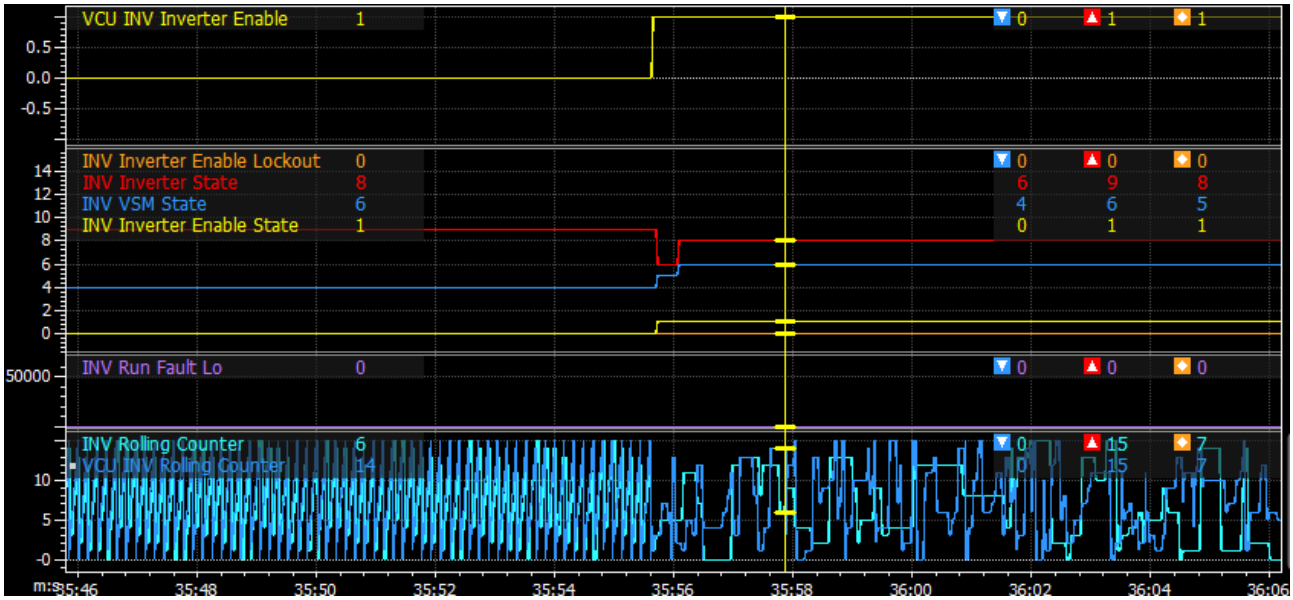


Figure 24: MoTeC Data of Rolling Counter during a Successful RTD Sequence with Noisy CAN-Bus

5.3. Electromagnetic Interference Diagnostics

Identification

Upon discovering this CAN instability, it was decided to start monitoring the CAN High to CAN Low with an oscilloscope. This was used to quantify the noise that was being observed on the network in order to discover its origin on the vehicle. Upon observing the signal, it was discovered that there was a periodic noise occurring at a frequency of 24kHz. There were higher frequency transients in this observed noise, however what is important is that 24kHz is exactly twice the programmed PWM frequency of 12kHz for the inverter. This is important as the inverter will switch its internal IGBT's at its PWM frequency, on both the rising and falling edge of each pulse. This means that for a 12kHz PWM frequency, the frequency of either a rising or falling edge is 24kHz.

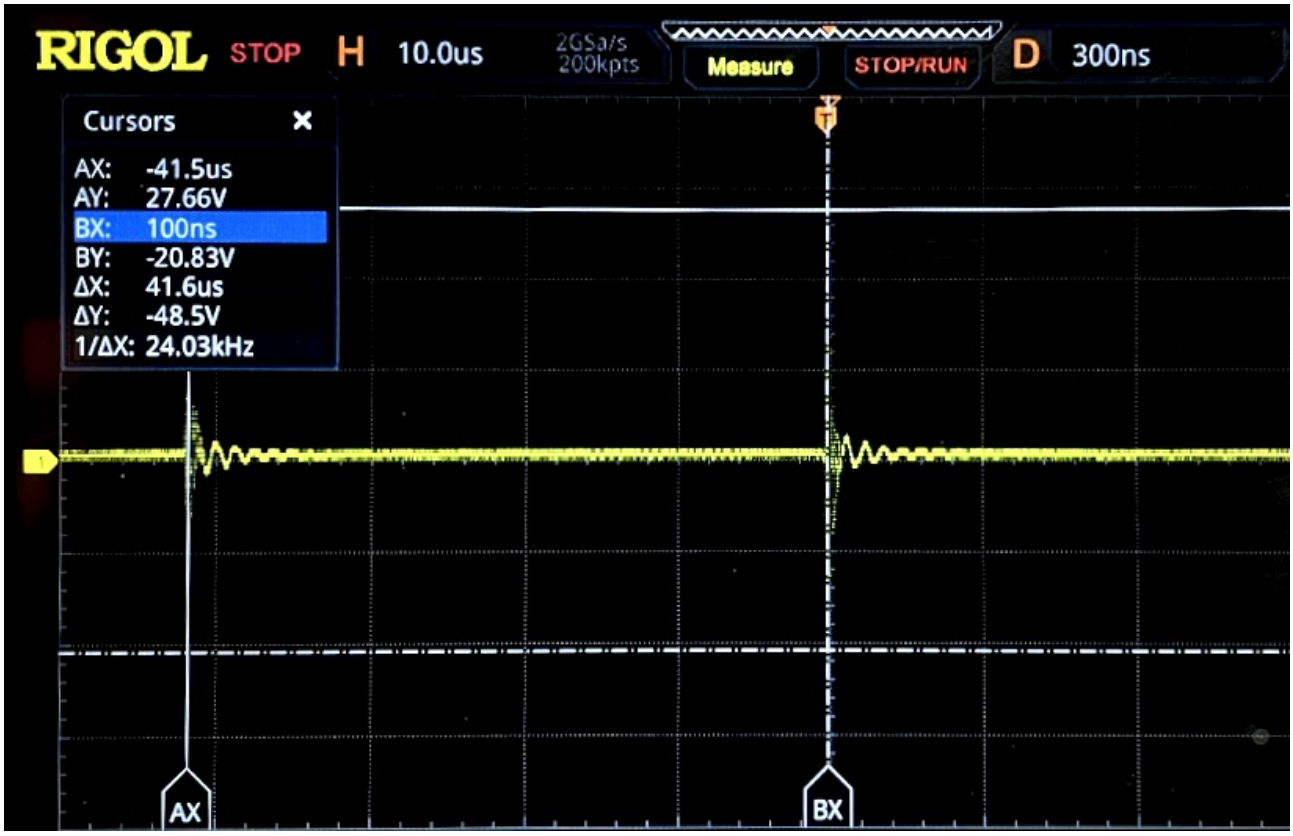


Figure 25: Oscilloscope Data referencing CAN High to CAN Low with 24kHz Noise.

By taking several cursor measurements on the oscilloscope, Figure 25 shows that the $1/\Delta X$ value for the period noise aligns exactly to 24kHz. The peak-to-peak amplitude of these transients was also measured to be 48.5V. This is extremely high given that a typical CAN-Bus waveform will have a peak-to-peak amplitude of just 1V.

Based on the information at hand, a shield was constructed to be placed over the CAN High and Low signals from the CM200DX. This was done to protect the twisted-pair wires from inductive noise generated by the internal switching mechanisms of the inverter. The shield was not effective however, and the noise was persistent after following the documentation provided to the team by Cascadia Motion.

Accumulator Diagnosis

This EMI was further investigated by eliminating the accumulator from the equation. This was done by powering the high voltage input to the inverter using a bench power supply that is otherwise used for charging NU24's accumulator. The power supply is capable of a high-voltage output which is necessary for surpassing the inverters low-voltage cutoff. This cut-off was set to 100 volts, but to mimic the system, a voltage of 410V was chosen on the power supply as it closest represented accumulator voltage at the time. This is important as the noise is only present when the inverter is given its enable command. In a fault state, the enable command is ignored and there is no change to the system.

It is key to keep as many variables constant to ensure that the experiment isolates a singular variable to narrow down the source of the EMI. With the high-voltage power supply switched on, the command message was sent, and the noise was detected immediately on the CAN reference, as well as when referencing +12V to Ground. This ruled out any involvement of the accumulator in being the source of the EMI. This does not mean it was not involved in amplification of the noise; however, it was not the root cause.

PEN Diagnosis

The PEN was able to be tested in isolation by sending the remote Inverter Enable request using a CAN expansion port on the CEN. The author utilised Kvaser CanKing to send the unique message sequence required to enable the inverter. For the inverter to be enabled it needs to receive an 8-byte CAN message at the hex id 0x0C0, corresponding to decimal 192, which contains the command message. This command message includes information such as requested torque, direction, command mode, torque limits and most significantly, the inverter enable bit. Only the inverter enable bit needs to be changed, with all other components of the message left as 0-bits. Beforehand, a blank command message must be sent as per the CAN protocol documentation provided by Cascadia.

The following two messages were sent over the CAN bus, where the ID is the decimal ID of the message, DLC is the data length code in bytes, and the Command message is the value of each byte in the message:

1. **Protocol message:**
   ID: 192
   DLC: 8
   Command Message: 0 0 0 0 0 0 0 0
2. **Inverter Enable message:**
   ID: 192
   DLC: 8
   Command Message: 0 0 0 0 1 0 0 0

Byte.Bit	Name	Format	Description
0,1	Torque Command	Torque	Torque command used when in torque mode.
2,3	Speed Command	Angular Velocity	Speed command used when in speed mode. Starting in version 2048 and 651D any Speed Command transmitted while in Torque mode will over-ride the Max Speed EEPROM parameter and provide a new maximum speed limit. The maximum speed limit will revert to the default EEPROM parameter value when the inverter returns to Speed mode.
4	Direction Command	Boolean	0 = "Reverse", 1 = "Forward"
5.0	Inverter Enable	Boolean	0 = Inverter Off, 1 = Inverter On
5.1	Inverter Discharge	Boolean	0 = Disable Discharge, 1 = Enable Discharge
5.2	Speed Mode Enable	Boolean	0 = Do not over-ride mode, 1 = If controller is in torque mode then controller will change to speed mode. This is a mode over-ride bit that will change the mode from torque to speed only.
6,7	Commanded Torque Limit	Torque	If set to 0, the default torque limits sets in the EEPROM parameters are used. If set to a positive number then the Motor and Regen Torque limits are set to the torque value sent.

Table 3: CAN Message Structure for the Cascadia Motion Inverter Command Message

This message was successfully read by the inverter, and it completed the enable sequence without the need for the RTD command from the PEN. The noise was however immediately picked up by both the CAN and power transmission wires as measured by the oscilloscope. This ruled out the PEN for being the source of the EMI.

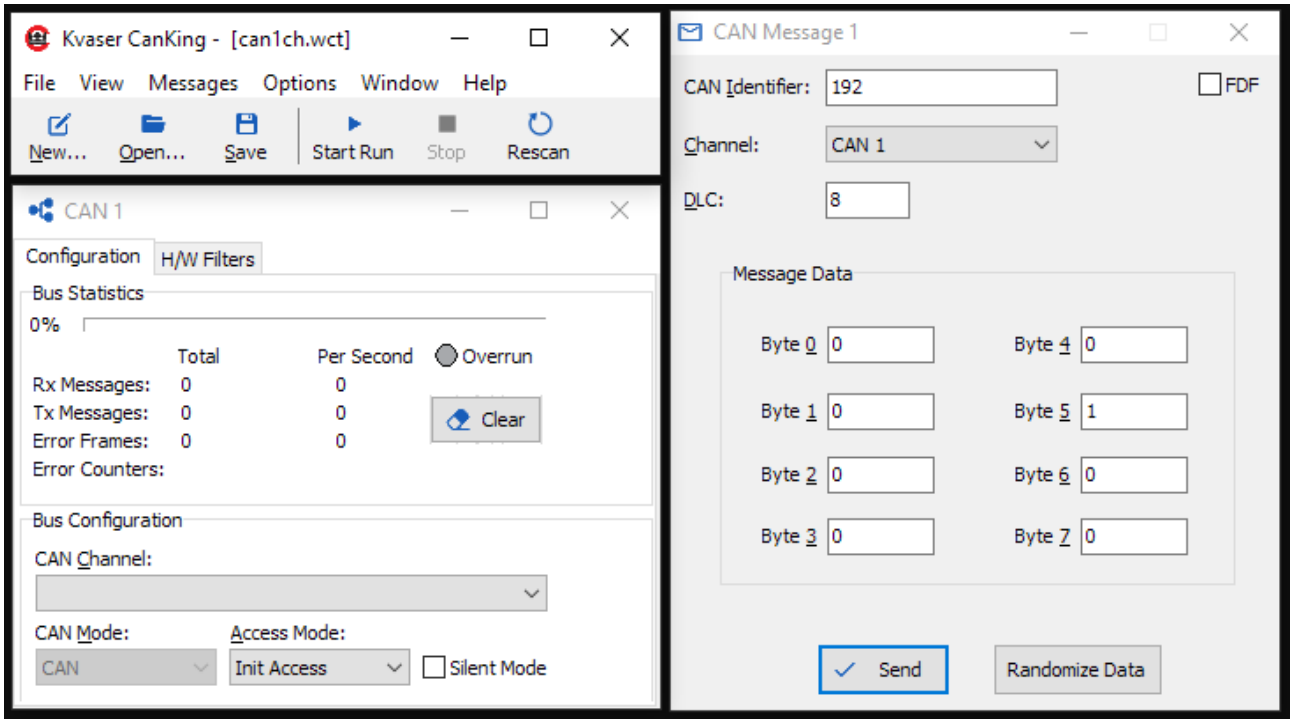


Figure 26: Inverter Enable message in Kvaser CanKing Universal Message Window.

CEN Diagnosis

Due to the accumulator and the PEN being ruled out as the vessel for EMI, it was decided that eliminating the CEN from the equation would be highly indicative of the EMI origin. This is due to the CEN being the primary PCB for both sending power and a large amount of CAN messages throughout NU24. It is regarded as the master board for the vehicle, where the other nodes act as satellite units. Removing the CEN from the equation would require disconnecting all other nodes as they would not function without the CEN. This was only made possible due to the inclusion of the Kvaser CanKing command message ability, and being able to use a bench power supply for both low-voltage and high-voltage supplies. The CAN expansion port on the CEN remained as the test point for the oscilloscope as in previous testing, and the CEN itself remained in position on top of the DCDC and inverter.

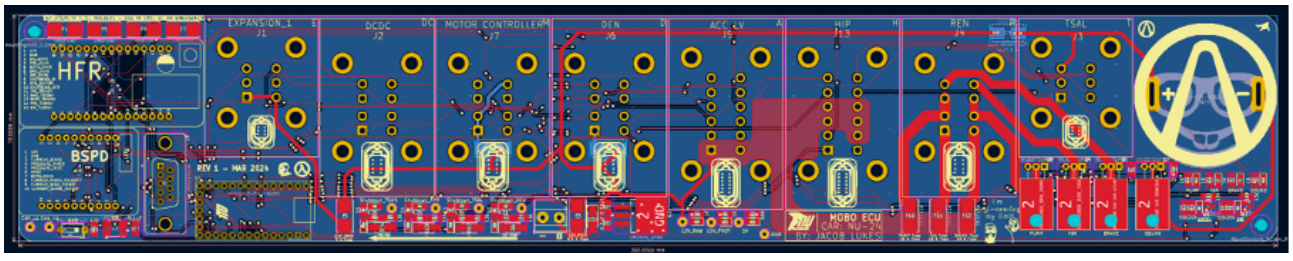


Figure 27: PCB View of the CEN as seen in KiCad.

When the inverter was enabled, the noise appeared on the oscilloscope. This was a key result from this experiment as it demonstrated that there was at least a component of the EMI that was purely inductive. This meant that there did not have to be a physical connection between components to

result in noise transmission. This experiment isolated the CEN as not being the originating source of noise, however raised suspicions that it may be acting as an antenna to capture the EMI given its position and shape in the car.

Pictured in Figure 27, the CEN is a long thin PCB that measures approximately 400mm in length. Its potential to act as an antenna is very real. Another experiment was conducted similar to that mentioned above that would test the area that was affected by EMI. This involved enabling the inverter while probing the CAN lines on the CEN with it completely disconnected from the vehicle. The inverter was enabled with the CEN in its usual position in NU24, it was then slowly picked up and removed from the car where no significant drop in noise was detected.

It was then transported away from the vehicle where after approximately 1 metre from the chassis the noise saw a sharp reduction in amplitude and eventually was undetectable through visual means on the oscilloscope. This was a fantastic result, as it was the first time during all testing that the author had managed to see no noise present whilst the inverter was enabled. It did show however, there was a large amount of inductive EMI leaking into the surrounding components.

5.4. Addition of EMI Shielding

With the noise identified as at least partially inductive, it was key to remove all components from the vehicle and lay them out on a bench to test the extent to which the proximity effected the magnitude of EMI generated. It would also serve as a controlled environment to probe for signs of EMI and allow for the source to be identified.

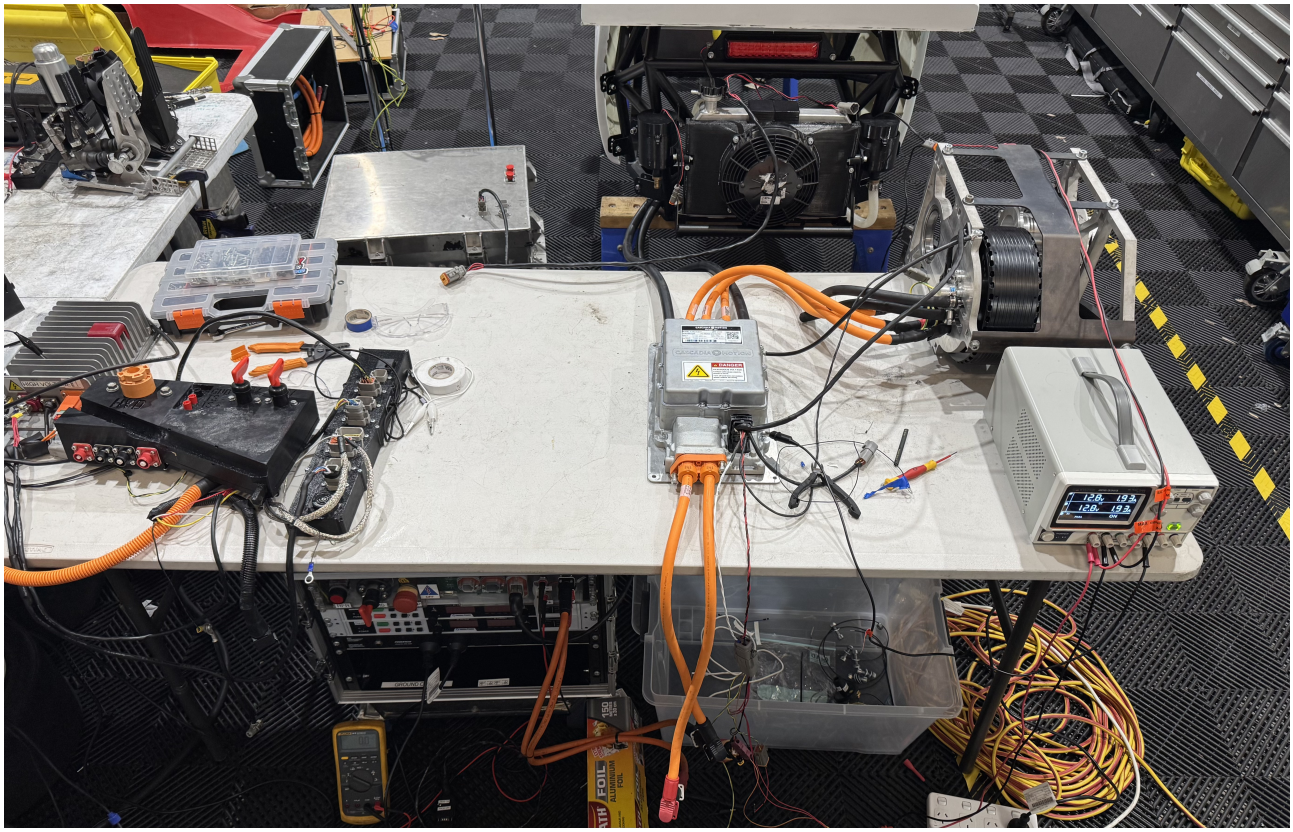


Figure 28: Bench Testing Configuration for Motor and Motor Controller

Figure 28 illustrates how the electrical system of NU24 was able to be reconstructed on the bench to

conduct testing in a controlled environment. This involved setting up a tabletop immediately behind the vehicle as it was necessary to always maintain coolant flow through the inverter whilst it was enabled. This was achieved by keeping the radiator and electric pumps mounted to the chassis whilst running the coolant loop out the rear of the vehicle. Upon constructing the vehicle on the bench-top, it was discovered that the system was reporting an IMD fault.

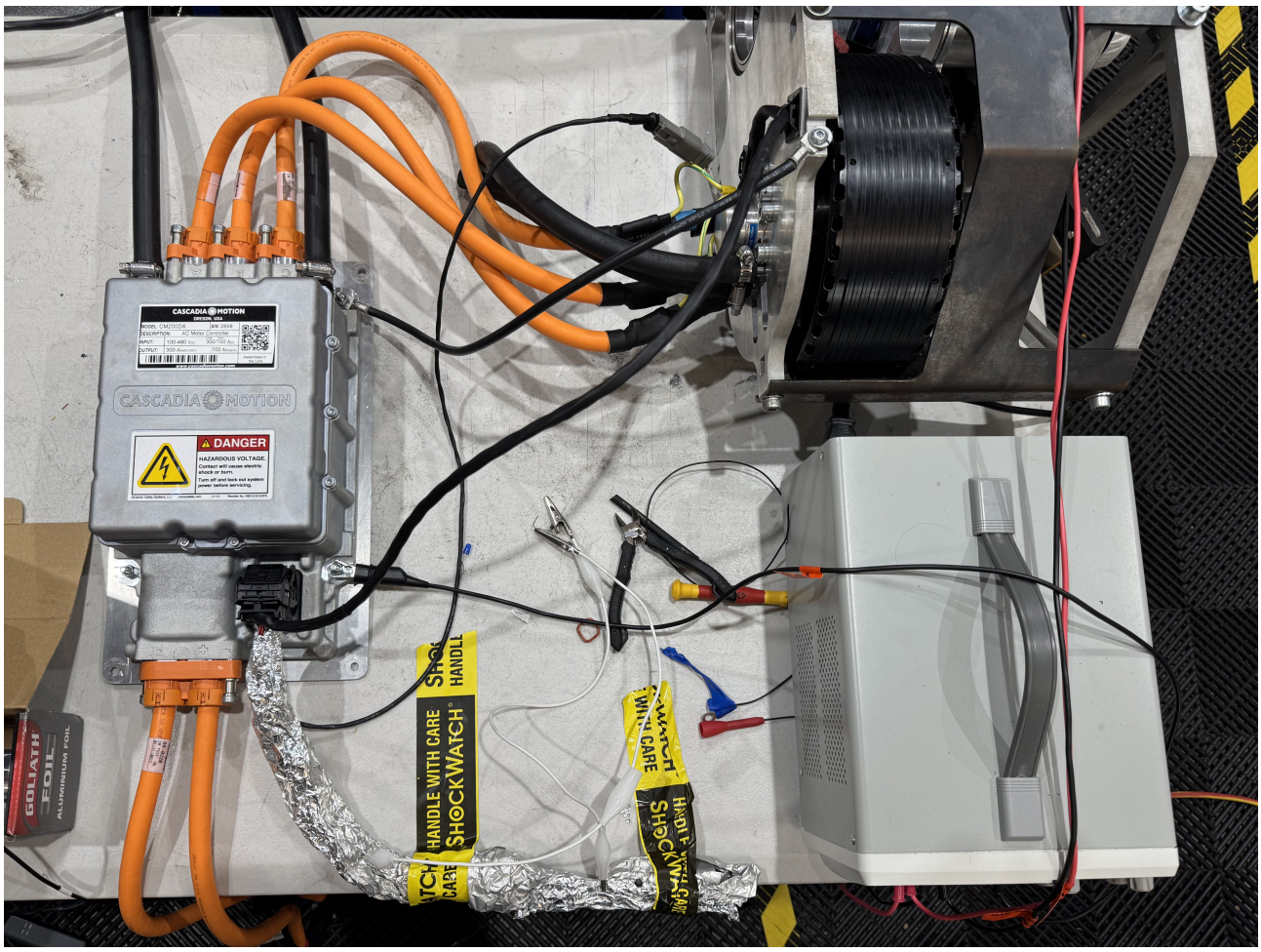


Figure 29: Shielding on Motor Controller Low-Voltage Connector using Aluminium Foil.

CM200DX Low-Voltage Wiring Harness

The inverter low-voltage harness was shielded using aluminium foil, this was unsuccessful in mitigating the EMI when using chassis ground as the shield drain. The author then used the isolated CAN ground supplied by the CM200DX connector as the shield ground reference. The noise measured by the oscilloscope didn't change significantly, however the error frames reduced from 100 to 20 per second. The CAN wires for the motor controller low-voltage harness are a twisted pair of 20-gauge wire. There was no shielding for the pair so the author removed the existing wiring and replaced them with a length of shielded and insulated twisted-pair wiring. This was configured with the following:

- **White & Blue Wire** - CAN High
- **White Wire** - CAN Low
- **Wire Shield** - CAN Ground (Not Connected to CEN)

The error frames remained at 20 per second with the shielded CAN wires. The oscilloscope was removed whilst still monitoring the CAN-Bus utilisation and it was noticed that the error frames dropped to 0 per second. The CEN, DEN, PEN and TSAL circuitry were all added back into the system and the error frames remained at 0 per second. When the accumulator was reintroduced the CAN error frames increased back to 10 per second.

Due to not knowing what the error frame count was during regular operation of NU23, it was decided to begin attempting to spin the motor. This was done by enabling the inverter using Kvaser CanKing due to the ability to send consistent, static messages. The inverter was enabled and a torque request of 10Nm was included in the command message. The motor would spin until it reached one of the motor poles and the it would lock in place. The torque request was increased to 20Nm and it was noticed that the motor would now shudder about a pole and make a significant amount of vibrations.

Resolver Harness Repairs

Resolver readings had been observed to be bad when enabling the inverter. The Delta Resolver reading was showing rapidly-changing changing random values. With the motor rotating, the voltage feedback speed was observed to vary in values between 0 to -3000rpm whilst the resolver feedback speed was not changing from 0rpm. When spinning the motor by hand, the delta resolver value was intermittently reading correctly, moving through the 0-360 degree range. Suspicions were confirmed when the inverter reported a resolver disconnected fault.

The Raychem DR-25 heatshrink was removed from the resolver shield and it was noticed that all of the solder joints connecting the resolver to the motor controller harness had failed. This meant that a majority of the resolver connections had become open-circuit and were causing poor readings. This was resolved by removing the existing solder joints and creating new, secure solder connections. These connections were given added strain relief with adhesive heat shrink to ensure the solder joints did not fail again during competition. A spare resolver harness was made to be used as a spare in the event that one were to fail during a critical time during competition where it would not be viable to redo the solder connections.

5.5. Cascadia Motion Support

Having exhausted diagnostic steps within the scope of the authors understanding, a lengthy email chain took place with Cascadia Motion, the manufacturer of the inverter.[9] They were able to offer back and forth support such as an analysis of our high-voltage topology to ensure that the new inverter would not be damaged in a similar fashion to the CM200DZ. The system was deemed to be safe, including the high-voltage disconnecting during operation caused by the FSAE shutdown circuit rules compliance.

They made several recommendations such as changing the EEPROM values surrounding variable PWM frequency control. This was due to the limited tuning Cascadia Motion has been able to do with the EMRAX series of motors. They added that the stall and low speed operation is particularly stressful to the inverter as the current can remain in one phase for long duration. This recommendation meant that it would be safer to disable variable PWM control and set the normal operational PWM frequency to 12kHz. All changes that needed to be made are shown in Table 4.

[9]This email chain is available in full in Appendix H

EEPROM Description	Old Value	New Value
Serial_Number_EEPROM	1933	2858
PWM_Minimum_Variable_Freq_EEPROM_(kHz)	6	12
PWM_Maximum_Variable_Freq_EEPROM_(kHz)	17	12
PWM_Stall_Freq_EEPROM_(kHz)	6	12
PWM_Var_Freq_Mode_EEPROM	1	0
Gamma_Adjust_EEPROM_(Deg)_x_10	716	-1625
Coast_Lo_EEPROM_(V)_x_100	92	0
Coast_Hi_EEPROM_(V)_x_100	108	0
Accel_Max_EEPROM_(V)_x_100	285	0
Pedal_Hi_EEPROM_(V)_x_100	500	0
Braking_Torque_Limit_EEPROM_(Nm)_x_10	150	0

Table 4: Comparison of EEPROM values after Cascadia Motion recommendations

They had also recommended to attempt grounding the motor phase cables at both the motor and inverter. Due to the Rosenberger connectors used in the inverter, they are self grounding be construction. The motor side however required stripping back of insulation and crimping lugs directly to a piece of shielding. This shielding was then grounded to the powerbox to ensure grounded connection at both ends of the shield. This is important for tractive system grounds due to the high frequency EMI that tends to be generated around the phase cables. This additional shielding did not solve any of the CAN-Bus issues however, and has since been reverted back to only grounding the shield at the inverter.

Cascadia Motion also recommended removing the CAN Ground connection at the CEN. This is different from the phase cable shielding where it is recommended to only ground the shield at the inverter end. This is due to the CAN ground pin supplied on the CM200 low-voltage harness plug is an isolated ground reference and should not be shorted to vehicle ground.

It was then arranged to have a Microsoft Teams meeting with Andrew Louie, a Controls Engineer at Cascadia Motion, to discuss the problem and follow a troubleshooting procedure. This meeting took place on the 26th of October 2024, just 6 weeks prior to competing at FSAE-A 2024. Shortly after explaining the behaviour being observed, Mr Louie notified the author that the Group Number of the firmware did not match that recommended to be used for the specific motor and resolver. In the software manual it states that:

> "**Important:** The hex file with the tag 'Group_1' in the filename should be used for motor types between 0 and 59. The hex file with the tag 'Group_2' in the filename should be used for motor types starting from 60 and onward." [10]

The firmware was updated to reflect the above information and the resolver calibration process was able to be completed as per the supporting documentation. Within the hour, the author had successfully calibrated the resolver and the motor was spinning smoothly and without failure. There was also no reported fault for the over-current issues that had been seen on the CM200DZ and the team was able to continue development and testing before the 2024 FSAE-A competition.

6. Integration of Regenerative Braking to NU24

6.1. Introduction

A regenerative braking research project was assigned to the author in the early stages of the 2024 FSAE campaign. This involved conducting research in the viability, effectiveness and risks involved with incorporating regenerative braking into NU24. It was part of the starter-project induction all final-year-project students undertake.

The primary advantage of having regenerative braking is the ability to recapture kinetic energy into electrical energy to be stored in the battery. This provides a direct performance increase in the Endurance event where teams are measured on their Efficiency throughout a long distance timed stage. By recapturing kinetic energy through the tractive system rather than losing it to heat through friction braking, the goal is to achieve a much higher efficiency score by having a smaller net energy usage. Braking performance gains may be seen due to the increased force acting to remove kinetic energy which will slow the car down. This comes at the consequence of moving the braking bias rearward as the regenerative braking will only apply to the rear wheels.

The other added benefit of having a well tuned regenerative braking system is the ability to run the car in a single-pedal configuration. This happens when the amount of braking force generated purely by regeneration is great enough to slow the car down before and during cornering. This is highly beneficial during the Endurance event as it makes the long and strenuous drive easier for the drivers by not having to consider braking points and being able to lift off the accelerator pedal to engage the regenerative braking instead.

6.2. Research Project

Due to a majority of NU Racing team members consisting of final-year students, it is important to maintain a high level of knowledge transfer from year to year. This is especially important when it comes to operating with OEM components such as the Cascadia Motion range of inverters. Due to receiving the CM200DZ inverter in December of 2023, the year prior to the author did not have a significant amount of time to document the operation of the inverter to be passed onto the 2024 team.

When the author begun conducting research into the regenerative braking project, it was first decided to fully document the range of Cascadia Motion documentation and compile them into a singular document. This would allow any problems that arise to be effectively investigated, and allow for effective knowledge transfer when the author graduates from the team. This document condenses the knowledge of 35 user manuals and documentation is available in full in Appendix J. Whilst it was written for the CM200DZ, all documentation carries over to the CM200DX inverter that had been commissioned.

Additionally, the supporting documentation for the Orion BMS2 was studied to understand what additional calculations could be made when sending the current limiting message from the BMS. It was found that this allowed taking into account the State of Charge of the accumulator as well as the pack temperature to provide further de-rating of the maximum available current.

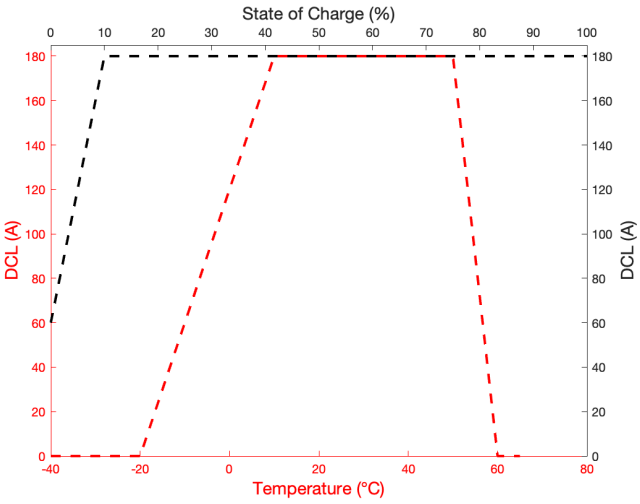


Figure 30: Orion DCL Graph

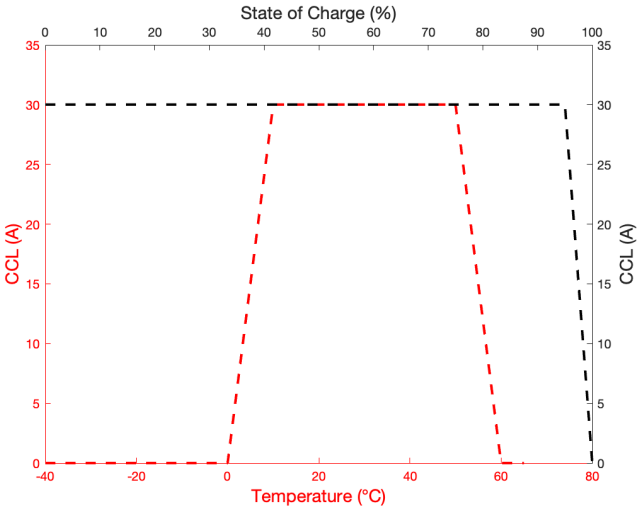


Figure 31: Orion CCL Graph

The two graphs shown in Figure 30 and Figure 31 illustrate the pre-configured Orion BMS de-rating algorithm for discharge and charge limit control. Both graphs limit the amount of current available at temperatures outside the regular operating range. This is to protect the cells from being damaged due to different chemistry properties outside of the safe operating temperature range. It can be seen that the discharge current is reduced at low State of Charge to reduce the risk of individual cells going undervoltage. Similarly, the charge current is reduced at high State of Charge to avoid having the cells experience an over-voltage when charging or regenerative braking.

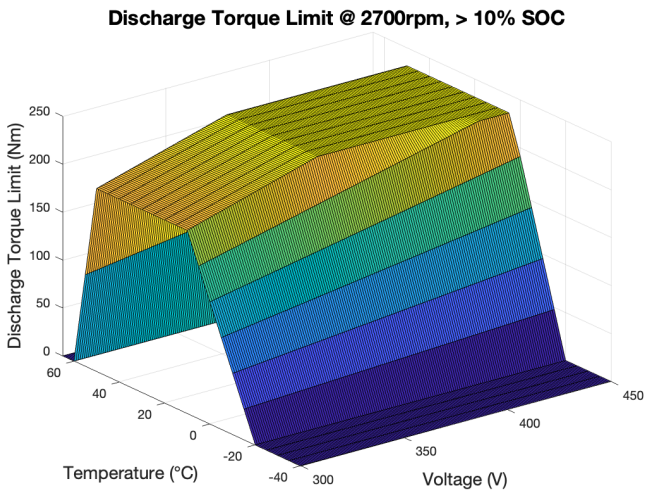


Figure 32: Discharge Torque Limit at 2700rpm.

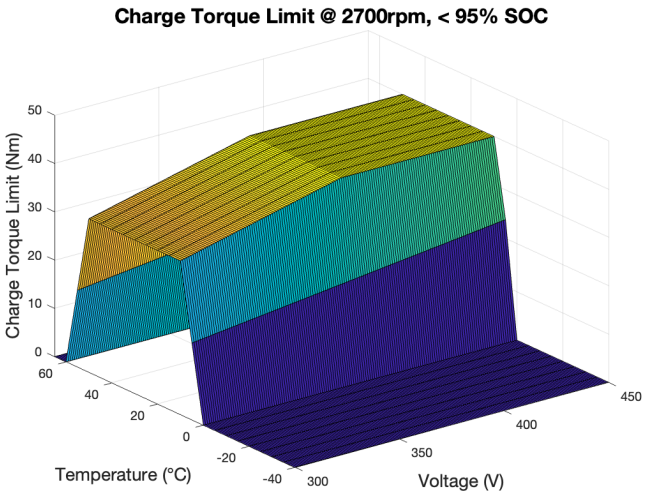


Figure 33: Charge Torque Limit at 2700rpm.

The resulting Figure 32 and Figure 33 show the torque-limiting map at a fixed motor speed of 2700rpm. This map will evolve with motor speed, as the torque available is a function of maximum current and motor speed. The sloping profile will remain the same whilst the 'plateau' will get lower as motor speed increases.

This torque limit can also be shown with a constant temperature rather than motor speed to produce the mapping in Figure 34. This is shown at 40°C where the effect both motor speed and voltage have on the maximum torque is largely evident. This relationship forms as a result of Equation 2. It is through this graph, which has been configured to normal operating conditions for pack temperature, that the importance of pack voltage is seen. At higher voltages, while remaining below 95% State of Charge, the magnitude of regenerative braking torque available is greater than at lower voltages. This highlights the significance of beginning a competition event at a high pack voltage as there is a clear performance gain to be had. This relationship follows for discharge torque limits, where at high pack voltages the vehicle is able to sustain its peak torque up to higher motor speeds as there is less current required compared to lower voltages.

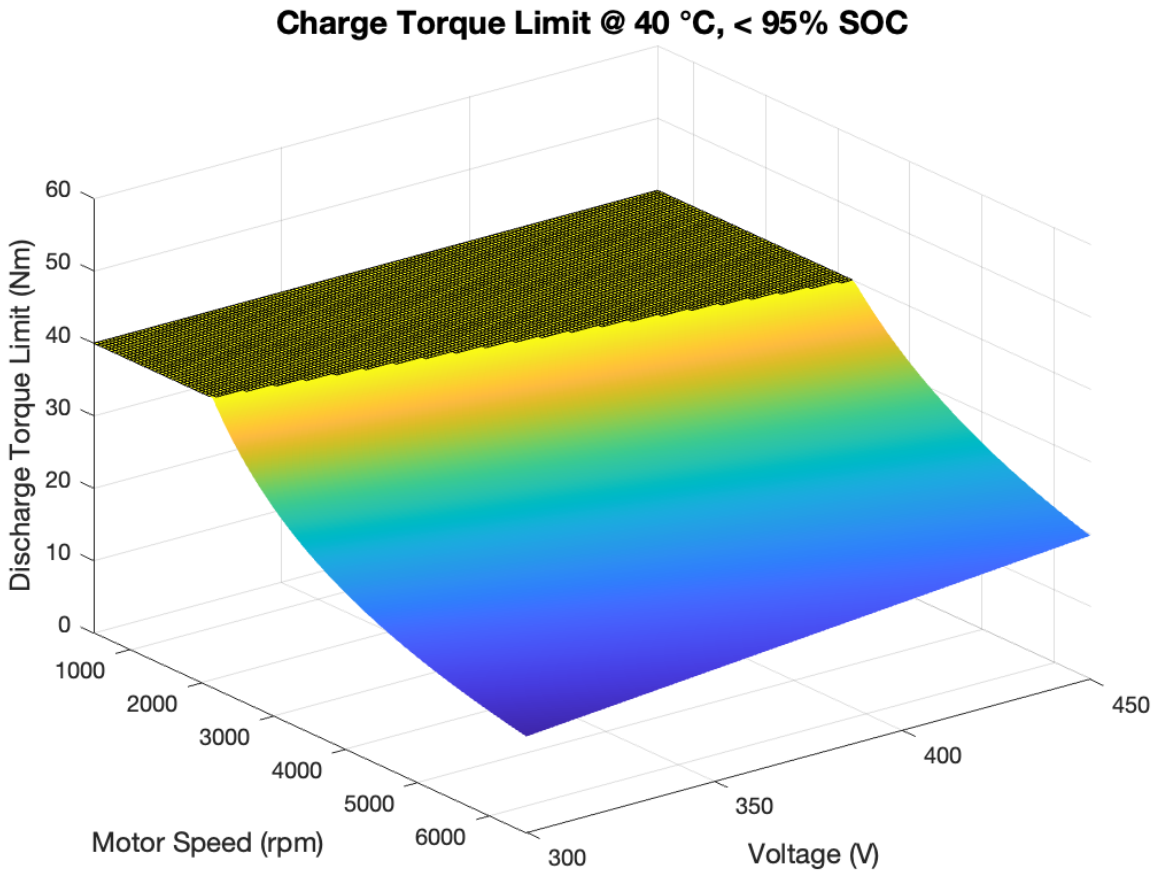


Figure 34: Charge Torque Limit mapped against Voltage and Motor Speed at 40°C.

The shaded region indicates where the torque is limited as a result of user configuration. In Figure 34 this is at 190Nm which was determined to be the safe mechanical operating limit for the NU24 powerbox. Should this be increased to 231Nm as per the manufacturer maximum torque for the EMRAX228, the effects of voltage and motor speed will be emphasised due to the smaller plateau region and torque fall-off effects being felt in a wider range of conditions.

Due to the much smaller EMRAX188 that was used throughout the operation of NU23, the risk of exceeding maximum current recommendations was far smaller. Refering to Figure 8, it can be seen that current limiting is only required above 90km/h, which the car rarely exceeded. In comparison, when configured to 231Nm, the EMRAX228 requires current limiting at 65km/h for the same voltage. This highlights importance of configuring the current limiting before making high torque requests in both forwards torque and regenerative braking scenarios.

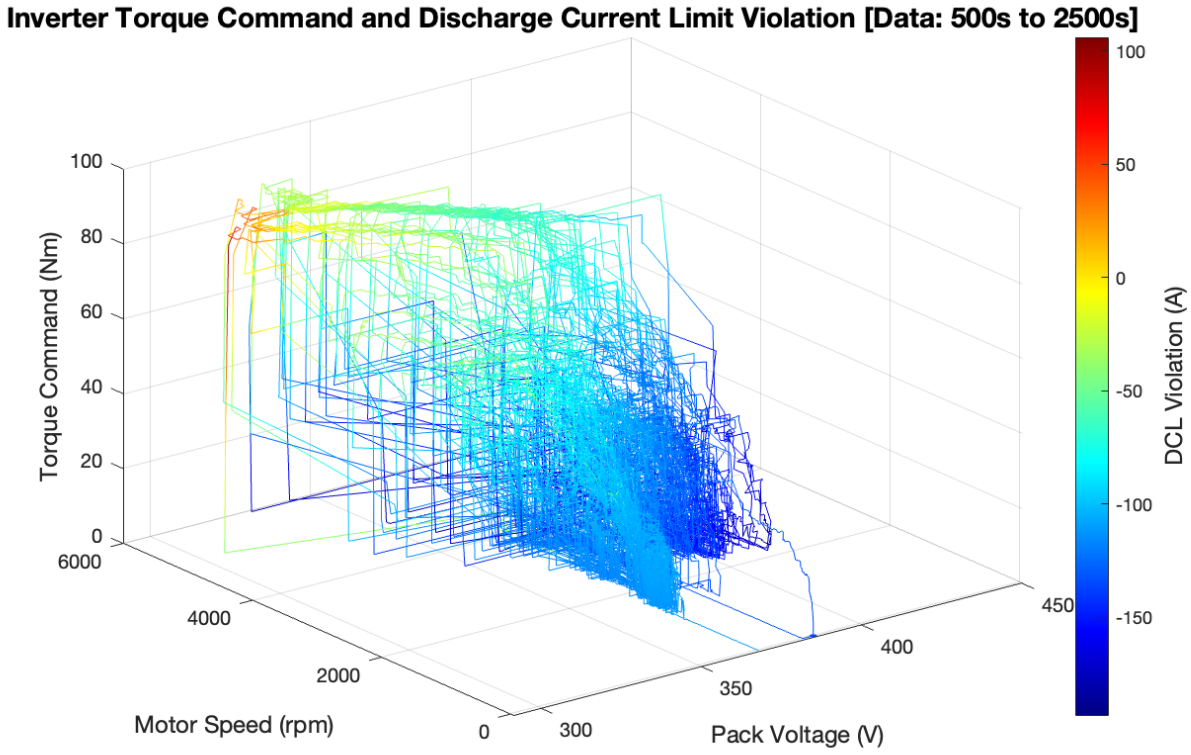


Figure 35: Analysis of NU23 Discharge Current Violation set to 180 Amps.

It was decided to study the behaviour of NU23 during a track day conducted at the ICT carpark on the University of Newcastle's campus. The colourmap shown on the righthand side of Figure 35 represents how much the DC current being used by the motor varies from the theoretical discharge current limit.

$$\text{DCL Violation} = \text{I}_{\text{Pack}} - \text{DCL} \tag{3}$$

Equation 3 shows discharge current limit violation and is calculated using methods previously outlined in this section, including temperature and state of charge. It is evident that while the system is above a pack voltage of 350V there is no violation of the discharge current limits. This is observed by the colour of the 3D plot above 350V not exceeding the orange portion of the colourbar, representing a net 0 violation of the discharge current limit.

As the voltage begins to fall below 350V, it can be seen the there is a net positive discharge current limit violation. The areas to highlight are those with a high torque command, high motor speed and low pack voltage. This is the condition that will result in the greater current require to meet all three conditions. This is the condition that is most likely to result in damage to the cells within the accumulator is the current draw is not managed by incorporating discharge and charge current limits.

6.3. Multivariate Current Limiting

It is crucial that there is a current limit set when implementing regenerative braking to ensure there is no damage to the accumulator cells due to exceeding manufacturer specifications. This means that fixed current limits must be set for charge and discharge currents. Given the datasheet for the Enepaq VTC6 cell modules[10] used in NU24, the maximum fast charge current is 30 Amps and the maximum discharge current is 180 Amps.

Originally, it was attempted to utilise the Orion BMS2 support provided by Cascadia Motion to implement the dynamic current limits. This was configured as per documentation and can be observed in Figure 36. It was noticed, however, that the current at which the inverter was limiting was equal to half of that being sent by the Orion BMS. This was able to be solved by simply doubling the value internally in the Orion BMS before transmitting it to the CAN-Bus.

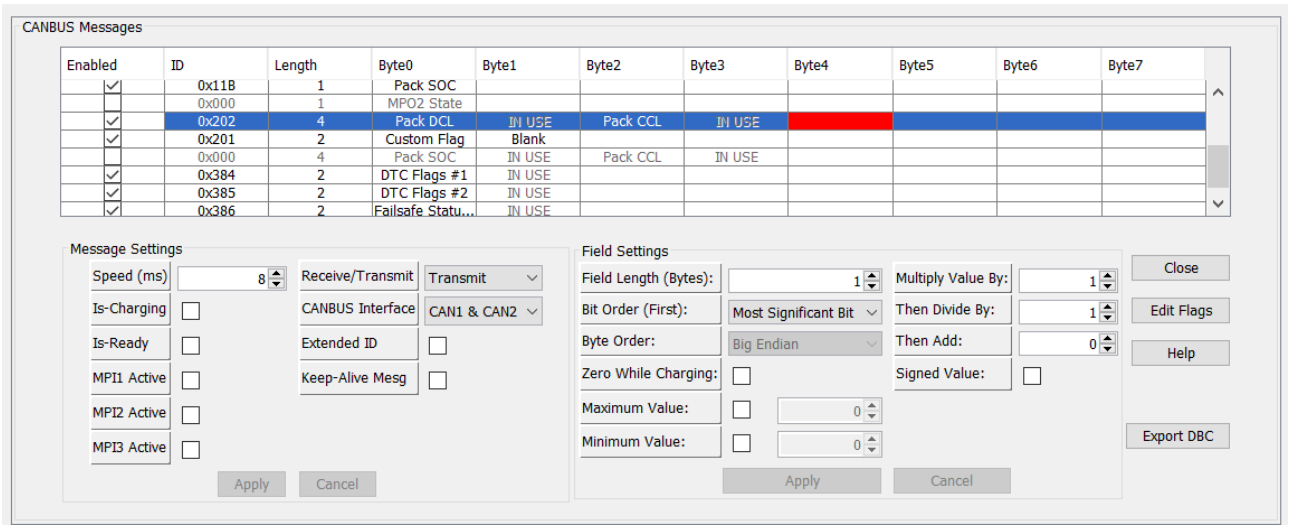


Figure 36: Current Limit Settings configured on Orion BMS2.

It was decided that a dynamic charge current limit was required to handle other variables to protect the system from various conditions such as over-voltage. This may happen when the pack is at a high state of charge and enters a period of regenerative braking. This inrush of current into the pack causes the voltage to spike and it is possible to exceed maximum cell voltage of 4.2V[11].

This implementation was not able to be achieved through sole use of the Orion BMS CAN transmission. It was decided to remove the current limiting messages from the BMS and migrate the functionality to the PEN. This involved the replicating the BMS current limit messages, which had to be added to the NU24_CAN.dbc file. Below is the BMS DCL and CCL .dbc addition.

```
BO_ 514 BMS_Current_Limit: 8 BMS
    SG_ BMS_Max_Discharge_Current : 0|16@1+ (1,0) [0|1000] "current:A" Vector__XXX
    SG_ BMS_Max_Charge_Current : 16|16@1+ (1,0) [0|1000] "current:A" Vector__XXX
```

This would enable the author to implement multivariate current limiting by considering other variables in the current limit calculations.

[10] The Enepaq VTC6 Cell modules datasheet can be found in Appendix D

[11] This would occur if the pack voltage exceeds 453.6 Volts (given a low cell-voltage delta)

```
void dataHandling(void) {
  if (SoC >= 95.0) {
    CCL = 1; // 0 will not work
  }
  else if (SoC > 75.0) {
    CCL = 30.0 * (1.0 - ((SoC-75.0) / 20.0 ));
  }
  else {
    CCL = 30;
  }
  .
  .
}
```

Code Block 1: SOC Based Current Limiting over CAN Bus

Code Block 1 shows the PEN code which handles the charge-current limiting for the regenerative braking command under high state of charge. This implementation was added to the code a week before competition as there was little time to do any field testing due to other low-voltage compliance issues. The code works by limiting the charge current to 1 amp above 95% State of Charge (It cannot be set to 0 as this disables the limit entirely by Cascadia Motion configuration) to disable any significant regenerative braking and avoid causing an over-voltage fault. Below 95%, the charge current is linearly increased from 1 amp to 30 amps until 75% State of Charge where the charge current limit is set to a static 30 amps. This code is especially important as the car begins its competition endurance event at 450V pack voltage which will require no regenerative braking until the state of charge begins to decrease. Causing an over-voltage condition would cause the vehicle to stop on track and result in a disqualification from the Endurance event.

The discharge current limit was set to a static 180 amps due to the extremely rare likelihood that the car would see low enough state of charge to warrant a risk of under-voltage conditions. It was decided that there was not enough time to test and validate the under-voltage current limiting and that it would not be implemented for FSAE-A 2024.

6.4. Motor Controller EEPROM Changes

Several EEPROM changes had to be made when transitioning over to a regenerative braking enabled vehicle. The CAN_BMS_Limit_Enable_EEPROM is required to indicate to the motor controller that it should be expecting discharge and charge current limits on the allocated BMS CAN channel. Even though these are not being sent from the BMS any more, this must be enabled as the same channel is being used for the transmission of current limits from the PEN. The Amp_Hours and DC_Volt_Limit were changed to better reflect the pack configuration during the commissioning process.

EEPROM Description	Value without Regen	Value with Regen
Regen_Torque_Limit_EEPROM_(Nm)_x_10	10	400
CAN_BMS_Limit_Enable_EEPROM	0	1
Amp_Hours_EEPROM_(AHrx10)	10000	180
DC_Volt_Limit_EEPROM_(V)_x_10	4500	4600

Table 5: Comparison of EEPROM values after addition of regenerative braking

6.5. Regenerative Braking PEN Code

When implementing regenerative braking there are several options for how it will be deployed. These will vary based on the criteria and desirables of the end user, varying from motorsport to the regenerative braking found in a road car. Different methods of deployment incldue:

1. Deployment only under mechanical braking - The regenerative braking torque is only applied when the user has their foot on the brake pedal. The torque may be adjusted proportional to brake pressure and results in greater braking force while shifting braking bias rearward.
2. Deployment at low accelerator pedal position - The accelerator pedal range is split into two components, where the forward torque is requested for the 'top' of the accelerator pedal range and regenerative braking force scales towards the 'bottom' of the pedal range. This requires setting a zero-torque crossover point, for which the pedal switch from regenerative braking to forward torque. This can result in single pedal driving where both accelerating and decelerating are able to be actioned from the accelerator pedal.
3. Deployment on hand-operated control - The torque request for regenerative braking is incorporated on a separate analogue signal device, such as a hand lever. This would allow the driver to make torque request at their own discretion when entering corners, but allow them to coast during the corner apex or when desired. This provides greater control of the regenerative braking, however requires additional hardware to generate the input request.

It was decided that due to the pre-existing hardware and PEN code implementations, that option 2 would lead to the greatest competitions points increase for the required difficulty of integration. This option required only a software solution of creating an acceleration pedal mapping to suit a single-pedal in both forward torque and regenerative braking.

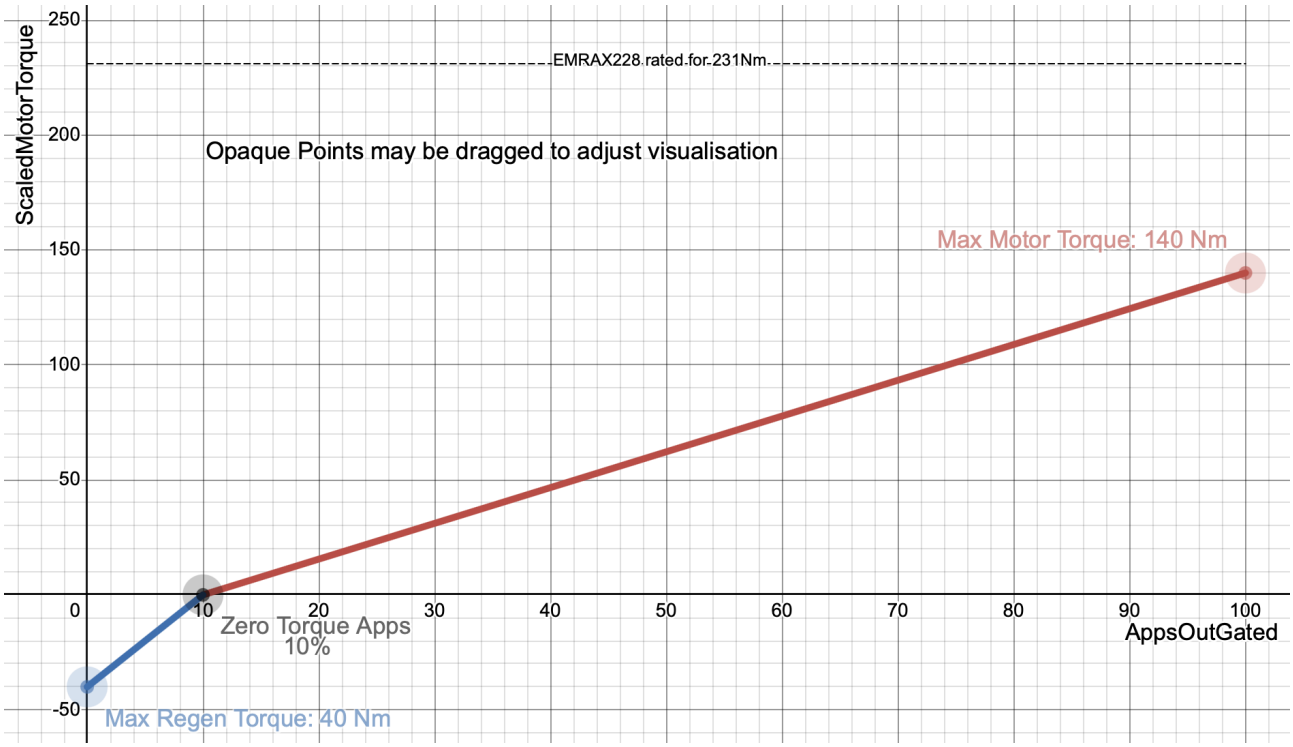


Figure 37: PEN Throttle mapping function visualisation

The throttle mapping function in Figure 37 is able to be viewed in an interactive browser environment on Desmos through: https://www.desmos.com/calculator/jevbkcgozk. For ease of user configuration, three user-defined variables are included at the top of the PEN ECU code.[12]

```
33 #define ZERO_TORQUE_APPS 10
       // Adjustable to desired zero torque crossover point as a function of APPS %
34 #define MAX_MOTOR_TORQUE 180
       // Adjustable to desired motor torque (Nm) [MAXIMUM VALUE = 231]
35 #define MAX_REGEN_TORQUE 40
       // Adjustable to desired regen torque (Nm) [MAXIMUM VALUE = 50]
```

Code Block 2: PEN Throttle mapping function inputs

These three variables allow the user to configure to the maximum forward torque, maximum regenerative braking torque, and the point in the acceleration pedal where the torque will transition from regeneration to acceleration. The torque is then linearly mapped between the zero-torque crossover point to each maximum torque value. It is a discontinuous function which means these three points are not required to be collinear. This enables custom setting of each parameter and allows for tuning of the accelerator pedal based on driver feel.

This is then implemented through Code Block 3, where nested if statements ensure that the resulting throttle map is accurate and FSAE-A rules compliant. Further considerations include the addition of an appsTrail boolean evaluation and the requirement that regenerative braking is only allowed for vehicle speeds above 5 km/h.

```
void sendTorque(void) {
  .
  speedKph = (motorSpeedRPM * 60 / (46/14)) * (3.14 * 16 * 0.0254 / 1000);
  // Kilometres/Hour = WheelRevs/Hour * Kilometres/WheelRev
  .
  if (appsOutGated >= ZERO_TORQUE_APPS) {
      // Positive torque
      scaledTorqueCommand = ((appsOutGated - ZERO_TORQUE_APPS) / (APPS_CEIL -
          ZERO_TORQUE_APPS)) * MAX_MOTOR_TORQUE * 10;
    } else if (appsOutGated < ZERO_TORQUE_APPS && appsTrail != 0) {
      // Regenerative braking torque
      if (speedKph > 5) { // Compliance rules
        scaledTorqueCommand = ((appsOutGated - ZERO_TORQUE_APPS) / ZERO_TORQUE_APPS) *
            MAX_REGEN_TORQUE * 10;
      } else {
        scaledTorqueCommand = 0;
      }
    } else {
      // No torque [ONLY OCCURS IF APPS DIES]
      scaledTorqueCommand = 0;
    }
  .
  .
}
```

Code Block 3: PEN Throttle Mapping Implementation

[12]The user-defined variables are available on lines 33-35 of Appendix I

Code Block 4 illustrates the functionality of the PEN code mapping in pseudocode so it's functionality can be better interpreted. This implementation is basic and serves to generate the unconstrained torque request based on the maximum configured torque limits and competition regulations. The scaledTorqueCommand output is later used in other code to act as the user input for power limiting.

```
void sendTorque(void) {
  .
  Calculate Motor Speed in Km/h
  .
  if Pedal above Zero Torque
      Generate Forward Torque Command
  else if Pedal below Zero Torque and no Trail Braking and over 5km/h
      Generate Regenerative Braking Torque Command
  else
      Zero Torque Command
  .
}
```

Code Block 4: PEN Throttle Mapping Pseudocode

6.6. Power Limiting

The FSAE-A competition stipulates that the maxmimum power draw from the accumulator must not exceed 80kW at any time during dynamic events or they will risk disqualification for that event. This is monitored through the use of the supplied Energy-Meter. This works by having a voltage reference for the pack, as well as a current sensor around the positive high-voltage cable to the accumulator. A simple V = PI calculation is then performed to determine the instantaneous power being delivered by the accumulator.

```
#define MAX_POWER 80000
// Sets the controllers max achievable power based on pack voltage
float efficiency = 0.9;
.
void sendTorque(void) {
  .
  // Convert motor speed from RPM to rad/s
  motorSpeedRadPerSec = motorSpeedRPM * 2.0 * PI / 60.0;
  if (motorSpeedRadPerSec > 0) { // Only care about forward rotation
    maxTorque = MAX_POWER*efficiency/motorSpeedRadPerSec; // Torque = Power/speed
    if (scaledTorqueCommand > maxTorque) { // Maximum power torque reduction
      commandedTorqueLimit = (maxTorque) * 10; // x10 for inverter expected value
    }
  }
  if (inverterRTD == 0) {
    commandedTorqueLimit = 0;
  } else if (commandedTorqueLimit > MAX_MOTOR_TORQUE*10) {
    commandedTorqueLimit = MAX_MOTOR_TORQUE * 10;
  }
  .
}
```

Code Block 5: PEN Power Limiting functionality

Code Block 5 shows the commandedTorqueLimit variable being dynamically calculated based on the programmed 80kW power limit. This is to ensure that if the scaledTorqueCommand would draw more than this limit, multiplied by the efficiency factor, the torque output of the motor will be saturated to ensure FSAE-A rules compliance. An efficiency variable is also included to ensure a margin of error, given the uncertainty involved with using inductance-based current sensors. It is also key to remember this functionality is limiting the power draw for the tractive-system alone, and as such the efficiency scaler contains headroom to run all of NU24's LV systems being powered by the DCDC.

```
Define maximum power
Define efficiency value
.
void sendTorque(void) {
  .
  Calculate Motor Speed in rad/s
  if Vehicle moving Forward
    Calculate Max Power Torque based on Power Limit
    if Requested Torque greater than Max Power Torque
      Use Max Power Torque instead
  .
  if Car is out of Ready-to-Drive
    Set Torque Limit message to 0
  else if Max Power Torque greater than Max Motor Torque
    Use Max Motor Torque instead
  .
}
```

Code Block 6: PEN Power Limiting Pseudocode

The pseudocode for power limiting has been provided in Code Block **??**. It should be noted that this follows Code Block 3 in sequence order for the sendTorque function within the PEN code. This can be viewed in its entirety in Appendix I.

6.7. Validation

An important part of the engineering process is to validate the models being developed by analysing the results and comparing them to the theoretical outcomes. The regenerative braking and current limiting were both validated during testing to ensure they would work as expected during competition. This involved testing the dynamic state of charge based current limit, the effectiveness of current limiting under regenerative braking, operation of power limiting and the increased efficiency of regenerative braking. These were done in the weeks leading up to competition by creating a scenario in which the condition would be forced.

For the dynamic state of charge current limiting, it was decided to charge the accumulator to a high state of charge and monitor the CAN channel which broadcasts the charge current limit to the inverter. Figure 38 shows the state of charge decreasing and as it does, the charge current limit begins to increase up to 30 amps. This functionality was performing as expected and no further modifications were deemed necessary as it was tested for the first time just 2 days from competition.

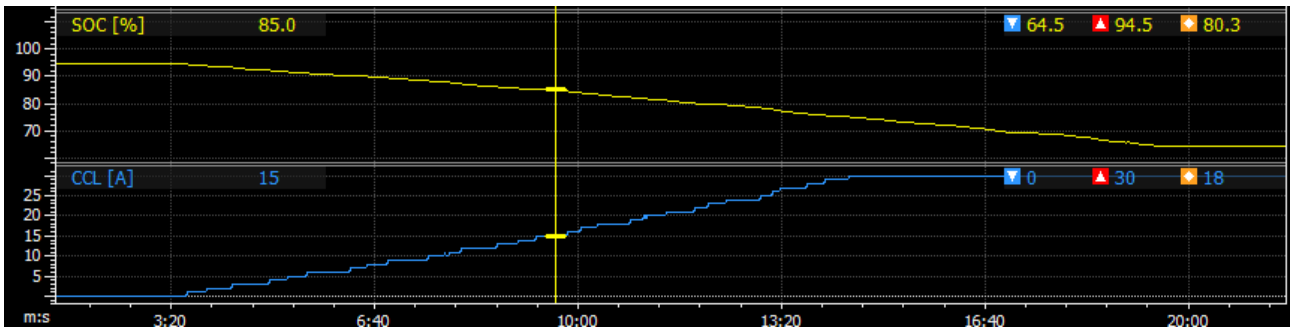


Figure 38: State of Charge based Charge Current Limiting

The regenerative braking current limiting was validated within the same testing session as data was gathered on the requested torque command as well as the inverter commanded torque. This can be seen on the MoTeC data in Figure 39 as the current is limited below 15 amps, and the commanded torque deviates from the requested torque command in order to satisfy the dynamic current limit. This deviation decreases as the vehicles begins to slow down due to the inverse proportionality the torque limit has to motor speed as shown in Equation 1.

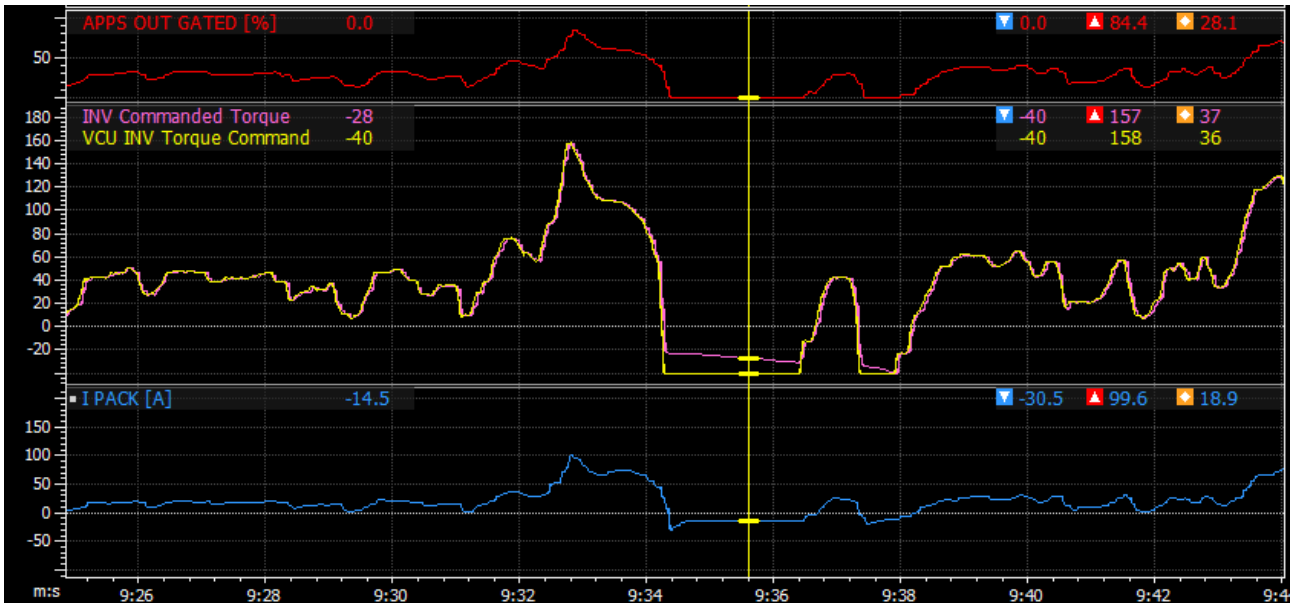


Figure 39: State of Charge based Charge Current Limiting

There is a very slight overshoot at the instant that current limiting is engaged, however due to the recommendations from the VTC6 manufacturer, this is safe given the short period of time for which the current limit is exceeded. This image also demonstrates the effect of the single pedal regenerative braking torque request. The torque command clearly follows the accelerator pedal position, shown in red, creating an intuitive driving experience for the drivers. This enables them to focus on driving line, cornering and composure during an endurance drive.

The power limiting code is implemented in the event that, for a given torque request and motor speed, the resulting power would be greater than the 80kW allowable by FSAE-A rules. This would be most likely to occur when at high voltages, with a large torque request and high motor speeds. Due to the properties of the lithium cells, the internal resistance results in a voltage drop over the pack due to high current flow. This can be calculated given the data in Figure 40 and Ohm's Law.

$$\begin{aligned} R &= \frac{\Delta V}{I} \\ &= \frac{447 - 337.6}{189.8} = 0.576\,\Omega \end{aligned} \tag{4}$$

This gives the net resistance of the 108S6P configuration of Sony VTC6 modules. Taking the average for the configuration yields a 5.3mΩ internal resistance per cell module. This is higher than the 2.8mΩ manufacturer initial impedance, however is well within safe limits for lithium-ion cells.

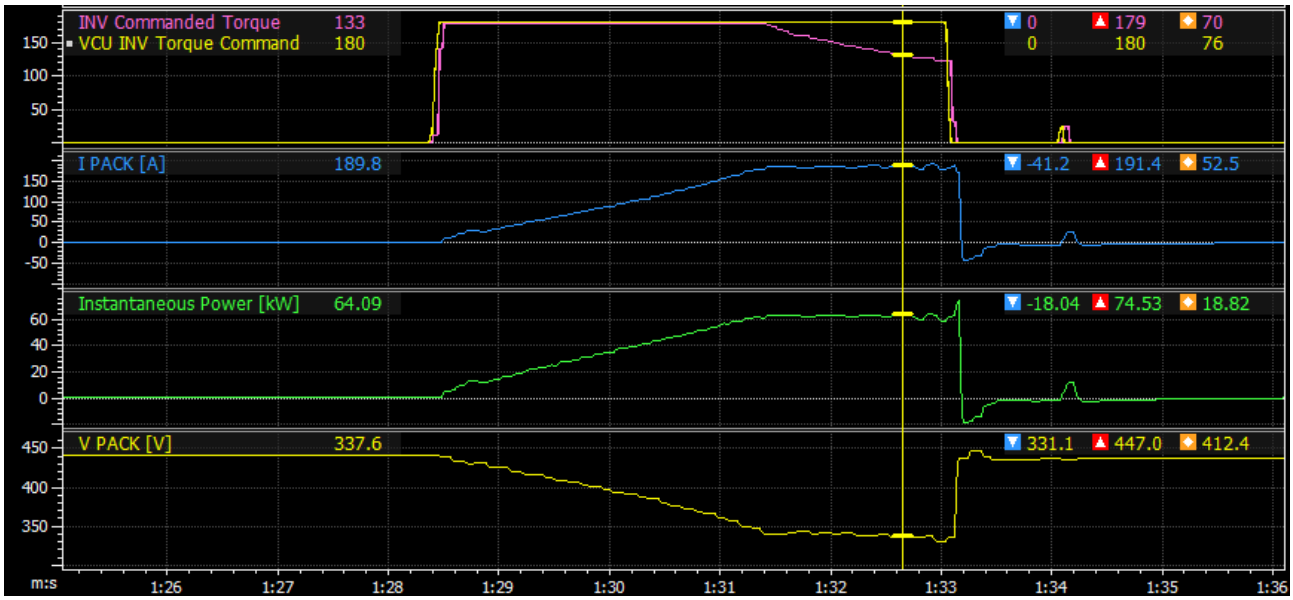


Figure 40: MoTeC Data showing power limit unable to be reached.

The figure also shows that the power limit is not reachable due to the 180 amp current limit and resulting voltage sag. The ideal power able to be delivered by the cells is 81.65kW, however due to the voltage sag experienced as a result of the internal cell resistance, the actual maximum power is 62.99kW. Figure 40 shows power greater than this due to the current limit being exceeded.

$$\begin{aligned} P_{\text{Ideal}} &= V \times I \\ &= 453.6 \times 180 \\ &= 81.65 \text{ kW} \end{aligned} \tag{5}$$

$$\begin{aligned} P_{\text{Real}} &= (V - \Delta V) \times I \\ &= (V - I \times R) \times I \\ &= (453.6 - 180 \times 0.576) \times 180 \\ &= 62.99 \text{ kW} \end{aligned} \tag{6}$$

A simulated endurance was conducted at Sydney Motorsport Park to gather data regarding the efficiency of NU24 before competition. This involved conducting a run with regenerative braking enabled and recording data on the power usage, power regenerated and power discharged. Figure 41 shows the MoTeC data gathered during the endurance run. The visible data shows the a small section of data which has valuable information on the following:

- **Peak Motoring Power** - 48.55kW
- **Peak Regenerative Power** - 20.46kW
- **Energy Discharged** - 1.86kWh
- **Energy Regenerated** - 0.46kWh
- **Net Energy Usage** - 1.40kWh

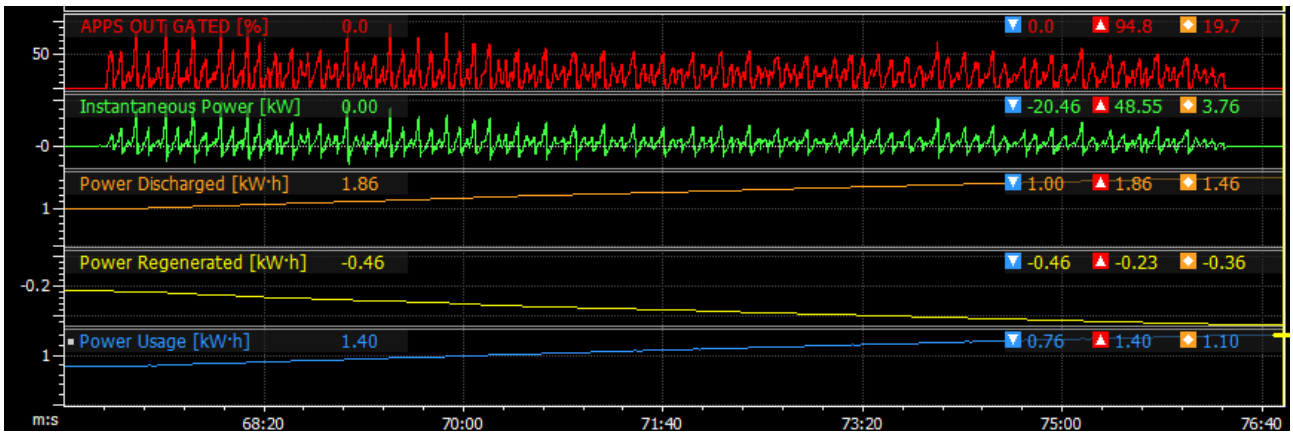


Figure 41: Regenerative Braking Energy Data

This dataset shows that 24.7% of all energy expended was to be recaptured through the use of regenerative braking. The battery pack was not adversely effected by heat through the additional current flow caused during regenerative braking. Figure 42 shows that with the addition of regenerative braking the temperature increase above ambient is only 2.4°C greater than when regenerative braking was not enabled. Comparatively, NU23's accumulator was much more thermally limited.

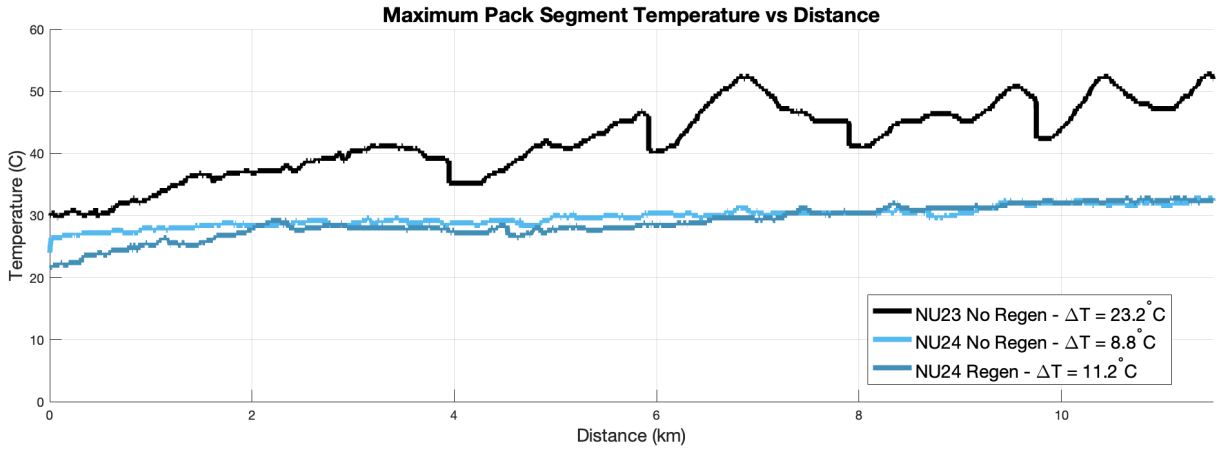


Figure 42: Regenerative Braking Temperature Data

7. Formula SAE-A Competition

7.1. Vehicle Reliability

The author worked with the mechatronics team to ensure the functionality and reliability of the EV Functional system. This involved ensuring that NU24 was compliant to the FSAE-A regulations while maintaining reliability. There were several issues of non-compliance that were solved throughout the year when not working on critical-path drivetrain commissioning.

One of the areas which had to be addressed was the ability to power cycle the tractive system and be able to re-enter ready-to-drive. When the driver would press their dash e-stop, upon returning the e-stop to its enabled position the car would enter ready-to-drive but the driver would have no accelerator pedal response. This was a crucial issue to fix as it could cause the car to become stuck on track and as the rules state that the driver cannot power cycle low-voltage, there would be no way to resolve the issue from within the car. It was discovered that this was due to the inverter enable message still transmitting to the inverter when breaking the shut-down circuit and disabling tractive system voltage. This caused the inverter to enter a fault state which required low-voltage power cycling to clear. To resolve this issue, the author developed new PEN code that would compute the inverter enable message through a ready-to-drive state machine.

```
void updateRTDStateMachine() {
    switch (rtdStateMachine) {
        case STATE_INIT:
          inverterRTD = 0;
          inverterEnable = 0;
          if ((int)rtdState == 1 && (int)sdTSMS == 1) {
              stateChangeTime = millis();  // Set the timestamp when rtdState becomes 1
              rtdStateMachine = STATE_RTD_WAIT;
          }
          break;
        case STATE_RTD_WAIT:
          inverterEnable = 1;
          // Wait until 3 seconds have passed since rtdState was set to 1
          if (((int)rtdState == 1 && (int)sdTSMS == 1) && (millis() - stateChangeTime >=
              3000)) { // Perform the logic after 3 seconds
            rtdStateMachine = STATE_RTD;  // Move to RTD state
          }
          // Reset if rtdState is no longer 1
          else if ((int)rtdState != 1 || (int)sdTSMS != 1) {
            rtdStateMachine = STATE_INIT;
          }
          break;
        case STATE_RTD:
          inverterRTD = 1;
          if ((int)rtdState != 1 || (int)sdTSMS != 1) {
            rtdStateMachine = STATE_INIT; // Go back to the initial state
          }
          break;
    }
}
```

Code Block 7: Ready-to-Drive State Machine Code

Code Block 7 shows the functionality illustrated by the flow-chat in Figure 43. This causes the inverter enable message to be set to disable whenever there is a break in the shutdown circuit. This works due to the shutdown signal at the tractive system master switch is dependent on the rest of the shutdown circuit. With the new PEN code, the vehicle was able to successfully be disabled by all switches and e-stops on the shutdown circuit and when switched back on would enter ready to drive with a torque response from the accelerator pedal.

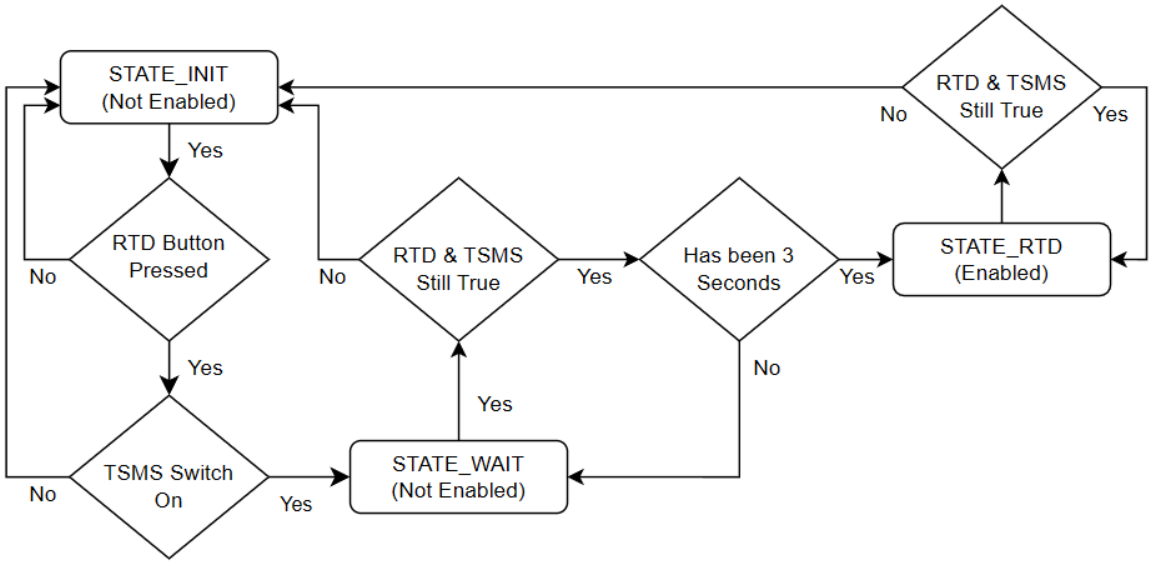


Figure 43: Ready-to-Drive State Machine Flowchart

Additionally, it was noticed that under certain circumstances the car would lose response from the accelerator pedal when initially put into ready to drive. Upon connecting to the inverter it was noticed that a CAN time-out fault was ocurring. This was due to the Teensy in the PEN taking long enough to boot that it would miss the allocated CAN time-out period defined by the inverter. By default this was 1000ms and it was decided to increase the value to 3000ms. Table 6 shows this value change, noting the naming convention of /3ms. This requires the user to divide their desired value by 3 and input the resulting number as the EEPROM value.

EEPROM Description	**NU23 Value**	**NU24 Value**
CAN_TimeOut_(/3ms)_EEPROM	333	1000

Table 6: Modification of CAN Time-out Duration on Motor Controller.

There was an issue with the circuit that addresses the accelerator pedal position plausibility check. The accelerator pedal has two analogue signal inputs to the PEN and as per the rules, both signals must be checked to be within 10% of the other. Typically this just requires a scale and offset in the PEN code as the two signals are linear. It was noticed with Figure 44 that the signal for TPS1 was non-linear. This meant that no matter the scale and offset, it was not possible to remain under 10% difference for the entire pedal range.

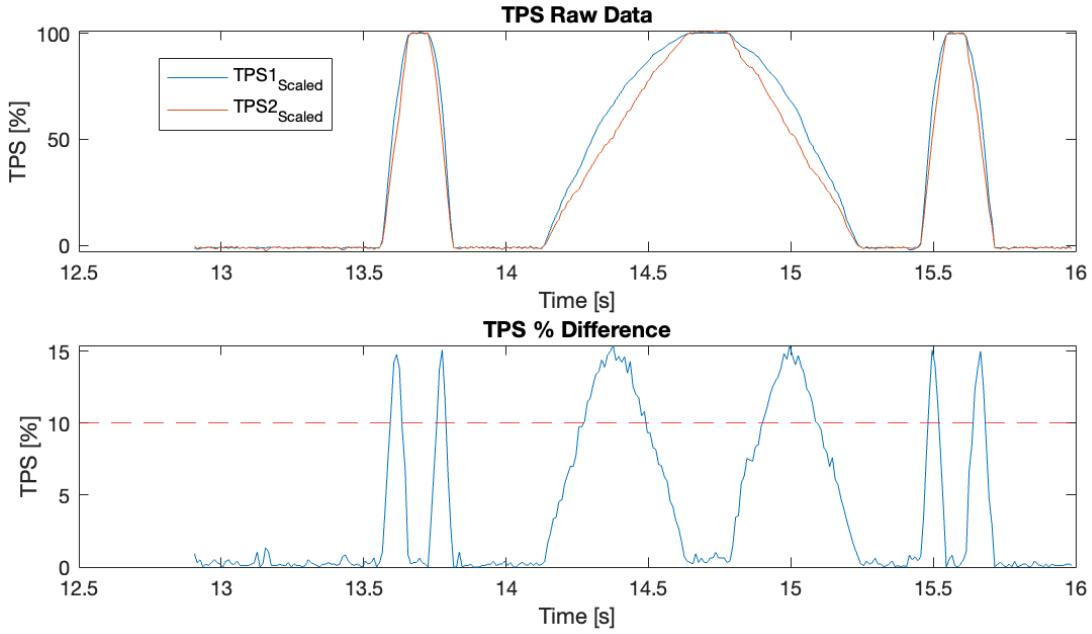


Figure 44: Accelerator Pedal Position Sensor Plausibility troubleshooting.

It was discovered that a zener diode, which is responsible for limiting the input voltage to safe levels for the Teensy, was causing the signal to be altered in a non-linear fashion. After removing this component the data was logged again to produce the graphs in Figure 45. This fixed the compliance issue by reducing the maximum percentage difference to below 2% where the rules specify below 10%.

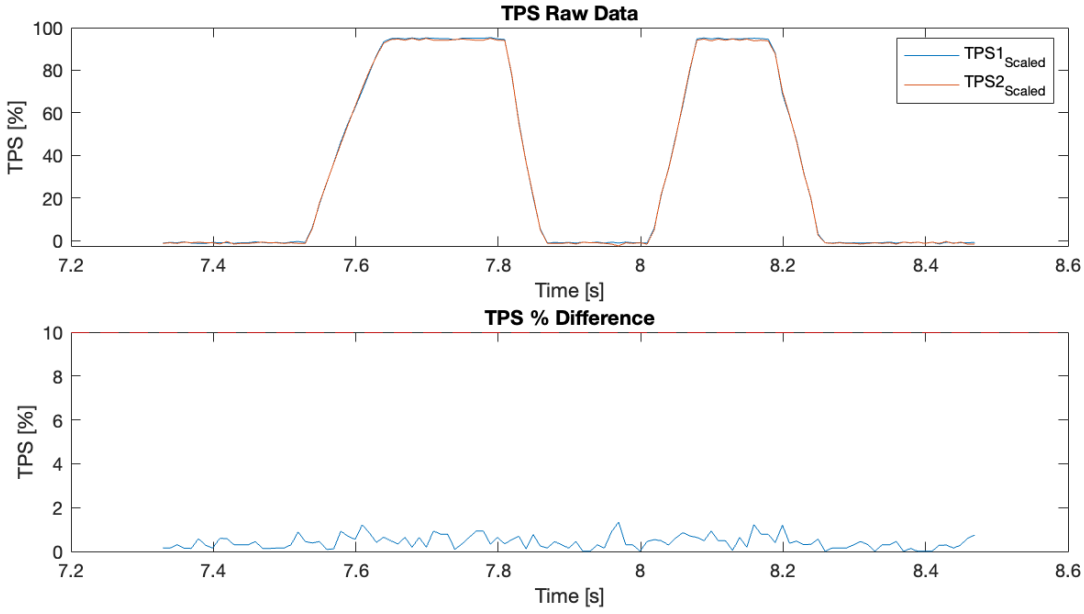


Figure 45: Accelerator Pedal Position Sensor Plausibility solution.

7.2. Technical Inspection

In order to compete in the FSAE-A competition, all teams must complete a set of rigorous technical inspections to ensure compliance to the regulations. These technical inspections are scheduled during the first two days of competition, where each team must pass the inspections, some of which require others to already have been passed. The criteria for each technical inspection are available prior to the FSAE-A competition week and allows the teams to complete practice technical inspections and develop a routine and choreography. The list of technical inspections that NU24 had to undergo are:

1. **EV Accumulator Inspection** - Involves complete inspection of the accumulator mechatronics subsystem. It involves providing evidence that all components used are certified and rated for the tractive system max voltage.
2. **EV Static Inspection** - Involves ensuring all low-voltage systems are compliant, ensuring the use of approved materials, accumulator placement and correct high-voltage system practices.
3. **EV Mechanical Inspection** - Involves the inspection of all mechanical subsystems to ensure compliance with the FSAE guidelines. It requires evidence of calculations made to ensure all components are rated for their expected loads.
4. **EV Functional Inspection** - Involves testing of all functional systems within the EV, with a close inspection of tractive system safety devices. A full inspection of the shutdown, precharge, discharge and hard fault latch systems must be conducted with reference to provided values.
5. **EV Rollover Test** - Involves placing the car on a ramp where it is strapped down and subjected to a high roll angle to ensure it meets the regulations surrounding rollover. This is completed on both sides with the tallest driver suited in the vehicle.
6. **EV Rain Test** - Involves a comprehensive rain test where the vehicle is subject to a mist of water spray for 90 seconds before resting for an additional 90 seconds whilst remaining in its ready to drive state. This ensures all electrical components have sufficient protection from water ingress.
7. **EV Braking Test** - Involves conducting a braking test where the vehicle must be able to lock all four wheels from a rolling start without diverging from a straight line.

The author was heavily involved in both the EV Static and EV Functional technical inspections. This involved being apart of several staged 'mock-tech' inspections where the author, Alec Chapman and Jacob Lukes would go through the inspection handbook and complete the line items within the allocated one hour period.

EV Static Inspection

Part of the EV Static Inspection requires the demonstration of appropriate high-voltage and grounded-low-voltage separation. It was the responsibility of the author to create a 'HV-GLV Seperation' document to provide the inspectors at competition. This document contained schematic and PCB views of all high-voltage circuit boards, aswell as datasheets to provide evidence on galvanic isolation for components that bridge both high-voltage and low-voltage circuitry. Additionally, the author served as an additional hand to assist Alec Chapman and Jacob Lukes during their demonstrations of other low-voltage and high-voltage systems.

EV Functional Inspection

The author was responsible for demonstrating the functionality of several sub-systems during the EV Functional inspection. The Brake Over-Travel Switch was demonstrated by showing that when the switch was moved to its fault position the shut-down circuitry would open, causing the car to shut down. All emergency stop switches were also demonstrated to be functional and within the rules definitions for breaking the shut-down circuit.

To demonstrate the tractive system safety requirements an additional variation of the PEN code was written for the purpose of EV Functional demonstration. This modified code would limit the torque command to only 20Nm forwards, whilst regenerative braking was disabled. This meant that at no point would the author or other attendees be at risk of the motors rapidly accelerating due to a small movement in the accelerator pedal. The Accelerator Pedal Position Sensor Plausibility was then demonstrated by depressing the accelerator pedal and unplugging each of the two sensors independently to show the vehicle would not send a torque command if there was greater than 10% variance in the signals. The pre-charge and discharge systems were demonstrated by Jacob Lukes, after which the author assisted in demonstrating the Braking System Plausibility Device. This involves holding the brake pedal whilst Jacob Lukes simulated a 5kW load through the tractive-system testing points which caused the car to shut-down. Additionally the trail braking safety system was demonstrated by carefully coordinating a sequence of events which involved:

1. Slightly depress the accelerator pedal until it is greater than 25% and the wheels are spinning.
2. While maintaining accelerator pedal position, quickly depress the braking pedal until the torque command stops.
3. Release the brake pedal while maintaining the accelerator pedal position and show there there is still no torque command.
4. Staying above 0% accelerator position, move the accelerator pedal through a range of positions to show there is no torque command through the pedal range.
5. Release the accelerator pedal back to its 0% position.
6. Depress the accelerator pedal to demonstrate the the torque command returns after fully releasing the pedal back to 0%.

After minor problems, such as the requirement to ground any hose-clamps within 100mm of tractive system voltage, the team was successfully able to pass all of its technical inspections and begin preparation for the dynamic competition events.

7.3. Design Event

The author was a part of the EV Tractive team in preparation for the FSAE-A design event. The design event involves students demonstrating to judges how decisions they made throughout the development process have been approached using engineering principles. NU Racing prepares booklets for the judges to follow while the students walk them through the relevant sub-system they are presenting. The EV Tractive system involved a team made up of Daniel Iveson, Kristopher Kerr, Jayden Hardinge and the author. The EV Tractive system booklet can be found in its entirety in Appendix K.

As the powertrain engineer, it was the authors responsibility to talk about the differences between NU23 and NU24 including the new powerbox, EMRAX 228 motor and regenerative braking. Krisopher Kerr was responsible with talking about the accumulator mechanical design including the new

aluminium container. Daniel Iveson was responsible for presenting the accumulator electrical design including the top plate, pre-charge and low-voltage distribution PCB. Jayden Hardinge was responsible for talking about the cooling system including design and validation for a new radiator and water pumps.

As part of the powertrain presentation, an acceleration performance simulation was conducted to illustrate the performance gains of NU24 over its predecessor. Figure 46 shows the simulated performance gain of the EMRAX 228, the modified gear ratio and overall 20kg weight decrease. The simulation was modelled using point-mass acceleration dynamics as well as rolling resistance and air resistance. This data was fitted to match the acceleration performance of NU23 during the FSAE-A competition in the previous year. With these calculations it was expected that NU24 would perform 0.18s faster due to these upgrades. The simulation included the limited torque due to powertrain stress, and it was estimated that with the full capability of the motor, the simulated time would decrease down to 4.00s.

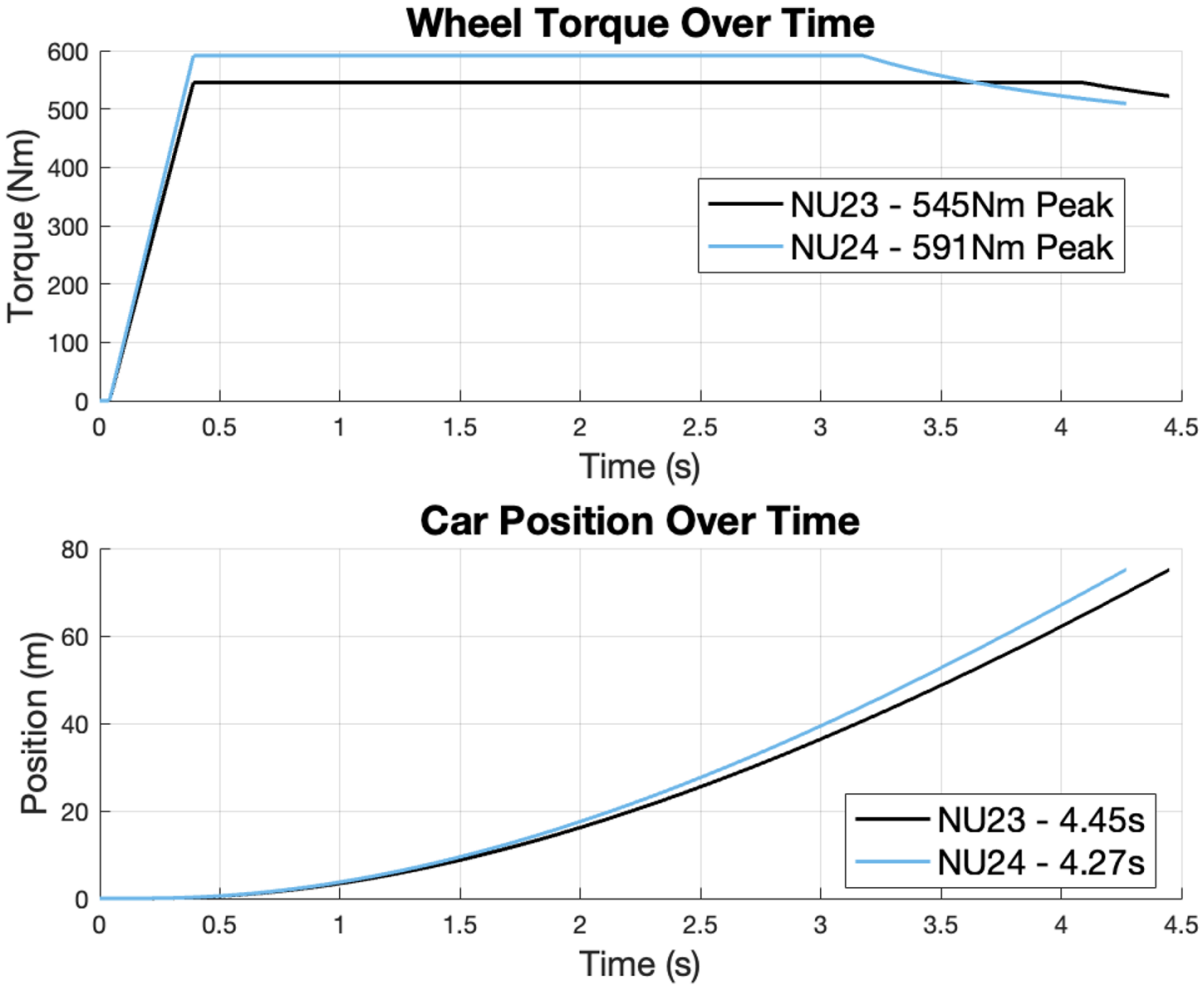


Figure 46: Matlab Acceleration Simulation between NU23 and NU24.

To demonstrate the effectiveness of the regenerative braking system, endurance data was presented showing the decreased energy usage. It can be seen in Figure 47 that NU23 used 2.83kWh during half of a complete endurance drive.[13] NU24, having less mechanical losses due to the lack of a chain tensioner, and a better thermally managed accumulator design, is able to achieve an energy usage of just 2.28kWh. Significantly, the NU24 endurance data with regenerative braking enabled demonstrates an energy usage of only 1.71kWh while maintaining the same faster pace as when it was disabled. Both NU24 times were able to use less energy while completing the 11km distance in two and a half minutes less time. This data would suggest that the powertrain improvements will lead to better results in both endurance and efficiency during competition.

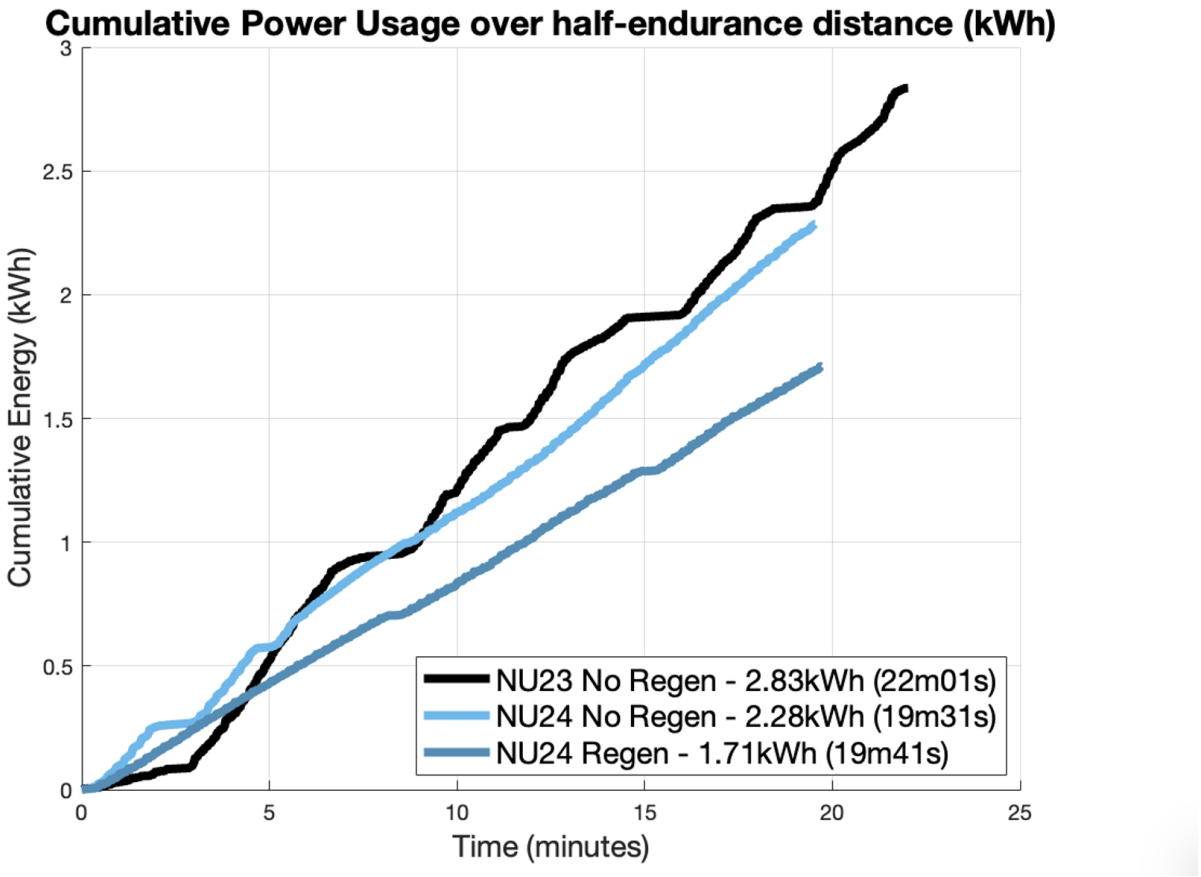


Figure 47: Energy Regenerated during Endurance testing for Design Report.

During the design event, the author also presented their multivariate, dynamic current-limiting functionality. Feedback from the judges suggested that more validation must be done on a component basis to ensure the results expected align with real world performance. It was also stated that the development of a lap-time simulation tool would help in justifying decisions made during the design phase. Overall the team scored 120 out of the available 150 points, with EV Tractive system receiving a points total of 11.4 out of 30.

[13]This data was taken from the FSAE-A 2023 competition where NU23 did not complete the second driver stint.

7.4. Acceleration Event

The FSAE acceleration event requires teams to stage their car and perform a standing start 75-meter timed acceleration run. Managing wheel slip, vehicle dynamics and torque output enables teams to compete for the fastest time to cover the 75 meter distance. Due to the increase in power and decrease in weight it was expected that NU24 would improve on the time of 4.45s, set by NU23 in the previous year of competition. Based on simulation data presented in Section 7.3, it was estimated for NU24 to achieve a time of 4.27s.

Upon completing the first of four acceleration runs it was noticed that there was no wheel-spin occurring, likely due to the warm conditions and track surface. This resulted in NU24 achieving a time of 3.94s during its third attempt. This far exceeded the expectations of the team, setting the fastest time in the history of NU Racing. It is likely that this time is a result of how the timing beacon is set up, with the time starting as the rear of the vehicle crosses the starting line. This would result in the time starting when the vehicle has an initial velocity, causing a faster time than the simulated data.

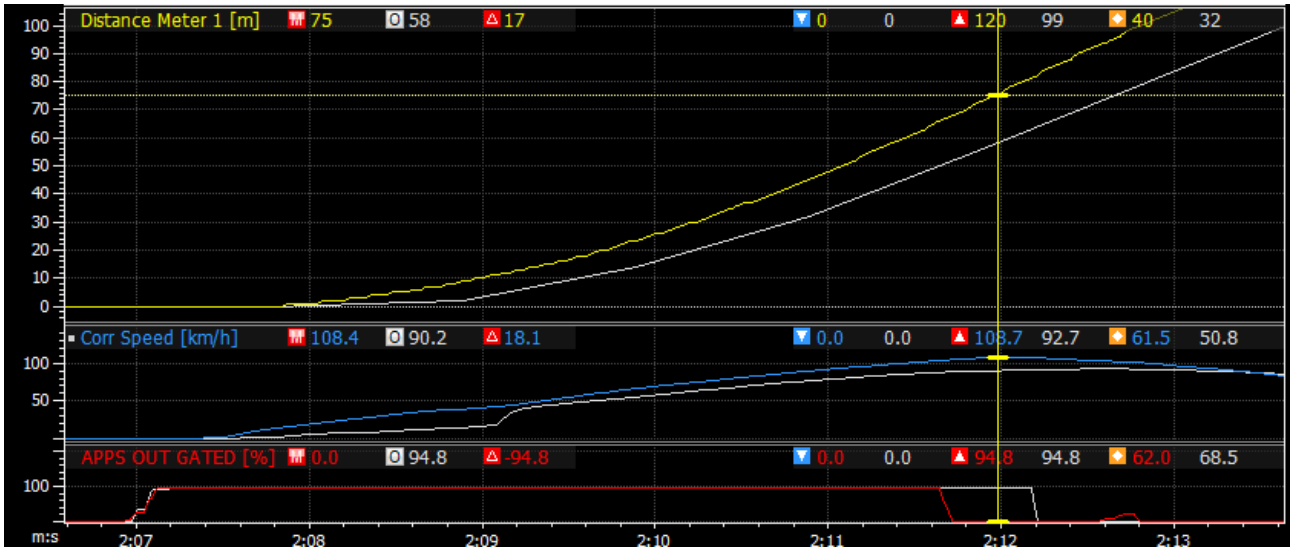


Figure 48: FSAE-A 2024 Acceleration MoTeC Data.

When evaluating the MoTeC data as observed in Figure 48, it can be seen that NU24 (represented by the coloured data) is able to cover the 75-meter distance much faster than NU23. It is also evident that NU24 reaches a significantly higher top speed at the end of the run, crossing the line at 108.7km/h compared to 92.7km/h achieved by NU23. All of this is made possible through the additional torque available to drive the lighter, more agile 2024 competition vehicle.

The time scored by NU Racing placed them 4th, scoring 71.79 out of the available 100 points. It should be noted that this was not expected to be a strong event for NU24, as the other top performing teams all high much more advanced tractive systems. For example, the University of Auckland, who won the event with a time of 3.551s implements dynamic torque vectoring with four hub-motors to ensure all wheels are managing an optimum slip angle during the entire run.

7.5. Autocross Event

The autocross event started by Timothy Kerr setting a 'banker lap' by ensuring that the team had a time on the board before beginning to push for a faster more competitive time. The first run was successful in setting a time and the run that followed it was intended to push the limits of the car and extract every bit of performance available. During the second running the vehicle there was a fault reported by the BMS causing the car to stop on track with no way of being retrieved.

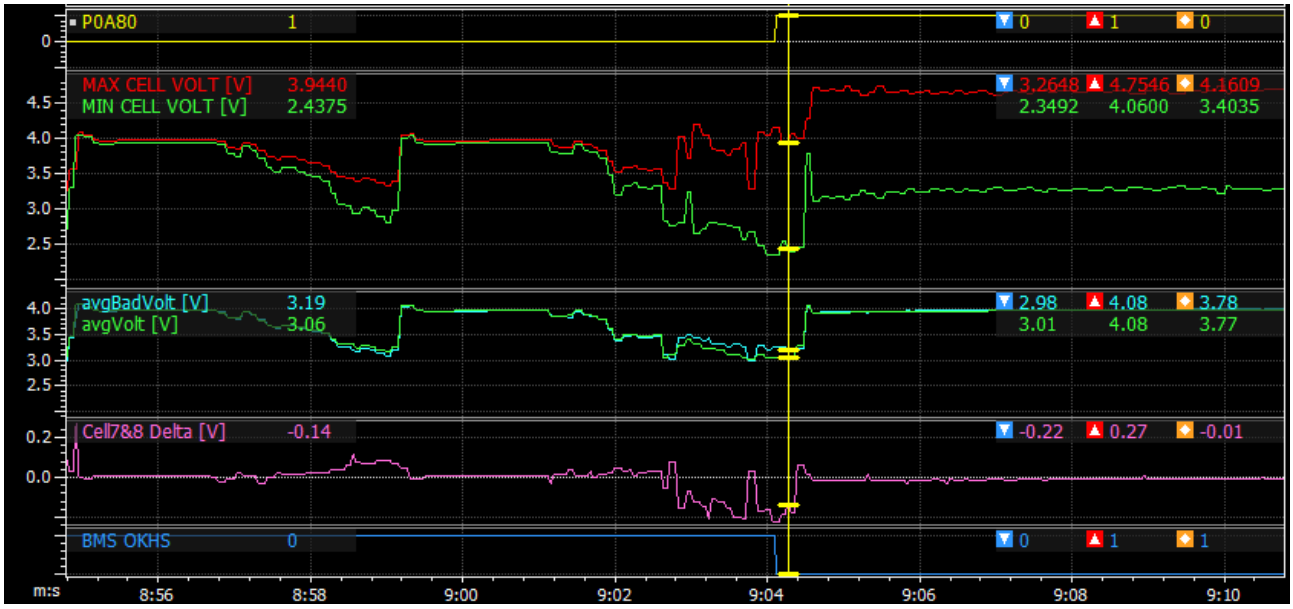


Figure 49: FSAE-A 2024 Autocross BMS Fault MoTeC Data.

The MoTeC data in Figure 49 shows the moment that the BMS fault occurred during the autocross event. The fault code P0A80 shown refers to the Weak Cell Fault defined by the Orion BMS as the "Battery cell determined to be abnormally weak, underperforming, out of balance or low in capacity."[12] This was an issue that had intermittently appeared through 2024, with cell modules 7 and 8 being reported as 'weak cells'. After several replacement modules it was determined that the voltage taps connected to these cells had a poor connection or breakdown in the solder joints.

It can be seen that the minimum cell voltage and maximum cell voltage deviate by greater than 1.5V at the time of the fault. This is a requirement of the FSAE competition and comes as a result of the BMS Okay-High-Signal. When analysing the rest of the pack however, it was noticed that when taking the sum of the two weak performing cells, it was equal to the expected sum of any two healthy cells. It was then decided to plot the average of the two bad cells, as indicated by avgBadVolt, and the average of all cells in the pack, as indicated by avgVolt. By comparing these two values, the author was able to make an estimation as to the actual open-cell voltage breakdown in the weak cells. This difference was then monitored through the Cell7&8 Delta channel where it was evident that this pair of cells was only deviating by 0.3V in the worst scenarios. Whilst this is not ideal, it was within the rules requirements of the FSAE-A competition and meant that the team could continue to run the vehicle. There was no way to implement this maths into the BMS however, so another solution had to be created to safely compete in the following endurance event.

7.6. Endurance Event

In order to compete in the endurance event, the author had to develop a solution to the BMS fault that had occurred during the autocross event. It is important to note that these two events run in order and there was only a two-hour window before the endurance event begun. Since the cells were deemed to be healthy it was decided that the BMS analogue signal for the OKHS would be disabled and the author would remotely monitor the MoTeC data window shown in Figure 49. NU24 makes use of a serial transmitter which allows the end-user to see live telemetry during the operation from the car while being a distance away.

The decision was made by the author to maintain audio contact with the driver over voice-call while monitoring the MoTeC data stream. If any conditions were met that would mean the vehicle would have triggered the hard fault latch as per the FSAE-A ruling, the author would signal to the driver to shut down the vehicle and terminate the endurance run. This was done to remain rules compliant in the eyes of the team and allow for fair competition. In order to disable the OKHS functionality, the author connected to the Orion BMS2 utility and navigated to the Multi-Purpose Output settings as seen in Figure 50. Multi-Purpose Output 1 & 2 were both disabled as these affect the analogue BMS OKHS signal. It was made sure not to change any of the CAN functionality as these signals still required to be monitored.

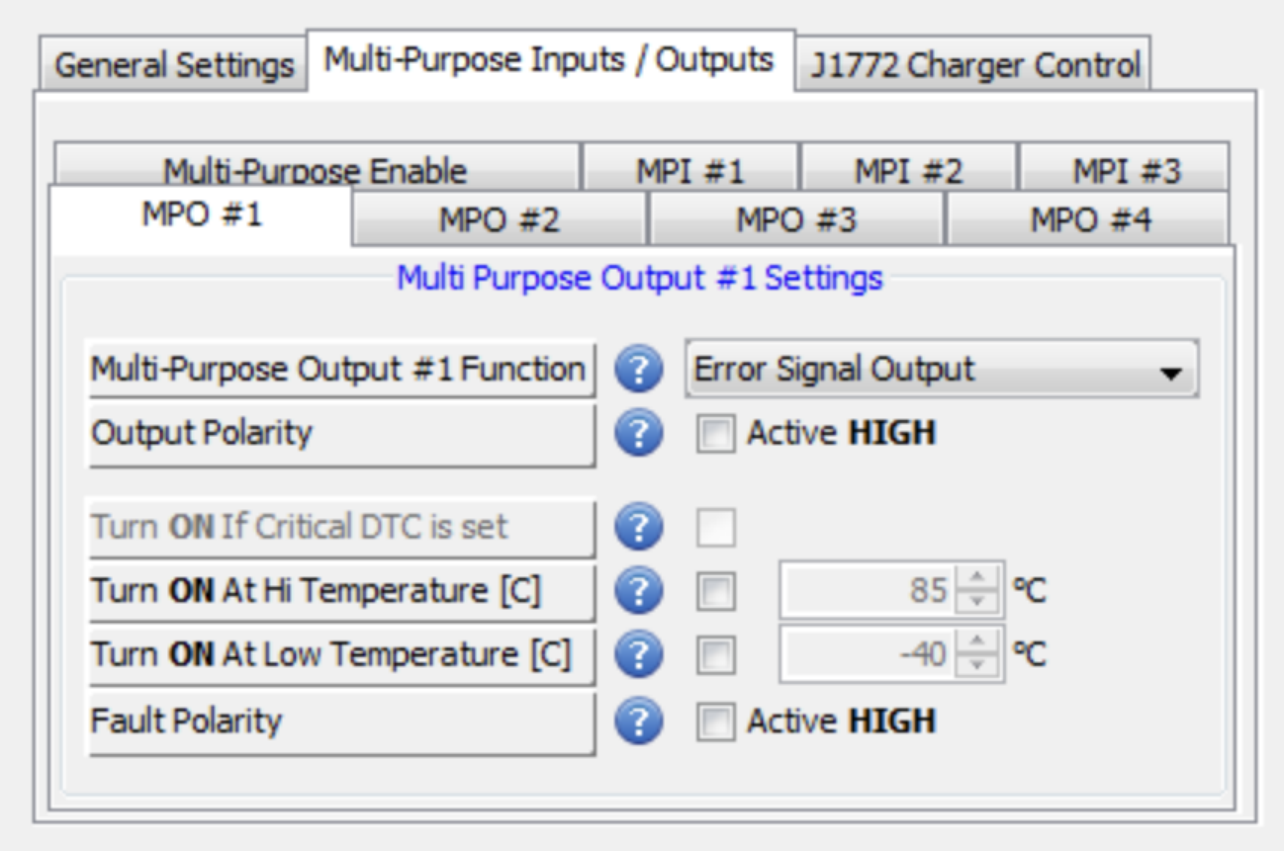


Figure 50: Orion BMS2 Multi-Purpose Output configuration.

The car was sent out for the endurance event and was able to complete all 22km, including the driver swap, with no indication of a cell fault. The regenerative braking deployment strategy was successful, with little regenerative braking current initially, and the current limit increasing as the state of charge begun to decrease. The team scored a high efficiency rating for NU24 whilst also performing well in the endurance timing event. It can be seen in Figure 51 that over the 22km distance, the maximum discharge power was 65.58kW whilst the maximum charge power from regeneration was 25.68kW. The total power usage of the run was 3.72kWh, of which 4.64kWh was used for driving whilst 0.92kWh was regenerated under braking.

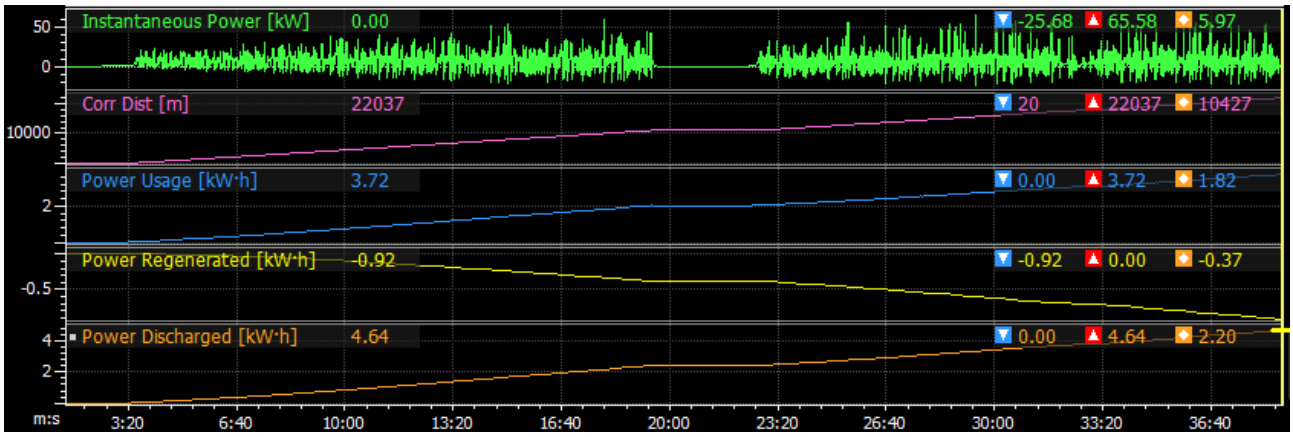


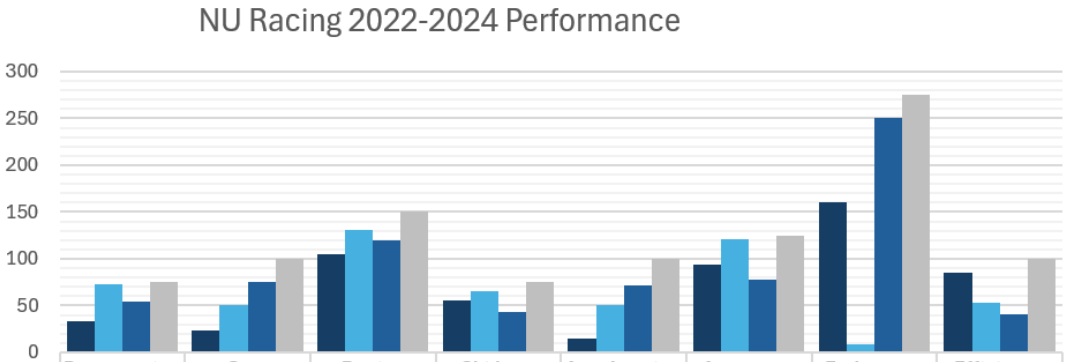
Figure 51: FSAE-A 2024 Endurance MoTeC Data.

This result demonstrated the immediate points gain as a result of recapturing kinetic energy, with the team scoring fourth place in efficiency with a score of 40.82 out of a possible 100. The team also performed strongly in the endurance classification, coming fifth with a score of 250.87 out of a possible 275. This was a very successful finish to the event, coming off the back of a difficult year where previously NU23 had failed to complete the endurance event.

7.7. Total Performance of NU24

The NU Racing 2024 performance at FSAE-A was full of many achievements, netting the team an overall points score of 733.86 out of a possible 1000 points. This result placed the team in fourth place overall, a best achievement in the history of NU Racing. The team was close to receiving several podium awards however fell slightly out of the top-three due to a very strong competition. All goals set by the author in terms of performance were met, with a direct influence on the points scored through the commissioning of the more powerful motor, addition of regenerative braking, and additional battery management.

The total points breakdown are shown in Figure 52 where the figures for the previous two years of NU Racing are also shown. The FSAE-A competition dynamic event points work by normalising the times/scores of all teams with reference to the best performing and worst performing team. It is due to this that the dynamic event points reflect the teams performance relative to the other teams which competed in the same event. For example, the efficiency score for both EV3 and NU23 are greater than NU24 as a result of the strong performance by the University of Canterbury at FSAE-A 2024. This strong result shifted the score of NU24 down due to the large difference in efficiency.



	Presentation	Cost	Design	Skidpan	Acceleration	Autocross	Endurance	Efficiency
2022 (EV3)	33.16	23.11	105	56.06	15.13	93.44	159.99	85.26
2023 (NU23)	72.55	51.06	130.33	65.77	50.92	120.99	9	52.52
2024 (NU24)	54.05	75.55	120	42.64	71.79	78.13	250.87	40.82
Total Points Available	75	100	150	75	100	125	275	100

Figure 52: FSAE-A Competition Results from 2022-2024

When comparing the static events, its can be seen that the 2024 team performed much better on the Cost event due to the work of Lachlan Fisher, Alec Chapman and the rest of the team. Design and Presentation scores were slightly worse than the very well performing 2023 teams. NU Racing scored similarly on the Design event compared to previous teams, scoring highly in the field. The Acceleration result was the best in NU Racing history and stands clear when compared to previous teams. The Autocross performance of NU24 was shadowed by the BMS fault, where the full potential of the vehicle was not able to be realised at FSAE-A. The Endurance score was particularly high due to the non-finish of the NU23 team. Overall the team managed to score 180.72 points greater than the previous year's effort, which indicates a great improvement on both car performance and reliability. There are improvements to be made as the team continues to develop and work on key areas through its many iterations.

8. Conclusion

Operating as the Powertrain Engineer for NU Racing in 2024 provided a wide range of experiences and insights into the intricacies of development on an electric vehicle powertrain. Important lessons were learned about following a procedural thought process, acting based off of evidence rather than feeling. Working as part of a competitive team within the FSAE-A program enabled growth into a mature role which developed skills important to working as an engineer. A central theme for the year was the transition from existing components to new components that had been chosen due to their marginal gains to further improve the performance of the vehicle. Upgrading to the EMRAX 228 initialised a step forward to pushing the boundaries of what is possible within the regulations of the FSAE-A competition. Where previous teams had opted for a motor which could not exceed the limitations, new challenges were discovered in harnessing as much of the performance as allowed. The substantial torque increase had to be managed to ensure the reliability and longevity of other surrounding components. The net result was a powertrain which provided more torque, at all speeds, while increasing the maximum achievable speed.

Significant time was spent integrating both hardware and software solutions to find a unique balance in performance, complexity and reliability. The malfunction of motor controller hardware introduced a severe disruption to scheduled activities and required a lengthy troubleshooting process. This has served as a learning tool to further develop diagnostic protocols and building a wealth of knowledge to be transferred to future editions of the NU Racing team. Addressing electromagnetic interference concerns and implementing better procedures to manage interference have resulted in a more robust system which will see an increase in reliability for future iterations of the NU Racing competition vehicle. Development of a methodical approach to commissioning new hardware including the calibration of sensitive equipment such as the resolver will reduce the complexity for future students working on the powertrain subsystem. A highlight in the importance for an attention to detail will echo throughout the team, reminding students to approach problems with reasoning and considering all available information.

Through iterative development, the team was able to implement many small changes to all aspects of the car and realise these changes in a net performance gain at the FSAE-A 2024 Melbourne competition. The team was able to secure an overall fourth place finish, while overcoming hurdles and setting achievable goals. The development followed a rigorous engineering process under which models were created, predicted and validated to ensure the theoretical approaches were having expected real-world effects. The result of this methodology was a vehicle which was able to perform competitively in all dynamic events during competition. The team was able to practice their engineering skills while working in a large group environment and developing keen social and real-world skills to be applied post-graduation.

It needs to be reinforced that the primary goal of NU Racing is to create better engineers to go out into the workforce and apply the lessons they have learned as a part of this project. If this is the criteria which is to be measured to define a successful year, it can be said that without a doubt all members of the team have grown into capable and effective engineers.

9. Recommendations for Future Work

Several projects had to be turned into recommendations as a result of the malfunctions and troubleshooting that had to take place during the second half of 2024. These are projects which the author has highlighted as beneficial to the team and suitable for the following powertrain engineer to investigate where possible.

The addition of a wide range of variables into the multivariate current limiting to ensure protection against all possible situations which may cause damage to the battery. Additional parameters include:

- Maximum and Minimum voltage on a per cell basis
- Maximum and Minimum temperatures on a per cell basis
- Internal resistance measurements and open cell voltage readings
- User-configurable limits, potentially to protect other tractive system components
- State of charge based discharge limits to prevent running the battery flat

Furthermore, additional variables may be passed into the torque limiting such as:

- Wheel slip angle using a front wheel speed sensor
- Yaw and Yaw-rate conditions
- Vehicle speed to avoid high current draw

Further tuning of the PID controls within the motor controller using the University dyno facility in EC basement. Tuning needs to be done to validate variable PWM implementation as well as fine tuning the torque-model PID values to avoid under/overshoot when the inverter is current limiting.

Investigation into closed loop launch control and traction control to further develop the acceleration performance. The primary way to initiate this is by adding a front wheel speed sensor to target a rear-wheel slip angle. The addition of a limited-slip differential will also help should the car begin to have traction issues in both straight lines and during cornering.

Further development and optimisation of PEN code as it transitioned into a major portion of the vehicles control unit. It is possible to implement all of these solutions through both hardware and software integration on the PEN. This board should be treated as inclusive to the powertrain engineer's work as it is the most impactful board on the car for making powertrain developments.

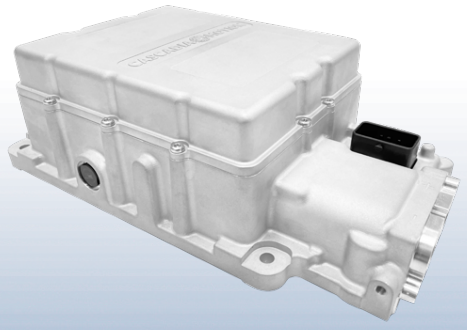
Further investigation should be made into the thermal efficiency of the accumulator, or the transition to pouch cells to enable a higher peak current discharge for the pack. This is the most limiting factor stopping the team from extracting more performance from the motor during high-speed high-torque situations.

Lastly, the deployment of driver adjustment regenerative braking strategies and strengths as well as variable peak torque through the use of steering wheel or dash dials. The groundwork for this has been completed and it needs to be integrated onto existing circuit boards through an iterative design process. The inclusion of a non-linear torque mapping within the PEN may aid in having different torque modes without increasing or decreasing the peak torque avaialble.

Appendix

A. CM200 Inverter Datasheet

CM200 INVERTER



Our newest inverter is the CM200, packing the punch of a PM150 but being smaller volume and lighter weight than a PM100. Also features HVIL, pluggable connectors and an EMI filter!

FEATURES

- 4 (0-5V) Analog Inputs
- 2 RTD inputs PT100/1000
- 4 Digital Inputs 4-STG
- 2 High Side Driver Outputs
- 2 Low Side Driver Outputs
- 1 Resolver Interface
- 1 Sin-Cos Encoder Interface (-SP Option)
- 2 CAN 2.0A/B Ports 125kb-1Mb adjustable rate and offset
- RS232 Programming and Diagnostic Connection
- Rosenberger Power Connectors
- Integrated DC-Link EMI Filter
- Designed to ISO16750 heavy vehicle climatic, mechanical, and environmental requirements
- ISO20653 high pressure wash rated IP6K9K / IP67
- Easy to use CAN-based network node
- Custom .dbc messaging
- J1939 compatible CAN messages available
- Extensive feedback broadcast messaging for datalogging
- PC-based setup and programming tools available free
- Robust, fault-tolerant IGBT power stage
- HVIL Interlock on connectors
- Command Safety Watchdog
- Variable PWM Rate

CM200	DX	DZ	Units
DC Voltage – operating	50-480	200-840	VDC
DC Overvoltage Trip	500	860	VDC
Maximum DC Voltage – non-operating	500	900	VDC
Motor Current Continuous	300	200	A
Motor Current Peak *	740	400	Arms
Output Power Peak (elect) *	225	225	kW
DC Bus Capacitance	650	255	µF
Size and Volume	330 x 188 x 97 / 3.9		mm / L
Weight	6.75		kg
Active Discharge via motor winding to <50V	< 1		sec
Passive Discharge (internal resistor) to <50V	< 120		sec
Vehicle System Power	9 .. 32 (12V & 24V systems)		VDC
Inverter PWM Frequency	12 (6 .. 16 variable)		kHz
Operating Temperature Range – coolant water	- 40 .. +80, (derate to zero 80 .. 100)		°C
Coolant Flow Rate	12 (3 GPM min)		LPM
Coolant Pressure Drop (60°C coolant / 12 LPM)	0.3 (30kPa / 4.3psi)		bar
Maximum Coolant Pressure (absolute)	3 (300kPa / 45psia)		bar
Operating Shock (ISO 16750-3, Test 4.2.2.2)	500 (50g), pending		m/s^2
Operating Vibration (ISO 16750-3, 4.1.2.7 Test VII)	57.9 (6grms), pending		m/s^2
EMC compatibility	IEC61000 / CISPR-25 pending		
Compatible Conductor Sizes	16, 25, 35, 50		mm^2

Ratings subject to change without notice—consult factory
* *Peak current is defined as a maximum of 30 seconds.*

Point Camera Here

To View Manuals

CASCADIA MOTION
cascadiamotion.com +1-503-344-5085

B. EMRAX 188 Datasheet

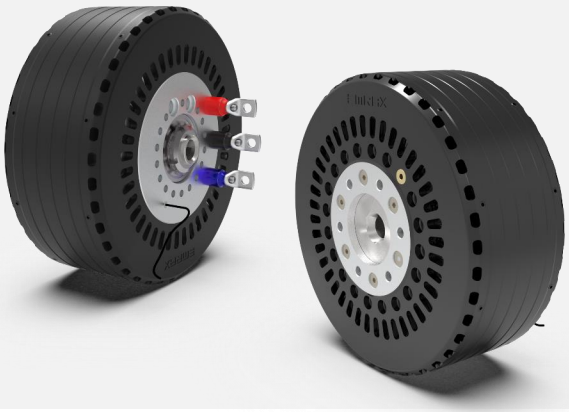
188

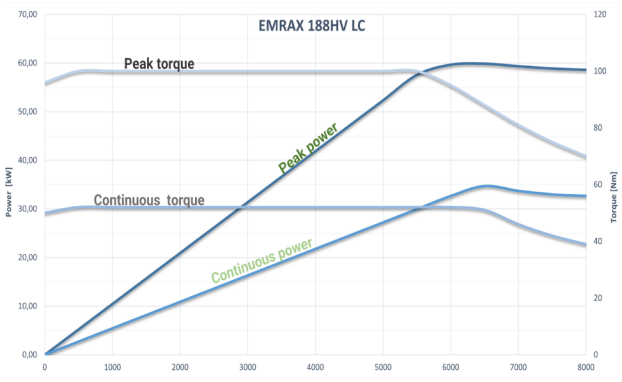
EMRAX
INNOVATIVE E-DRIVES

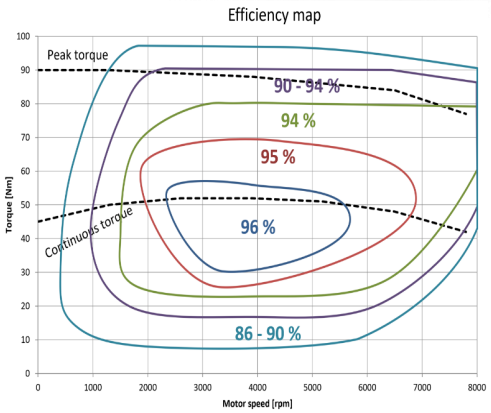
EMRAX 188 is a compact axial flux permanent magnet synchronous electric motor with high power/torque density.
Because of its low weight, it is ideal for VTOL, ultralight aviation, motorcycles, automotive and marine outboard applications. It has gained a favorable status among FSAE competitors.

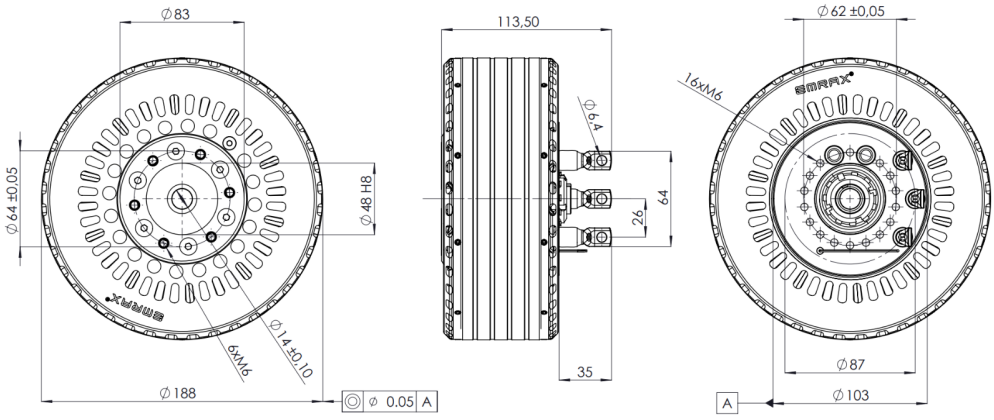
EMRAX 188	
DIAMETER | LENGTH	188 mm | 79 mm
WEIGHT	7,1-7,9 kg
COOLING	air / water / combined
PEAK | CONTINUOUS POWER	60 kW | 37 kW*
PEAK | CONTINUOUS TORQUE	100 Nm | 56 Nm*
MAXIMUM SPEED	8000 RPM
OPERATING VOLTAGE	50 - 660 V
EFFICIENY	up to 96%*
POSITION SENSOR	resolver / encoder

*Subject to motor configuration, drive cycle, thermal conditions, and controller capability.









Version 1.5

	EMRAX 188 High Voltage			EMRAX 188 Medium Voltage			EMRAX 188 Low Voltage		
AC = Air cooled LC = Liquid cooled CC = Combined cooled (Air + liquid)	AC	LC	CC	AC	LC	CC	AC	LC	CC
Ingress protection	IP21	IP66	IP21	IP21	IP66	IP21	IP21	IP66	IP21
Cooling specifications	ambient air 20ºC 20 m/s	min. 6 l/min, max. 50°C	AC+LC*	ambient air 20ºC 20 m/s	min. 6 l/min, max. 50°C	AC+LC*	ambient air 20ºC 20 m/s	min. 6 l/min, max. 50°C	AC+LC*
Maximum motor temperature [°C]	120								
Motor connection type	UVW or 2x UVW			UVW or 2x UVW			UVW or 2x UVW		
Voltage required for peak power [V_{DC}]**	660 Vdc			390 Vdc			160 Vdc		
Motor peak efficiency [%]	96%								
Peak power S2 2min [kW]	60 kW at 6500 RPM								
Continuous power S1 (kW)	27	34	37	27	34	37	27	34	37
Peak torque [Nm]	100								
Continuous torque [Nm]	40	52	56	40	52	56	40	52	56
Limiting speed [RPM]	8000								
K_V constant at no load [rpm/V_{DC}]	17,73			29,58			72,82		
K_V constant at nominal load [rpm/V_{DC}]	13,61			22,83			56,21		
K_V constant at peak load [rpm/V_{DC}]	9,81			16,61			40,87		
K_T constant [Nm/A_{RMS}]	0,54			0,32			0,13		
Peak motor current [A_{RMS}]	190			310			900		
Continuous motor current [A_{RMS}]	100			160			400		
Internal phase resistance at 25 °C [mΩ]***	14,37			5,04			1,02		
L_D induction of 1 phase [µH]	188,5			40,2			12,5		
Induced voltage [V_{RMS}/RPM]	0,04201			0,02521			0,01024		
Magnetic flux – axial [V_S]	0,03275			0,01965			0,00798		
Temperature sensor on the stator windings	KTY 81/210								
Number of pole pairs	10								
Winding configuration	star								
Rotor Inertia [kg*m²]	0,00989								
Bearing configuration	6205 | 3204								
Weight [kg]	7,1	7,9	7,6	7,1	7,9	7,6	7,1	7,9	7,6

*Combined cooled motor (CC) requires cooling specifications from air and liquid cooled motors, to reach its specifications. It cannot only be cooled as an air-cooled motor.
Every EMRAX motor requires sufficient air circulation. The motors should not be completely enclosed in any condition. Please check EMRAX motor manual to learn more.
Performance in your application will depend on your installation details and boundary conditions. Please contact us to learn more.
**All motors are tested for 833V maximum voltage.
***Measured Phase to Phase divided by 2.

Values given are for a standard 3 phase UVW version, please consult EMRAX on 2x UVW values. $R_{1UVW}=2*R_{2UVW}$

C. EMRAX 228 Datasheet

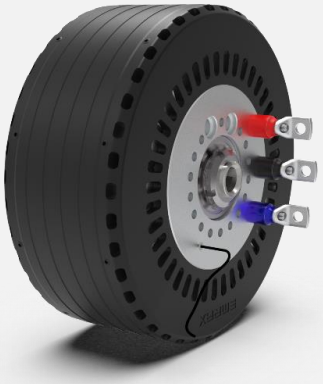
228

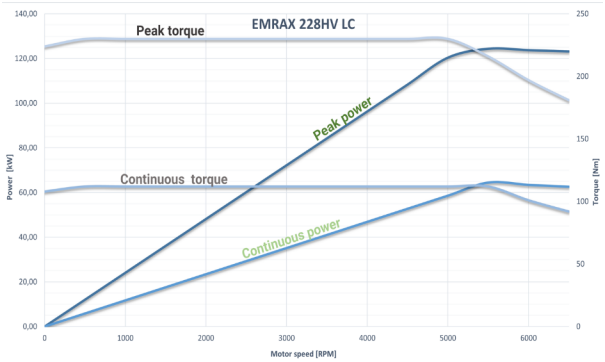
EMRAX
INNOVATIVE E-DRIVES

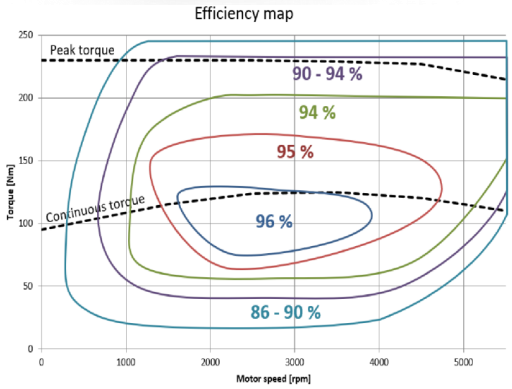
EMRAX 228 is a compact axial flux permanent magnet synchronous electric motor with high power/torque density.
It offers the middle of the range performance and is a great fit for where high power output in a small package is needed. Contact us to find out about its typical applications!

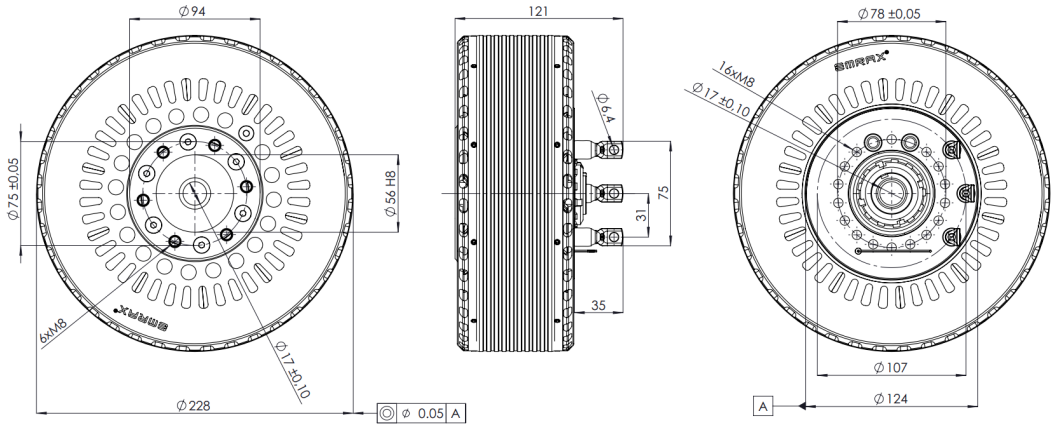
EMRAX 228	
DIAMETER | LENGTH	228 mm | 86 mm
WEIGHT	12,9-13,5 kg
COOLING	air / water / combined
PEAK | CONTINUOUS POWER	124 kW | 75 kW*
PEAK | CONTINUOUS TORQUE	230 Nm | 130 Nm*
MAXIMUM SPEED	6500 RPM
OPERATING VOLTAGE	50 - 830 V
EFFICIENY	up to 96%*
POSITION SENSOR	resolver / encoder

*Subject to motor configuration, drive cycle, thermal conditions, and controller capability.









Version 1.5

	EMRAX 228 High Voltage			EMRAX 228 Medium Voltage			EMRAX 228 Low Voltage		
AC = Air cooled LC = Liquid cooled CC = Combined cooled (Air + liquid)	AC	LC	CC	AC	LC	CC	AC	LC	CC
Ingress protection	IP21	IP66	IP21	IP21	IP66	IP21	IP21	IP66	IP21
Cooling specifications	ambient air 20ºC 20 m/s	min. 6 l/min, max. 50°C	AC+LC*	ambient air 20ºC 20 m/s	min. 6 l/min, max. 50°C	AC+LC*	ambient air 20ºC 20 m/s	min. 6 l/min, max. 50°C	AC+LC*
Maximum motor temperature [°C]	120								
Motor connection type	UVW or 2x UVW			UVW or 2x UVW			UVW or 2x UVW		
Voltage required for peak power [V_{DC}]**	830			630			250		
Motor peak efficiency [%]	96%								
Peak power S2 2min [kW]	104 kW at 4500 RPM			124 kW at 5500 RPM			124 kW at 5500 RPM		
Continuous power S1 (kW)	55	64	75	55	64	75	55	64	75
Peak torque [Nm]	220								
Continuous torque [Nm]	96	112	130	96	112	130	96	112	130
Limiting speed [RPM]	6500								
K_V constant at no load [rpm/V_{DC}]	10,14			15,53			40,30		
K_V constant at nominal load [rpm/V_{DC}]	7,85			12,05			30,94		
K_V constant at peak load [rpm/V_{DC}]	5,65			8,68			21,91		
K_T constant [Nm/A_{RMS}]	0,94			0,61			0,24		
Peak motor current [A_{RMS}]	235			360			920		
Continuous motor current [A_{RMS}]	120			180			470		
Internal phase resistance at 25 °C [mΩ]***	15,48			7,06			1,35		
L_D induction of 1 phase [µH]	225,5			96,5			15,0		
Induced voltage [V_{RMS}/RPM]	0,07348			0,04793			0,01840		
Magnetic flux – axial [V_S]	0,05728			0,03737			0,01434		
Temperature sensor on the stator windings	KTY 81/210								
Number of pole pairs	10								
Winding configuration	star								
Rotor Inertia [kg*m²]	0,02521								
Bearing configuration	6206 | 3206								
Weight [kg]	12,9	13,5	13,2	12,9	13,5	13,2	12,9	13,5	13,2

*Combined cooled motor (CC) requires cooling specifications from air and liquid cooled motors, to reach its specifications. It cannot only be cooled as an air-cooled motor.
Every EMRAX motor requires sufficient air circulation. The motors should not be completely enclosed in any condition. Please check EMRAX motor manual to learn more.
Performance in your application will depend on your installation details and boundary conditions. Please contact us to learn more.
**All motors are tested for 833V maximum voltage.
***Measured Phase to Phase, then divided by 2.
HV option is operating at speeds lower than its limiting, due to 830 V voltage limitations.

All values given are for a standard 3 phase UVW version, please consult EMRAX on 2x UVW values. $R_{1UVW}=2*R_{2UVW}$

D. Enepaq VTC6 Datasheet

Enepaq

Li-ion building block with Sony/Murata VTC6 datasheet

FEATURES

- Small size: 237-411 Wh per liter
- Low weight: 183-193 Wh per kg
- Individually fuse-protected cells
- Ultra low and equal self-discharge
- Rapid prototyping of battery pack
- Convenient thermal control
- Built-in temperature sensors
- UL94-V0 rated, fire-retardant plastics
- UN38.3 certified

APPLICATIONS

- Performance electric vehicles
- Special purpose machines
- Backup energy storage

INTRODUCTION

A Li-ion building block was developed with simplicity and safety in mind. Using 18650 lithium-ion technology, such building blocks offer the most modern and energy-dense solution in easy to use package. 18650 is a very well established and time-tested battery cell standard, especially common in consumer devices. These cells are designed to withstand consumer's rough treatment, while offering the best in class energy density.

SAFETY

In addition to internal protection techniques, the module includes two fuses for each cell. These act as a second level protection devices in case of cell failure. In case of cell venting, released gas is travelled through dedicated channels to avoid pressure buildup.

FLEXIBILITY

Due to simple nature of these building blocks, desired battery pack configuration can be built in minutes, connecting them in series and parallel using bolt connections. Even MWh-scale batteries can be assembled with ease.

Table 1: Working temperature

	Min.	Typ.	Max.
Discharge	-20 ^oC	25 ^oC	60 ^oC
Charge	0 ^oC	25 ^oC	45 ^oC

Table 2: Product characteristics (all parameters rated at 22°C if not specified otherwise)

Module	Battery voltage (V)			Battery capacity (A)		Fast charge current (A)	Discharge current (A)*1	Initial internal impedance ($m\Omega$)*2	Internal fuse rating (A)
	Min.	Typ.	Max.	Ah	Wh	Max.	Max.	Typ.	Typ.
Li1x1pVTC6	2.5	3.6	4.2	3	10.8	5	30	16.2	45
Li1x2pVTC6				6	21.6	10	60	8.0	90
Li1x3pVTC6				9	32.4	15	90	5.3	135
Li1x4pVTC6				12	43.2	20	120	4.0	180
Li1x5pVTC6				15	54	25	150	3.2	225
Li1x6pVTC6				18	64.8	30	180	2.8	270
Li1x7pVTC6				21	75.6	35	210	2.4	315
Li1x8pVTC6				24	86.4	40	240	2.2	360
Li1x9pVTC6				27	97.2	45	270	2.0	405
Li1x10pVTC6				30	108	50	300	1.8	450
Li2x1pVTC6				6	21.6	10	60	7.7	90
Li2x2pVTC6				12	43.6	20	120	4.0	180
Li2x3pVTC6				18	64.8	30	180	2.8	270
Li2x4pVTC6				24	86.4	40	240	2.1	360
Li2x5pVTC6				30	108	50	300	1.7	450
Li2x6pVTC6				36	129.6	60	360	1.4	540
Li2x7pVTC6				42	151.2	70	420	1.3	630
Li2x8pVTC6				48	172.8	80	480	1.1	720
Li2x9pVTC6				54	194.4	90	540	1.0	810
Li2x10pVTC6				60	216	100	600	1.0	900

*1 - With 80 deg temperature cutoff. *2 - Measurements are done with B&K Precision BA6010 device. *3 - Approximately.

DISCHARGE CHARACTERISTICS

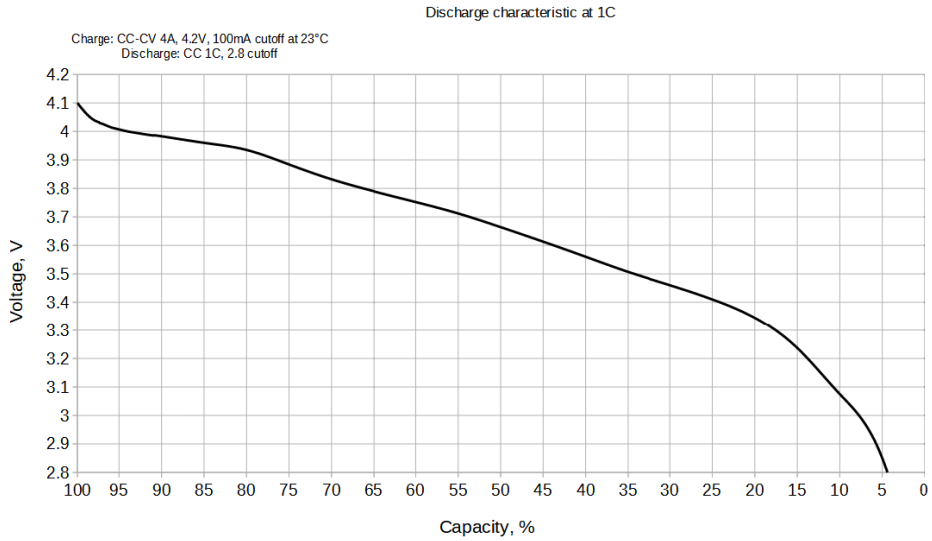


Figure 1: A typical discharge slope at 1C rate.

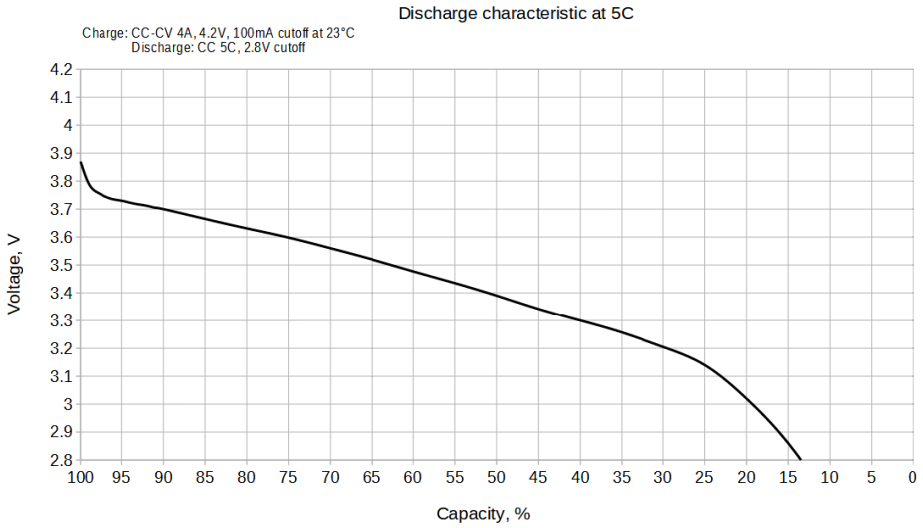


Figure 2: A typical discharge slope at 5C rate.

Test current	Measured energy (Wh)
1C	10.595
5C	9.638

Table 3: Measured energy at different load.

TEMPERATURE SENSOR

The module has a temperature sensor built in, which meets and exceeds safety requirements of FSAE and FSG 2025 regulations. Each one of the sensors is in physical contact with negative pole of two adjacent cells and provides very fast temperature measurement response. Such construction provides inexpensive monitoring of all 100 % of cells ($\geq$ 30 % is required by FSG).

Innovative analogue signal OR'ing technique allows all sensor signals to be read with two-wire acquisition system: output acts as a hot spot detector and reports only the maximum temperature. When battery is operating within safe limits, all sensors report similar temperatures and such measurement accurately represents overall temperature of the module. However, in case of failure event, hot spot is very quickly noticed.

The sensor is a special-made temperature-variable voltage shunt reference. In simple words, it acts as a zener diode, whose voltage drop depends on temperature. It requires a pull-up resistor (680Ω) to operate at cell voltage level. For convenience, the module can be used to power the sensor as given in test circuit in Figure 4.

The signal is non-linear, as given in Figure 5 below. It is compensated internally to provide flattest possible curve in operating range of –40...+120°C. For convenience, conversion values are given in Table 2. Linear interpolation can be used to calculate more values with reasonable accuracy.

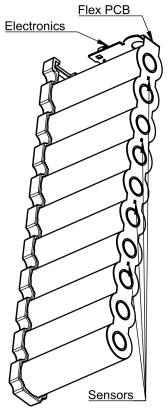


Figure 3: Sensor layout.

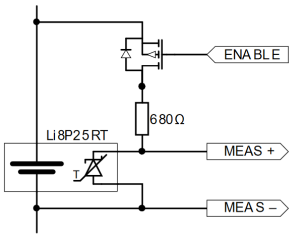


Figure 4: Test circuit.

Table 4: Sensor quantity for modules

Module	1x1	1x2	1x3	1x4	1x5	1x6	1x7	1x8	1x9	1x10
Qty	0	1	1	2	2	3	3	4	4	4
Module	2x1	2x2	2x3	2x4	2x5	2x6	2x7	2x8	2x9	2x10
Qty	1	1	2	2	2	3	3	4	4	4

Table 5: Voltage-to-temperature conversion values

Temp, °C	-40	-35	-30	-25	-20	-15	-10	-5	0	5	10	15	20	25	30	35	40
V_{out}, **V**	2.44	2.42	2.40	2.38	2.35	2.32	2.27	2.23	2.17	2.11	2.05	1.99	1.92	1.86	1.80	1.74	1.68
Temp, °C	45	50	55	60	65	70	75	80	85	90	95	100	105	110	115	120	
V_{out}, **V**	1.63	1.59	1.55	1.51	1.48	1.45	1.43	1.40	1.38	1.37	1.35	1.34	1.33	1.32	1.31	1.30	

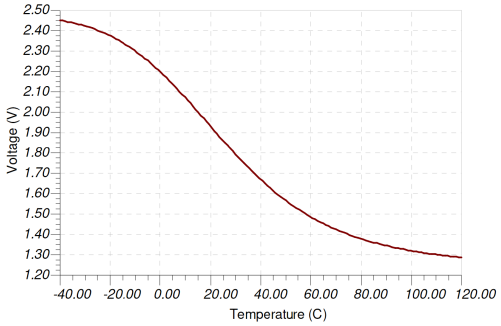


Figure 5: Temperature-Voltage response.

Temperature sensor is galvanically isolated from cell terminals and signal can be safely read with separate circuit. However, it is very convenient to use a standard stack measurement ICs, usually used for battery monitoring and balancing. A circuit example with widely available LTC6803 is given in Figure 6.

To measure temperature, balancing switch is activated on the IC. After doing so, voltage difference between Cn+1 and Cn represents temperature. During such measurement, sensor current flows from cell positive tab through series resistor to sensor, then to internal balancing FET of IC, and then to cell negative via another series resistor. Thus 330Ω resistors are used to form a total of 680Ω resistance for the sensor in this case.

When switch is disabled, cell voltage can be measured. Note that extra care should be taken when adding capacitors for filtered measurement as this could lead to overcurrent condition in the sensor. Also note that adjacent balancing switches must not be enabled as this would also lead to overcurrent condition. If such technique is chosen, measurements should be done in two cycles, on every second cell at a time (for example: 1, 3 and 5, then 2, 4 and 6).

It is recommended to use separate ICs for battery management and temperature measurement, however, with extra care and smart engineering it is possible to use a single IC for cell voltage measurement, temperature measurement and balancing: cells can be bleed-balanced during temperature measurement if additional bleed resistor and MOSFET is added.

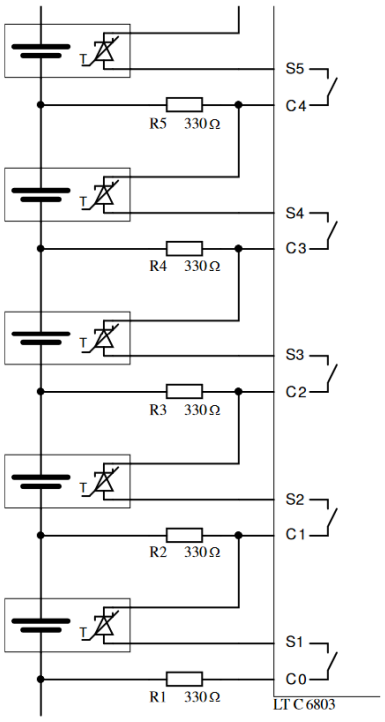


Figure 6: Suggested stack temperature acquisition circuit (simplified).

Table 6: Sensor characteristics (all parameters rated at 25°C if not specified otherwise)

Parameter	**Comment**	**Min.**	**Typ.**	**Max.**	**Unit.**
Supply voltage	$V_{min} = V_{cc} - V_{out}$	10	20	-	mV
	When T = -40°C	1.21	1.24	-	V
Forward current	$I_{reg} = \frac{V_{cc} - V_{out}}{R}$	0.40	1	15	mA
Leakage current	When $V_{cc} < V_{out}$	-	5	400	nA
Measurement range	$V_{cc} > 2.5V$	-40	-	120	°C
Isolation	From cell terminals	-	-	60	V

MECHANICAL DATA

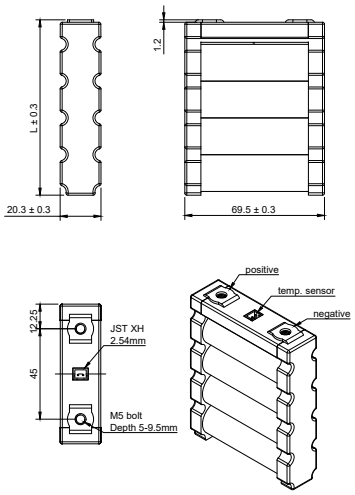


Figure 7: Mechanical dimensions of 1xNp modules

A simplified 3D STP model is available upon request.
1xNp notes:

1. Inner M5 nut is stainless steel
2. Tightening torque: 6 Nm
3. Self-locking washers recommended
4. Modules should be mounted in a firm enclosure to avoid mechanical damage
5. Modules should be protected from direct water ingress
6. Temperature sensor connector: JST XH series

Module	L (mm) ±0.3	Weight* (g)
1x1	32.4	59
1x2	50.7	112
1x3	69.0	168
1x4	87.3	225
1x5	105.6	278
1x6	123.9	331
1x7	142.2	390
1x8	160.5	444
1x9	178.8	500
1x10	195.7	555

*-Without fasteners

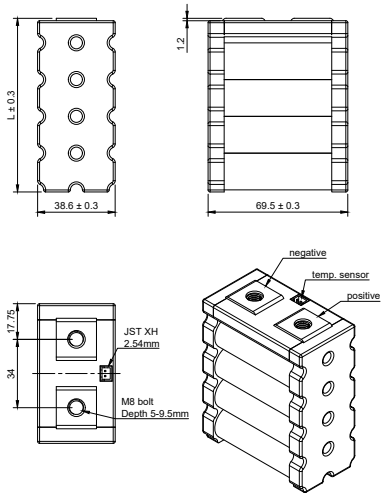


Figure 8: Mechanical dimensions of 2xNp modules

2xNp notes:

1. Inner M8 nut is stainless steel
2. Tightening torque: 10 Nm
3. Self-locking washers recommended
4. Modules should be mounted in a firm enclosure to avoid mechanical damage
5. Modules should be protected from direct water ingress
6. Temperature sensor connector: JST XH series

Module	L (mm) ±0.3	Weight* (g)
2x1	31.5	112
2x2	49.8	225
2x3	68.1	336
2x4	86.4	448
2x5	104.7	560
2x6	123.0	672
2x7	141.3	784
2x8	159.6	896
2x9	177.9	1008
2x10	195.7	1120

*-Without fasteners

TERMS OF USAGE

1. Storage conditions
Module should be stored within a range of temperatures specified as below: Store the battery at 0 ∼ 23℃, low humidity (below 65%), no dust and no corrosive gas atmosphere. Otherwise, it may cause loss of performance characteristics, leakage and/or rust.

2. Long-term storage
2.1 If long-term (but not longer than Warranty Period) storage is necessary, Module shall be stored at shipping voltage, because storage with higher voltage may cause more loss of performance characteristics.
2.2 All devices or components connected to module clamps must be disconnected, preventing current leak from module.
2.3 Enepaq shall not be liable for any defects of module after Warranty Period even if Module is stored in accordance with Sections above.

3. Warranty
Enepaq warrants that Module will be free from defects in manufacturing for a period of 12 months from the date of shipping ("Warranty Period"). In case of defects, Enepaq will only replace the affected modules. However, Enepaq shall not be liable for if:
3.1 The buyer undertakes to check the received goods immediately after receiving the goods. If the Buyer receives the order and notices any quality issues or discrepancies, he must notify the Seller in writing within 7 days from receiving the order. In case of concealed defects, the Buyer must notify the Seller within the warranty term, but always within 7 days from noticing the defect.
3.2 Module was improperly installed, repaired, altered or otherwise modified (other than by Enepaq)
3.3 Module was subjected to misuse, abuse, negligence or accident
3.4 Module was used, handled, stored, sold or distributed in a manner contrary or inconsistent to the handling /use instructions provided in this product specification sheet.

4. Aplication
4.1 The products do not have required certification for aircraft use (any type of aircrafts, drones, planes or other flying, gliding or hovering mechanisms, contraption, devices or other objects) and Enepaq can not be held liable for any damage incurred if the modules are used in such applications. The Buyer takes their own risk for any damages resulting from such use or misuse.
4.2 The Seller does not undertake to indemnify the Buyer of any loss or expenses which are caused by misguided use of Products.

5. Usage
5.1 The Buyer is exclusively responsible for the installation, storage, operation and maintenance of the Goods. The Buyer must ensure that the personnel installing and operating the Goods has appropriate qualifications entitling the personnel to provide handling services with respect to the Goods (including toxic, flammable). The Buyer carries out the installation, storage, operation, maintenance of the Goods, personnel training at its own risk and expense.
5.2 The Seller is not liable for defects in the Goods or inconsistencies of the Goods with the Order if they occurred after the transfer of the Goods to the Buyer due to failure of the Buyer and/or the Buyer's staff to ensure proper installation, storage, operation within specified limits, maintenance of the Goods or due to the fault of third parties or force majeure or other similar circumstances.

Additional notes: *This product is made to order and is non- cancelable and non-returnable (NCNR) and once the order is placed with the factory no changes may be made.*

Table 7: Revision history

Revision	Date	Description
A	2021-10-26	Initial Release.
B	2022-03-24	Company rebranded to Enepaq.
C	2023-09-06	Edited discharge graphs. Added table 3: Measured energy at different load. Added section "Terms of usage"
D	2024-10-28	Updated description of the temperature sensor.

E. NU23 to NU24 All Changed EEPROM Values

EEPROM Description	NU23 Value	NU24 Value
Serial_Number_EEPROM	1933	2858
PWM_Minimum_Variable_Freq_EEPROM_(kHz)	6	12
PWM_Maximum_Variable_Freq_EEPROM_(kHz)	17	12
PWM_Stall_Freq_EEPROM_(kHz)	6	12
PWM_Var_Freq_Mode_EEPROM	1	0
Motor_Type_EEPROM	198	128
Veh_Flux_EEPROM_(Wb)_x_1000	31	34
Gamma_Adjust_EEPROM_(Deg)_x_10	716	-1625
CAN_BMS_Limit_Enable_EEPROM	0	1
CAN_TimeOut_(/3ms)_EEPROM	333	1000
IQ_Limit_EEPROM_(Amps)_x_10	2830	4530
ID_Limit_EEPROM_(Amps)_x_10	1500	2210
Amp_Hours_EEPROM_(AHrx10)	10000	180
DC_Volt_Limit_EEPROM_(V)_x_10	4500	4600
Coast_Lo_EEPROM_(V)_x_100	92	0
Coast_Hi_EEPROM_(V)_x_100	108	0
Accel_Max_EEPROM_(V)_x_100	285	0
Pedal_Hi_EEPROM_(V)_x_100	500	0
Motor_Torque_Limit_EEPROM_(Nm)_x_10	1320	2310
Regen_Torque_Limit_EEPROM_(Nm)_x_10	10	400
Braking_Torque_Limit_EEPROM_(Nm)_x_10	150	0
Motor_Overspeed_EEPROM_(RPM)	7000	6500
Max_Speed_EEPROM_(RPM)	6500	6000
Break_Speed_EEPROM_(RPM)	6500	5500
Speed_Rate_Limit_EEPROM_(RPM/sec)	100	0

Table 7: Comparison of EEPROM values between NU23 and NU24 competition cars

F. CM200DZ Email Chain with Cascadia Motion

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Cascadia CM200DZ fault diagnostics
Date: August 12, 2024, 10:40 PM

Hi,

I am contacting regarding a Cascadia CM200DZ Inverter issue we (University of Newcastle) are having a fault with for our FSAE application.
Since last contacting Cascadia motion directly I have contacted StealthEV (our supplier), however they have not gotten back to me in just under a week now and this a very urgent problem as we are facing pressures as we ramp up our preparations for competition.
Between now and my last contact with CM I have performed a firmware update on the motor controller which was mentioned during a prior email chain.

The fault has presented itself as follows:
a) We have been using the CM200DZ for the last 9 months with our EMRAX188 and recently changed over to an EMRAX228
b) We have performed resolver calibration, setup EEPROM and had success using the EMRAX228MV at our last outing.
c) We began to experience a hardware over-current fault when requesting large torque at low rpm when bench testing
- This setup involved having the motor being attached to a drivetrain and operating normally when slowly brought up to speed by low torque requests
- When stopping the motor and then applying a large torque request (less than 30rpm and greater than 100Nm) the motor will shudder before the inverter throws a HW over-current fault
- The HW over-current fault is occurring consistently on the Phase-A cable
c) We have updated the firmware from 6526 to 6530
d) Following this firmware update the motor no longer reacts to any torque requests
e) We attempted re-calibration of the resolver and noticed
- The motor voltage feedback speed matches the polarity as the feedback speed
- The voltage feedback speed has twice the magnitude as the feedback speed when spinning the motor with an external force up to 1000rpm
- The feedback speed measures accurately when referenced to our external force, whereas the voltage feedback speed reads twice the magnitude
- The resolver calibration Delta_resolver_in_Fil will fluctuate about zero regardless of our Gamma_adjust when rotating the motor
f) At this stage we have not changed our EEPROM settings and were unable to calibrate the resolver due to this.
g) We have now noticed that when powering up the inverter and entering ready-to-drive the phase voltages spike up to 290VDC and remain their until our BMS faults and forces the vehicle out of HV operation.
h) When removing HV +- DC connections to the motor controller we no longer see this 290VDC on the phase cables.

We are now at a position where we cannot re-calibrate the resolver readings. I am wary about their

being internal damage to the inverter as we have not changed around our phase cables recently and now we are unable to accurately measure the delta_resolver when spinning the motor.

I have attached our most recent EEPROM Settings as well as a few diagnostic files downloaded through RMS GUI.
I am eager to hear your response, if there is any more data or information you need please contact me.

Regards,
Joshua Hayward
Powertrain Engineer
University of Newcastle FSAE

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: RE: Cascadia CM200DZ fault diagnostics
Date: August 13, 2024, 3:03 AM

Hi Joshua,

Stealth EV must provide the support since they sold the unit, as we are unable to provide support for 3rd party customers. Please pursue help from Stealth EV.
You've done a nice job outlining the situation and providing data files.

Some quick thoughts:
I see by the EEPROM provided that you're motor type is consistent with the EMRAX 288MV you're running now.
If there's been any changes to the resolver wiring harness wiring and/or routing since it last operated normally, review that.
Consider reverting back to FW 6526 and last working EEPROM to confirm it operates ok as before.

Best regards,
Kirk

Kirk Swaney
Sales & Program Manager
+1 503.482.2690
CASCADIA MOTION
Wilsonville and Hood River, Oregon

G. CM200DZ Email Chain with Stealth EV

From: Support Stealth <support@stealthev.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: CM200DZ trouble shooting
Date: August 29, 2024, 3:30 AM

Hello Joshua,

Can you give me more information about how your system is setup?

What is your system voltage?
What are your battery specs? Cells, BMS, kWh?
Are you connecting your battery directly to the inverter or are you using an HVJB?
Are you using a VCU?

Have you tried going back to the Emrax188? What version of the Emrax188 are you using?

Thank you,
Stealth EV LLC
3830 Oceanic Dr #401
Oceanside, CA 92056
Contact 888-515-7514
Email: Sales@StealthEV.com

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Support Stealth <support@stealthev.com>
Subject: Re: CM200DZ trouble shooting
Date: August 30, 2024, 1:23 AM

Hi,

I have the following information:
We use Energus li6pvtc6t modules in 108S at 18Ah
Our nominal pack capacity is 7.2kWh @ our nominal voltage of 399.6V
Our maximum pack voltage is 453.6V

The inverter is connected directly to the battery, there is a node on the car we use but the circuitry contains only busbars and a HVD
We don't use an off the shelf VCU, however we run the inverter in CAN operation with our command message being send from our pedal box node which sends the following:
Direction command, Torque Command and Inverter enable.

We have not tried putting the 188HV back into the car as it would require refrabrication of our old chassis as we have moved onto using a new chassis for competition in FSAE.

The car worked as expected for one day of shakedown testing after commissioning the 228MV, on our next outing it was unable to move.
We have tested the 228MV for isolation and resistance across the phase windings to match our second 228MV in stock.

Regards,
Joshua Hayward
Powertrain Engineer
University of Newcastle FSAE

From: Support Stealth <support@stealthev.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DZ trouble shooting
Date: August 30, 2024, 3:11 AM

Hello Joshua,

Attached are the manuals for the CM200 and Cascadia's instruction manual on setting up the Emrax motor.

You mentioned that there is a fault on phase cable A. Are all your phase cables shielded? From the CM manual on page 31:

The CM200 uses Rosenberger HPK connectors for the AC connections. The cables must use shielded wire. The shield is connected to the chassis of the inverter and the chassis of the motor.

Other things to check:
BMS DCL settings
If possible, connect your inverter to the 188HV and spin.
Make sure the inverter is getting enough cooling.

Thanks

Thank you,
Stealth EV LLC
3830 Oceanic Dr #401
Oceanside, CA 92056
Contact 888-515-7514
Email: Sales@StealthEV.com

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Support Stealth <support@stealthev.com>
Subject: Re: CM200DZ trouble shooting
Date: September 5, 2024, 1:26 AM

Hi,

We have setup the EMRAX228MV as per the documentation, it worked on our first outing with the vehicle and now seemingly does not work.

All phase cables are shielded between the motor and the inverter.

The BMS DCL settings have been removed from the application with no change in operation. We are still unable to get the inverter to spin our motor, and it cannot correctly determine the Voltage Feedback Speed.

It is not possible for us to connect the 188HV as our only remaining resolver is attached to the 228MV with retaining compound and it would risk damaging the component.

The inverter receives sufficient cooling as we have an engineer overseeing the cooling system and assures me it has the necessary flow and pressure, which our data backs up as nothing has overheated.

Is there anything else we can do as I am worried we have exhausted our diagnostic testing and need to if this is an issue caused by us or if it is the CM200DZ malfunctioning before we buy a replacement.

Regards,
Joshua Hayward
Powertrain Engineer
NU Racing FSAE team

H. CM200DX Email Chain with Cascadia Motion

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: CM200DX Purchase - University of Newcastle
Date: September 14, 2024, 2:56 AM

Hi,

I am reaching out to Cascadia after our recent purchase of a CM200DX inverter. (I have attached our purchase invoice)

We are a university based FSAE racing team who have used Cascadia Inverters in our previous race cars. I had previously contacted Cascadia Support regarding a malfunction that occurred with our CM200DZ during a track testing day. I am now emailing to acknowledge that I have attempted to obtain support from Stealth EV, and subsequently HyperCraft USA, to find the root-cause of our CM200DZ hardware malfunction. This has been rocky, and after relaying our specifications and configuration to their support team I have not had a response as to identify the inverter malfunction.

I want to preface that I am not seeking support for the CM200DZ that we purchased from Stealth EV (although I would love some). I more-so want to touch base regarding the setup of our inverter and what we can do to mitigate any potential further damage we may incur to our new CM200DX inverter. Through our own examination, we cannot draw any conclusions as to why our inverter failed, but due to the lack of Stealth EV support and the time constraints placed upon the team it was decided best to purchase a new inverter directly through Cascadia.

I would like to hear from you regarding your stance on this type of support, as I would like to maintain a good relationship with Cascadia as I am also a customer of their products outside of University (we plan on using an IM-375 for our EV conversions at my workplace). Ideally, I would like to arrange for a meeting with an engineer to discuss our current implementation of the CM200DZ which followed all Cascadia documentation in setup, and if there is any adjustments we should make moving forward with the CM200DX. If this is something you would consider please let me know so I can arrange for a time with our Chief Engineer's, this may be need organising considering Cascadia's operating hours fall into 2am-10am Australian local time.

I have documented the events that led up to this malfunction, including capturing our EEPROMs, and several diagnostic data files recorded through RMS GUI & CoolTerm.

As always I appreciate your timely response and if there is anything I can do such as sending diagnostics, our system overview or arranging for a meeting do not hesitate to contact me.

Best regards,
Joshua Hayward
Powertrain Engineer
Newcastle University FSAE Team.

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: September 21, 2024, 7:42 AM

Hi Josh,

Regarding the CM200DZ purchased through a third party. That support needs to be through the party you purchased from. Regarding the CM200DX purchased with us we will support. It sounds like we are in a gray area here since the original failure was with an inverter aquired from a third party, but you want to make sure that there is no issues with using the CM200DX purchased with us. Our support will mainly focus on the CM200DX purchased with us.

What is the failure that you are concerned about?
Is there any system differences? The original failure you had was on a CM200DZ. The new inverter is a CM200DX which is for a different voltage range. Is this the same vehicle with some modifications or an entirely different vehicle?
A high voltage wiring diagram of your system as well as a low voltage wiring diagram of your system would be helpful.
Regarding a meeting, I would like to gather more information about the potential issue, and information about your system before a potential meeting. This is to make sure that the correct team members at Cascadia Motion are up to speed on this.

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)
Wilsonville, Oregon (USA)
www.cascadiamotion.com — alouie@cascadiamotion.com

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: September 24, 2024, 5:29 PM

Hi Andrew,

We are concerned about a failure which I have contacted Cascadia previously regarding. I have provided an overview of what I had previously mentioned.

We are now at a position where we cannot re-calibrate the resolver readings. I am wary about their being internal damage to the inverter as we have not changed around our phase cables recently and now we are unable to accurately measure the delta_resolver when spinning the motor.

Our system topology hasn't changed from when we changed from and EMRAX188HV to EMRAX228MV Our battery voltage is 399.6V nominal with 453.6V maximum.

We changed to a DX unit from a DZ unit as we do not plan on utilising the higher voltage capacity of the DZ motor controller in the near future.

Our Low voltage connections to the motor are simple due to the application:
KL30, KL15 and KL15_EN are connected to a common +12V ignition switch, KL31 is connected to Ground.

For our HV system I have attached a topology of the connections between the accumulator and the HV input of the CM200 inverter.

I have also attached the topology of our pre-charge circuit.

Previously, our trigger for the shutdown circuit would receive the same CAN message as the inverter disable command for the inverter CAN Command message.
This means we would disconnect HV at the same moment as removing the inverter enable message.
The trigger for the shutdown circuit now has a 250ms delay from when the inverter receives an inverter disable command over CAN from our VCU.

Please let me know if you need any other information, we have in-depth schematics for most of our electronics.

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering
Email: Joshua.Hayward@uon.edu.au
Web: www.nuteams.org
University of Newcastle Callaghan Campus, NSW 2308, Engineering Cl

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: September 26, 2024, 9:39 AM

Hi Joshua,

The attachments did not transfer over correctly. They just show up as the broken image picture even after I hit trust sender. I'm not sure why that happened. Might need to try making a share point or some other type of cloud storage sharing solution and sharing the link to that.

It's difficult for me to determine what happened with your CM200DZ. There was some odd behavior going on and it's not clear if that's because it was already broken at that point. For example the double reported voltage speed can occur if there is something wrong with the hardware but can also occur if there is a setup issue between EEPROM and firmware.

For the EMRAX 228MV motor, as long as you don't exceed the max speed of 6500rpm it is safe to the inverter to spin that motor with an external force. For this particular motor you can do so without applying any DC high voltage to the inverter. The back-emf will remain low enough to not exceed the maximum voltage of the inverter. For human saftey, do keep in mind that the inverter DC bus voltage will charge up from the back-emf of the motor and needs to be safly discharged before handling. Therefore, I would start with resolver calibration by spining with an external force and not apply high voltage to the inverter during this test.

As for testing with high voltage applied I still need to review the schematics that didn't come through.

Regarding the safe shutdown. On this particular motor the inverter should be able to handle DC bus disconnecting form high voltage in the middle of operation. It may not be safe for the rest of the system depending on the setup but the inverter should be able to handle it in this case. However if something is shorting the DC bus on shutdown then that could cause an issue. For example some DC power supplies will short the DC when turning off. That would not be good if that happened while the motor was still spinning. If there is any safty discharge things from other devices such as BMS that would short the bus out then we should review that. The inverter it's self has some bleed resistance to slowly discharge the bus but it's not agressive enough to draw significant current.

Keep in mind, I'm giving advice that is specific to this setup and may not apply to every setup.

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)
Wilsonville, Oregon (USA)
www.cascadiamotion.com — alouie@cascadiamotion.com

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: September 27, 2024, 7:58 AM

Hi Joshua,

We reviewed the information provided. The state flow diagram doesn't show when discharge relay is active. Keep in mind that if discharge is engaged while the motor is spinning the discharge circuit can have power generating through it from the motor. The system diagram does not show how the pre-charge circuit is connected to the system.

Also, below is the feedback from our electrical engineer. "They need a full current rated contactor to run the power once it's up and running.

On their pre-charge drawing, I can say two things.
The 2n7002 is a tiny, old Mosfet. It can be up to 7.5 ohms of RdsON, which is not good for closing the relay. I'd look for a FET that was 0.1ohm Rds or better and 100V breakdown voltage for relay driver.
Usually the use of a freewheeling diode across a relay coil is NOT advised. You need to let the voltage fly up a little bit so the relay "opens fast" Otherwise it can weld on opening! They could use a "bidirectional TVS" if they're concerned about their FET, for example a TP6KE30CA –bidirectional—TVS. Or take a look at this... https://www.te.com/en/products/relays-and-contactors/relays/intersection/relay-coil-suppression-dc-relays.html?tab=pgp-story " - Alex

So far we did not find anything that would damage an inverter.

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)
Wilsonville, Oregon (USA)
www.cascadiamotion.com — alouie@cascadiamotion.com

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: September 28, 2024, 1:15 AM

Hi Andrew,

I have talked to our Chief Mechatronics engineer who has notified me that our discharge is not Engaged while the motor is spinning, it is only engaged whilst the SHUTDOWN_IN is low (NC).

Our contractor is the GigaVac Gx12 for which I have attached the data sheet.

The only other thing I can think of is sending diagnostic files from when our previous inverter entered a fault state.

I have been attempting to contact Stealth EV/HyperCraft USA for support on this old CM200DZ unit however they no longer reply to my emails.

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering
Email: Joshua.Hayward@uon.edu.au
Web: www.nuteams.org
University of Newcastle Callaghan Campus, NSW 2308, Engineering Cl

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 3, 2024, 9:05 AM

Hi Joshua,

We reviewed the additional documents provided. We don't see anything related to system setup that is a danger to the inverter. I reviewed the diag data from the failed inverter and it looks like the cause of failure was loss of current regulation in stall. Stall and or low speed is particularly stressful to the inverter because all the current can remain in one phase for long durations. There is derating that try's to protect the inverter during stall but when current regulators lose regulation that software protection doesn't protect the inverter from damage. The reason for loss of current regulation seems to be due to change in PWM that occurred during stall. Variable PWM is more refined for the motors we sell. We do a lot of additional tunning for our motors to work with variable PWM. The EMAX motor has not been tuned to work with variable PWM. From the data it appears that it's not working well with variable PWM. Particularly, it's not working well at 6kHz. I recommend only operating the EMAX motor at 12kHz. That is the PWM frequency that it was tunned for. Use the following settings:

PWM_Nominal_Freq_EEPROM_(kHz) 12
PWM_Minimum_Variable_Freq_EEPROM_(kHz) 12
PWM_Maximum_Variable_Freq_EEPROM_(kHz) 12
PWM_Stall_Freq_EEPROM_(kHz) 12
PWM_Var_Freq_Mode_EEPROM(0=NOM_1=VARS_2=VAR) 0
Settings in bold are the important ones. The others should not actually apply.

I think that will resolver the issue. Keep in mind adequate coolant flow is needed at all times when the inverter is enabled. I'm not saying cooling was an issue but just giving a reminder.

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)
Wilsonville, Oregon (USA)
www.cascadiamotion.com — alouie@cascadiamotion.com

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 3, 2024, 9:30 AM

Hi Andrew,

Thank you very much for your ongoing support with this new inverter. We appreciate your time and look forward to having our FSAE car back up and running.

We have calibrated our resolver and plan on doing our first spin testing tomorrow. I have one question regarding the calibration process; When spinning the motor with an external force at 800rpm feedback speed we have calibrated to a value of 900 for Delta_Resolver_In_FIL(DEG)(x10). When we speed up the external force to 1100rpm we approach 890 for the Delta resolver watch value.

The operating speed of our EMRAX228 is up to 6500rpm. Do you suggest we calibrate for the higher rpm? If so do you have a minimum RPM we should target for calibration?

Lastly, I wanted to understand the effects on the inverter when we are spin testing the motor with the vehicle on jack stands? Typically this is low commanded torque due to the low-load nature of the setup. Should we implement additional torque and/or rpm limits for this type of testing? Before we go out for testing days with the car we have to spin it on the stands to ensure it is working correctly.

Thank you again Andrew, we appreciate your insight.

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering
Email: Joshua.Hayward@uon.edu.au
Web: www.nuteams.org
University of Newcastle Callaghan Campus, NSW 2308, Engineering Cl

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 3, 2024, 10:00 AM

Hi Joshua,

Take a look at our latest resolver calibration manual on the website.
https://app.box.com/file/23982244943?s=35n0u8b2sv1cc5pyf86pote73ffgi9xj
It was update in April this year. There are some changes with the latest firmware on how we do resolver calibration. In particular get familiar with Delta_Resolver_In_Fil_On_Coast(DEG)(x10)

Delta_Resolver_In_Fil_On_Coast(DEG)(x10) already grabs the measurement at the speed that we think is ideal for calibration. See the manual for how that is selected. I think section 3 appendix gives a good example.

Angle_Advance_Factor_EEPROM_x_100 is also a new EEPROM setting that is related to resolver calibration. Angle advance factor is now compensating to give better calibration across the entire speed range. Setting to zero will not use it and is the same as what we used to do. Unfortunately, it looks like there is currently no good documentation on Angle_Advance_Factor_EEPROM_x_100 and how it should be used in resolver calibration. I have notified the team to add that documentation. I will notify you when that documentation is added.

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)
Wilsonville, Oregon (USA)
www.cascadiamotion.com — alouie@cascadiamotion.com

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 5, 2024, 1:00 AM

Hi Andrew,

I have spent the greater part of today setting up our CM200DX into our FSAE vehicle.

I have a question regarding the CAN integration, particularly if I have migrated the Cascadia DBC correctly with our own local DBC.
This is something we have adapted from our CM200DZ which ran on older firmware 6526. I see there have been some additions to the CAN messages that are being sent. In 6530.

At the moment our system is operating correctly and the CAN bus is at 20% utilisation with the inverter disconnected.
When I connect the inverter (both power and CAN) to our network, we see high utilisation (upwards of 95%) and many error frames over CAN.
This gets particularly evident when setting the Inverter Enable byte of the command message to 1. This leads our inverter to have a CAN message timeout as it consumes all the traffic of the bus.
It then enters inverter enable lockout and we are unable to power the vehicle.

I have attached our DBC, current EEPROMS and relevant VCU code for sending the CAN command message.

We have organised to hold a track testing day tomorrow (in 8 hours) so I figured I would email to see if you have any suggestions on this integration.

If there is anything else I can provide to help with this please email me.

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering
Email: Joshua.Hayward@uon.edu.au
Web: www.nuteams.org
University of Newcastle Callaghan Campus, NSW 2308, Engineering Cl

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 5, 2024, 7:14 AM

Hi Joshua,

See our latest CAN manual.
https://app.box.com/s/vf9259qlaadhzxqiqrt5cco8xpsn84hk/file/27334613044

Turning off the High Speed message will free up the bus a lot, however you already have that turned off. You can also turn off some massages you don't need to free up the bus more. Increasing CAN_Fast_Msg_Rate_EEPROM_(ms) and CAN_Slow_Msg_Rate_EEPROM_(ms) could also help.

Below is a link to the latest DBC on our website
https://app.box.com/s/gu8b4utzjsrx17in8vipgfqxpt8fuf1u/folder/134048861771
That one is going to match the new software better. Differences in the DBC will not effect bus utilization though.

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)
Wilsonville, Oregon (USA)
www.cascadiamotion.com — alouie@cascadiamotion.com

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 8, 2024, 12:34 AM

Hi Andrew,

I have addressed the CAN dropout we were experiencing by adding ground straps to the following:
- CM200DX Housing
- EMRAX228 Powerbox (where it is mounted)
- DCDC
- HV Battery enclosure

These all now ground to a common chassis ground which is tied to our DCDC negative terminal.

This had resulted in the CAN network now partially operating when sending our enable inverter command message. I have included logged CAN data from before and after installing these ground straps, they are attached as 'Inverter Enable before grounding wires' and 'Inverter Enable after grounding wires' . Specifically note the rolling counter CAN messages which are monitored at 100Hz. There is however still noise on the network.

From here I began testing our inverter supply power with an oscilloscope. We found noise that was induced corresponding to the PWM periods.
I have attached another image 'Inverter power to ground' which shows the period between these noise artefacts corresponds to 24kHz, where our PWM frequency was set to 12kHz.
This was validated by changing the PWM frequency to 10kHz where we observed 20kHz noise.

I am assuming this is a result of the IGBT's inducing noise into our system either through the wires in the CM200DX low-voltage loom, or through the phase cables?
The noise is being induced into our CAN signals from CAN Ground I assume and it is still causing many error frames.

My questions are:
1. How should we be shielding the CAN lines from the inverter to our VCU? Should CAN ground be inside or outside of the shield? Should the shield be connected to ground at the VCU or inverter side?
2. Should we also be shielding the other LV connections to the inverter? If so, should we include this inside the CAN shield? Should this be grounded to the inverter housing or chassis ground?
3. Should we be shielding our high voltage DC bus cables?
4. Are there any other EMI prevention practices we should be implementing?
5. Is there any testing we can do by either measuring resistance, capacitance, or testing with an oscilloscope? Or any other system topology that would be useful for your team?

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 9, 2024, 12:54 AM

Hi Andrew,

I just thought I would add some context to my prior email below:
- I have tested the inverter with a spare EMRAX 188HV that was laying around
- I have tested the inverter with our previous phase cables
- I have tested the inverter being powered with a remote power supply for LV power wires
- I have tested the inverter being on an isolated CAN bus and having the enable message sent from a computer using Kvaser CanKing

All of this testing results in the same 24kHz noise we have been experiencing.

I hope this helps with giving us advice on how to diagnose and/or isolate this issue so we may get back to dynamic testing as soon as possible.

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering
Email: Joshua.Hayward@uon.edu.au
Web: www.nuteams.org
University of Newcastle Callaghan Campus, NSW 2308, Engineering Cl

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 9, 2024, 7:42 AM

Hi Joshua,

The most critical is shielding the motor phase cables. That has the largest effect and so you should double check that first. You must have the shield terminated to both the drive housing and at the other side to the motor housing. The termination length to the motor housing is somewhat critical, it should be as short as possible. How it's terminated on the motor side is also important. The shielding should be crimped to a ring terminal and that ring terminal should be bolted to the motor. Shield current on the motor phases are significant and should be done with components that can handle significant current, and minimal contact resistance. Verify that you have good continuity from phase shield to motor chassis. Verify that you have good continuity from phase shied to inverter chassis.

Some don't run CAN_GND between the VCU and the inverter. I'm not sure if using the CAN_GND helps or hurts the noise issue. We don't have any indication that it would but may be worth a test with CAN_GND disconnected. Let me know how that test goes. If you do run a discrete wire for CAN_GND I'd have it within this shield or along the shield. We recommend terminating the CAN shield to the CAN_GND pin position D2.

I don't think shielding the other LV wires will do much for this, but you could try. I'd shield the CAN separately. I would try a separate twisted pair shielded cable for 12V inverter power. Terminate both ends of the shield to chassis. I have not tried this but on one application I did find that routing the 12V cables away from everything else helped significantly.

Using shielded wire on the HVDC may help. Our Rosenberger connectors do provide shield to chassis links.
Some people terminate the shield to the battery enclosure, and I believe some don't. You can try both and see what works best.

If you're using a discrete 120 ohm terminator at the drive, you might see improved CAN performance by removing this terminator and making use of the CAN_T feature, where you short CANA_T to CANA_L. This engages an internal terminator in the drive with some additional filtering. The wire that shorts CANA_T to CANA_L should be short, right at the connector into the drive. Unfortunately this means splicing the CANA_L wire or crimping two wires into the CANA_L pin.

You might try clipping a common mode choke bead over the CAN wires (both CANA_L and CANA_H and if included CAN_GND). If you do use beads, they should be the low frequency type, very high permeability. We like the Laird LFB series in solid cores, for a clip on the Wurth Elektronik STAR-TEC LFS series.. you might try the Wurth Elektronik 74272222

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 18, 2024, 11:56 PM

Hi Andrew,

After this week of troubleshooting our noisy CAN bus we have been able to reduce the CAN transmission errors down from 10% to approximately 1% of the bus. This was primarily due to running new, twisted pair shielded CAN wires from the inverter, without connecting the CAN ground to our system 12V ground. We did however notice that connecting the phase cable shielded ground to the motor chassis we saw an increased amount of CAN transmission errors to around 2-3% of the bus.

When we then tried to spin our motor, without load, by applying a 10Nm torque command, we noticed it would partially spin, before locking itself into position. When increasing this torque command to 20Nm, we see that the motor begins to shake and jitter in place.

At this stage it was noticed that when the motor is not spinning but only after enabling the inverter, our Delta_Resolver_in_Fil is reading sporadic values. It was also noticed that the voltage feedback speed would read sporadically random values in the range of 0-3000rpm.

When the motor is not spinning and not enabled, we see constant, 0 values until spinning the motor by hand where we see the correct angle changes. It should be noted that in both scenarios, the value of Feedback_speed (corresponding to the resolver) is 0, unless the motor is spun by hand where it correctly indicates speed.

Is there any advice or EEPROM's we should be considering to resolve this issue? At this stage we are happy with the CAN, but are still not able to have the motor complete more than a quarter of a revolution before locking itself into a position and shaking. We have also done our best to shield the resolver wires as they sit quite close to the phase cables.

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering
Email: Joshua.Hayward@uon.edu.au
Web: www.nuteams.org
University of Newcastle Callaghan Campus, NSW 2308, Engineering Cl

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 19, 2024, 3:17 AM

Hi Joshua,

Is the partial spin behavior the same if the phase shield is connected to the motor. Also is the motor connection a good electrical connection. I think the EMRAX motor has some black coating and I don't think that is very conductive. Conduction should be through a mating surface not through screw threads.

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 21, 2024, 11:32 PM

Hi Andrew,

Regarding your previous email, our motor will not spin with any configuration of connecting phase cable shields to motor, changing phase cables, swapping phase terminals, reprogramming old EEPROMs, firmware update, or otherwise.

In an attempt to try and get the motor re-calibrated we used our previous bench setup and spun the motor up to 2000rpm using a drill. It was at this point that we noticed Feedback speed was reading 2000rpm whilst the Voltage feedback speed was reading 4000rpm. We tried different speeds and always noticed that the polarity was the same but the magnitude of the voltage feedback speed was always double that of our feedback speed. I verified this using a tachymeter and can confirm the Feedback speed is reading true, whilst the voltage feedback is double the actual motor rotational speed. This stayed true even when we changed our motor out for our second EMRAX 228 that is unused. This was not the case when we received the CM200DX, as we were able to successfully calibrate the resolver, however the motor did not spin.

For added context, we are using a CM200DX, EMRAX228MV and Tamagawa 5X resolver, TS2620N1095E161 as per the 'Setting up Emrax Motors' guide. Our motor type EEPROM is 128.

I appreciate your ongoing support to our University. Our priority as a team is to get our vehicle operational as soon as possible and it is difficult to make substantial progress with currently being able to exchange only 1 email per day due to the time zone differential. I was wondering if it would be possible to organise either a Zoom meeting, phone call or even just an allocated time where myself and our Chief Engineers could email you back and forward to maximise our chances of diagnosing and resolving this as soon as possible as we have our FSAE competition coming up very soon. Our time zone is Australian Eastern Daylight Time (AEDT) UTC+11.

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering
Email: Joshua.Hayward@uon.edu.au
Web: www.nuteams.org
University of Newcastle Callaghan Campus, NSW 2308, Engineering Cl

From: Cascadia Motion Support <support@cascadiamotion.com>
To: Joshua Hayward <Joshua.Hayward@uon.edu.au>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 25, 2024, 8:53 AM

Hi Joshua,

I set up a teams meeting for 5pm your time, I think. Let me know if that works for you. I can move it later if needed. I can also move it 30 minutes earlier if needed. I can't move it any earlier than that though.

If you have time before the meeting, please use the RMS GUI and in memory view tab add Group_Number. Send me a screenshot of the RMS GUI, showing the top bar of the RMS GUI and the Group_Number.

For our meeting please be ready to share the screen of the RMS GUI connected to the inverter. We will probably do some tests where you spin the motor with a drill and I look at various parameters on the RMS GUI.

Regards,
Andrew Louie
Controls Engineer — Cascadia Motion LLC (a BorgWarner company)

From: Joshua Hayward <Joshua.Hayward@uon.edu.au>
To: Cascadia Motion Support <support@cascadiamotion.com>
Subject: Re: CM200DX Purchase - University of Newcastle
Date: October 25, 2024, 9:24 AM

Hi Andrew,

Thank you very much for organising the meeting, I can see your teams meeting has been organised for 4am AEDT which is our local time zone. We appreciate this and are able to meet this time, if it is possible to move it back a few hours closer to 7-8am AEDT that would be optimal. If you are based in Oregon this would be around 2-3pm for you.

If this is not convenient for you, we are happy to accomodate 4am AEDT (10am Oregon).

I will get the RMS GUI Group_Number screenshot for you today.

Kind regards,
Joshua Hayward
NU Racing - Powertrain Engineer
NU Teams is the official on-campus professional student engineering teams at the University of Newcastle.
Bachelor of Mechatronics Engineering
Email: Joshua.Hayward@uon.edu.au
Web: www.nuteams.org
University of Newcastle Callaghan Campus, NSW 2308, Engineering Cl

I. PEN Autocross ECU Code

```
// Pedal Box ECU
// Jacob Bush & Jacob Lukes & Alec Chapman
// Regenerative Braking, Current Limiting and Power Limiting code by Joshua Hayward
// 25/5/24

// Include necessary libraries and define pins
#include <NU24_CAN.h>
#define BPSF_PIN A1
#define BPSR_PIN A0
#define TPS2_PIN A2
#define TPS1_PIN A3
#define BSPSPRESOCSC_PIN 2
#define BSPDPRESTHRESH_PIN 3
#define SDSTART_PIN 4
#define SDBOTS_PIN 5

// Throttle calibration values
#define TPS1_OFFSET -152
#define TPS1_SCALE 100/682
#define TPS2_OFFSET -76
#define TPS2_SCALE 100/338
#define APPS_FLOOR 2.5            // [0-100%] Minimum throttle above FLOOR
#define APPS_CEIL 95              // [0-100%] Maximum throttle possible
#define BRAKETHRESHOLDFRONT 100   // [psi] Front Brake Press to trigger BRAKE_SIG
#define BRAKETHRESHOLDREAR 100    // [psi] Rear Brake Press to trigger BRAKE_SIG

// CM200 Variables
/*-------------------------------------------------------------------------------------*/
  // The scaledTorqueCommand Graph can be viewed and below variables adjusted to visualise the 'Torque Curve'
  // https://www.desmos.com/calculator/jevbkcgozk
/*-------------------------------------------------------------------------------------*/
#define MAX_POWER 80000             // Sets the controllers max achievable power based on pack voltage
#define ZERO_TORQUE_APPS 10         // Adjustable to desired zero torque crossover point as a function of APPS %
#define MAX_MOTOR_TORQUE 180        // Adjustable to desired motor torque (Nm) [MAXIMUM VALUE = 231]
#define MAX_REGEN_TORQUE 0        // Adjustable to desired regen torque (Nm) [MAXIMUM VALUE = 50]
#define COMMAND_MSG_ID 0x0C0        // Motor Torque ID as defined by Cascadia Motion

// APPS appsPlaus state machine states
enum STATEVAR {
  STATE_PLAUS,
  STATE_WAIT,
  STATE_IMPLAUS
};

// Define states for the RTD state machine
enum RTDState {
    STATE_INIT,
```

```
48      STATE_RTD,
49      STATE_RTD_WAIT
50  };
51
52  // Initialise and declare variables
53  // RTD
54  RTDState rtdStateMachine = STATE_INIT;
55  unsigned long stateChangeTime = 0;
56  int inverterRTD = 0;
57  // APPS
58  int plausFlag = 0;
59  int timeStamp = 0;
60  elapsedMillis timer;                // Timer for APPS_brake floor
61  int implausStart = 0;               // APPS implaus timer
62  STATEVAR state = STATE_IMPLAUS;     // Initialise appsPlaus as implaus
63  float appsOutRaw = 0;               // [0-100] Percent throttle request
64  float appsOutGated = 0;             // [0-100] Percent throttle request gated
65  float appsTrail = 0;                // [boolean] brkSig & Throttle at same time OKHS
66  float appsPlaus = 0;                // [boolean] throttle gate OKHS
67  float tps1Raw = 0;                  // [0-1023] Raw input from TPS1 pin analogRead
68  float tps2Raw = 0;                  // [0-1023] Raw input from TPS2 pin analogRead
69  float tps1Scaled = 0;               // ~[0-100] TPS1 scaled to percent range
70  float tps2Scaled = 0;               // ~[0-100] TPS2 scaled to percent range
71  float tps1Bounded = 0;              // ~[0-100] TPS1 scaled to percent range and bounded
72  float tps2Bounded = 0;              // ~[0-100] TPS2 scaled to percent range and bounded
73  // BPS
74  float bpsFRaw = 0;                  // [0-1023] Raw input from BPS1 pin analogRead
75  float bpsRRaw = 0;                  // [0-1023] Raw input from BPS2 pin analogRead
76  float bpsFScaled = 0;               // [kPa] Front brake line pressure
77  float bpsRScaled = 0;               // [kPa] Rear brake line pressure
78  float bpsFBounded = 0;              // [0 - 2000] bounded pressure
79  float bpsRBounded = 0;              // [0 - 2000] bounded pressure
80  float brkSig = 0;                   // [boolean] brake 'switch' output from pressures
81  // States
82  float sdStart = 0;                    // [boolean] SD Circuit Post DEN (pre BOTS) OKHS
83  float sdBots = 0;                   // [boolean] SD Circuit Post BOTS (pre Dash E-Stop) OKHS
84  float bspdPressureOCSC = 0;         // [boolean] BSPD pressure open-short fault detection OKHS
85  float bspdPressureThresh = 0;       // [boolean] BSPD pressure threshold fault detection OKHS
86
87  // Set Motor Controller variables
88  static CAN_message_t commandMessage;
89  float packVoltage = 0;
90  float packCurrent = 0;
91  float motorSpeed = 0;
92  float motorSpeedRPM = 0;
93  float speedKph = 0;
94  float rtdState = 0;                 // [boolean] CAN input ready to drive signal to trigger
        enable pin
95  float sdInertia = 0;
96  float sdEStopDash = 0;
97  float sdTSAL = 0;
98  float sdHFR= 0;
99  float sdTSMS = 0;
```

```
100 float SoC = 0;
101
102 // Current Limit variables
103 static CAN_message_t CURRENT_LIMIT;
104 float motorSpeedRadPerSec = 0;
105 float maxTorque = 231.0;
106 float requiredCurrent = 0;
107 float DCL = 180.0;
108 float CCL = 30.0;
109 float efficiency = 0.9;
110
111 //  CAN Bus Variables
112 //  inputmsgs defines message, outputVar defines location for incoming data
113 float *outputVar[] = {&rtdState, &sdInertia, &sdEStopDash, &sdTSAL, &sdHFR, &sdTSMS, &
        motorSpeed, &packVoltage, &packCurrent, &SoC};
114 canmsg *inputmsgs[] = {&RTD_STATE, &SD_INERTIA, &SD_ESTOP_DASH, &SD_TSAL, &SD_HFR, &SD_TSMS,
         &INV_Motor_Speed, &V_PACK, &I_PACK, &SOC};
115 int numreceive = 10;
116 int numsend = 20;
117 // CM200 Command Message format
118 int scaledTorqueCommand = 0*10;      // Bytes 0, 1. Units of Nm * 10. Previously defined as
        a float, changed to integer since the remainder operation (%) does not work with floats
119 int speedCommand = 0;                // Bytes 2, 3. Signed value, +ve = forwards, -ve =
        backwards
120 int directionCommand = 0;            // Byte 4, bool. 1 = forward, 0 = backwards
121 int inverterEnable = 0;              // Byte 5, bit 0, bool. Enable after rtd. Turn enable
        off whenever rtd is lost
122 int inverterDischarge = 0;           // Byte 5, bit 1, bool. 0 corresponds to disable
        discharge
123 int speedModeEnable = 0;             // Byte 5, bit 2, bool. 0 corresponds to staying in
        torque mode
124 int commandedTorqueLimit = 0*10;        // Bytes 6, 7. 0 corresponds to using EEPROM limit
125 // Define and initialize the rolling counter (assuming it is stored somewhere)
126 static uint8_t rollingCounter = 0; // Static to retain its value across function calls
127
128 /*-----------------------------------------------------------------*/
129 // Setup Function
130 void setup() {
131   Serial.begin(9600);
132
133   // Set pin modes
134   pinMode(BSPSPRESOCSC_PIN, INPUT);
135   pinMode(BSPDPRESTHRESH_PIN, INPUT);
136   pinMode(SDSTART_PIN, INPUT);
137   pinMode(SDBOTS_PIN, INPUT);
138   pinMode(BPSF_PIN, INPUT);
139   pinMode(BPSR_PIN, INPUT);
140   pinMode(TPS1_PIN, INPUT);
141   pinMode(TPS2_PIN, INPUT);
142
143   //CAN Bus initialisation
144   NUCAN_init(numsend, numreceive);
145   commandMsgInit();
```

```
146 }
147
148 /* ------------------------------*/
149 // Main Loop
150 void loop() {
151   //Serial.println(millis());
152   NUCAN_read(outputVar, inputmsgs, numreceive);
153
154   EVERY_N_MILLIS(100) {
155     generateBrakeSig();
156     appsFlags();
157     serialOut();  // For commissioning
158     dataHandling();
159   }
160
161   EVERY_N_MILLIS(10){
162     updateThrottle();
163     sendTorque();
164   }
165   updateappsPlaus();              // DO NOT put state machine into every n millis
166   updateRTDStateMachine();
167   NUCAN_heartbeat(&HB_PEN);
168 }
169
170 /* ------------------------------*/
171
172 void updateThrottle() {
173   // APPS Signals
174   // Raw data [0-1023]
175   tps1Raw = analogRead(TPS1_PIN);
176   tps2Raw = analogRead(TPS2_PIN);
177
178   // Scaled data ~[0 - 100] - used for plausibility checking
179   tps1Scaled = (tps1Raw + TPS1_OFFSET)*TPS1_SCALE;
180   tps2Scaled = (tps2Raw + TPS2_OFFSET)*TPS2_SCALE;
181
182   // Bounded data [0 - 100] - over/undershoot removed, used for datalogging
183   tps1Bounded = applyFloorCeil(tps1Scaled, 0, 100);
184   tps2Bounded = applyFloorCeil(tps2Scaled, 0, 100);
185
186   // Build raw torque
187   appsOutRaw = (tps1Bounded + tps2Bounded)/2;
188   timeStamp = timeStamp + 1;  // For APPS tuning data
189 }
190
191 // Read and generate signals used for various stages of error checking and output generation
      and send over CAN
192 void dataHandling(void) {
193   // Current limits sent to the inverter:
194   //CCL = 30.0;  // static Charge current limit in Amps
195   //DCL = 170.0; // static Discharge current limit in Amps
196   //NUCAN_write(&BMS_Max_Discharge_Current, 2*DCL); // x2 for Inverter
197   //NUCAN_write(&BMS_Max_Charge_Current, 2*CCL);    // x2 for Inverter
```

```
198   // Ensure scaledTorqueCommand fits in the range of a signed 16-bit integer
199   if (SoC >= 95.0) {
200     CCL = 1; // Unsure if this can be set lower? 0 will not work
201   }
202   else if (SoC > 75.0) {
203     CCL = 30.0 * (1.0 - ((SoC-75.0) / 20.0 ));
204   }
205   else {
206     CCL = 30;
207   }
208   short CCL_out = (short)CCL*2;  // Convert to signed 16-bit integer
209   short DCL_out = (short)DCL*2;  // Convert to signed 16-bit integer
210   CURRENT_LIMIT.id = 0x202;      // Command Message ID, 0x202
211   CURRENT_LIMIT.len = 8;
212
213   CURRENT_LIMIT.flags.extended = 0;       // Standard Messages, Not Extended ID
214   CURRENT_LIMIT.flags.remote   = 0;       // No Remote Transmission Requests (RTR)
215   CURRENT_LIMIT.flags.overrun  = 0;       // Unsure what this does
216   CURRENT_LIMIT.flags.reserved = 0;
217
218   // Assign the values to the buffer in little-endian format
219   CURRENT_LIMIT.buf[0] = DCL_out & 0xFF;           // Low Byte (LSB)
220   CURRENT_LIMIT.buf[1] = (DCL_out >> 8) & 0xFF;    // High Byte (MSB)
221   // Assign the values to the buffer in little-endian format
222   CURRENT_LIMIT.buf[2] = CCL_out & 0xFF;           // Low Byte (LSB)
223   CURRENT_LIMIT.buf[3] = (CCL_out >> 8) & 0xFF;    // High Byte (MSB)
224   CURRENT_LIMIT.buf[4] = 0;
225   CURRENT_LIMIT.buf[5] = 0;
226   CURRENT_LIMIT.buf[6] = 0;
227   CURRENT_LIMIT.buf[7] = 0;
228   NUCAN_direct_write(CURRENT_LIMIT);
229
230   // CAN bounded data - datalogging purposes
231   NUCAN_write(&TPS1_SCALED, tps1Bounded);
232   NUCAN_write(&TPS2_SCALED, tps2Bounded);
233   NUCAN_write(&APPS_OUT_RAW, appsOutRaw);
234   NUCAN_write(&APPS_OUT_GATED, appsOutGated);
235
236   // BPS Signals
237   // Raw data [0-1023]
238   bpsFRaw = analogRead(BPSF_PIN);
239   bpsRRaw = analogRead(BPSR_PIN);
240
241   // Scaled data ~[0 - 2000] - used for brake signal
242   bpsFScaled = bpsFRaw*2.4438-250; // [psi] assuming 0.5-4.5V maps to 0-2000 PSI Linearly
243   bpsRScaled = bpsRRaw*2.4438-250; // [psi] assuming 0.5-4.5V maps to 0-2000 PSI Linearly
244
245   // Bounded data [0 - 2000] - over/undershoot removed, used for datalogging
246   bpsFBounded = applyFloorCeil(bpsFScaled, 0, 2000);
247   bpsRBounded = applyFloorCeil(bpsRScaled, 0, 2000);
248
249   // CAN bounded data - datalogging purposes
250   NUCAN_write(&BRKPRS_F, bpsFBounded);
```

```
251   NUCAN_write(&BRKPRS_R, bpsRBounded);
252
253   // Shutdown Circuit State
254   // Raw data [0-1]
255   sdStart = digitalRead(SDSTART_PIN);
256   sdBots = digitalRead(SDBOTS_PIN);
257
258   // CAN data - datalogging purposes
259     //NUCAN_write(&SD_START, sdStart);
260   NUCAN_write(&SD_BOTS, sdBots);
261
262   // BSPD States
263   // Raw data [0-1]
264   bspdPressureOCSC = digitalRead(BSPSPRESOCSC_PIN);
265   bspdPressureThresh = digitalRead(BSPDPRESTHRESH_PIN);
266
267   // CAN data - datalogging purposes
268   NUCAN_write(&BSPD_PRESSURE_OCSC, bspdPressureOCSC);
269   NUCAN_write(&BSPD_PRESSURE_THRESH, bspdPressureThresh);
270 }
271
272 /* ------------------------------*/
273
274 // Apply a max and min to input variable and bound accordingly
275 float applyFloorCeil(float var, float minimum, float maximum) {
276   if (var < minimum) {
277     var = minimum;
278   } else if (var > maximum) {
279     var = maximum;
280   }
281   return var;
282 }
283
284 /* ------------------------------*/
285
286 // Determine whether braking based on front brake pressure
287 void generateBrakeSig(void) {
288   // Generate the brake signal
289   if ((bpsFScaled > BRAKETHRESHOLDFRONT) || (bpsRScaled > BRAKETHRESHOLDREAR)) {
290     brkSig = 1;
291   } else {
292     brkSig = 0;
293   }
294   // send CAN message
295   NUCAN_write(&BRK_SIG, brkSig);
296 }
297
298 /* ------------------------------*/
299
300 // Function to update the RTD state machine
301 void updateRTDStateMachine() {
302     switch (rtdStateMachine) {
303         case STATE_INIT:
```

```
304           inverterRTD = 0;
305           inverterEnable = 0;
306           if ((int)rtdState == 1 && (int)sdTSMS == 1) {
307               stateChangeTime = millis();  // Set the timestamp when rtdState becomes 1
308               rtdStateMachine = STATE_RTD_WAIT;
309           }
310           break;
311         case STATE_RTD_WAIT:
312           inverterEnable = 1;
313           // Wait until 3 seconds have passed since rtdState was set to 1
314           if (((int)rtdState == 1 && (int)sdTSMS == 1) && (millis() - stateChangeTime >=
                  3000)) { // Perform the logic after 3 seconds
315             rtdStateMachine = STATE_RTD;  // Move to RTD state
316           }
317           // Reset if rtdState is no longer 1
318           else if ((int)rtdState != 1 || (int)sdTSMS != 1) {
319             rtdStateMachine = STATE_INIT;
320           }
321           break;
322         case STATE_RTD:
323           inverterRTD = 1;
324           if ((int)rtdState != 1 || (int)sdTSMS != 1) {
325             rtdStateMachine = STATE_INIT; // Go back to the initial state
326           }
327           break;
328     }
329 }
330
331 /* ------------------------------*/
332
333 // Plausibility state machine
334 void updateappsPlaus(void) {
335     switch (state) {
336       case STATE_PLAUS:
337         appsPlaus = 1;
338         if (checkPlausible() == 0) {
339           state = STATE_WAIT;
340           implausStart = millis();
341         }
342         break;
343       case STATE_WAIT:
344         if (checkPlausible() == 1) {
345           state = STATE_PLAUS;
346         } else if ((millis() - implausStart) >= 100) {  // 0.1 second timer
347           state = STATE_IMPLAUS;
348         }
349         break;
350       case STATE_IMPLAUS:
351         appsPlaus = 0;
352         plausFlag = plausFlag + 1;
353         if (checkPlausible() == 1) {
354           state = STATE_PLAUS;
355         }
```

```
356         break;
357     }
358 }
359 
360 /* ------------------------------*/
361 
362 // Generate trail flag and send both trail and plaus over CAN
363 void appsFlags(void){
364 
365   // Plausibility
366   // NOTE: Plausibility updated asynchronously with state machine as it requires a timer
367   //Send state over CAN - datalogging
368   NUCAN_write(&APPS_PLAUS, appsPlaus);
369 
370   // Trail braking
371   // clear the flag if throttle below threshhold
372   if (appsOutRaw <= 5) {
373     appsTrail = 1;
374   } else if ((brkSig==1) && (appsOutRaw > ZERO_TORQUE_APPS)) {
375     appsTrail = 0;
376   }
377   //Send state over CAN - datalogging
378   NUCAN_write(&APPS_TRAIL, appsTrail);
379 }
380 
381 /* ------------------------------*/
382 
383 int checkPlausible(void) {
384   int plausCheck;
385   if ((tps1Scaled < -5) || (tps1Scaled > 120) || (tps2Scaled < -5) || (tps2Scaled > 120) ||
       (abs(tps1Scaled-tps2Scaled)>10)) { //10 for comp
386     plausCheck = 0;
387   } else {
388     plausCheck = 1;
389   }
390   return plausCheck;
391 }
392 
393 /* ------------------------------*/
394 
395 void commandMsgInit(void) {
396   commandMessage.id = COMMAND_MSG_ID;      // Command Message ID, 0x0C0
397   commandMessage.len = 8;
398 
399   commandMessage.flags.extended = 0;       // Standard Messages, Not Extended ID
400   commandMessage.flags.remote   = 0;       // No Remote Transmission Requests (RTR)
401   commandMessage.flags.overrun  = 0;       // Unsure what this does
402   commandMessage.flags.reserved = 0;
403 
404   commandMessage.buf[0] = 0;
405   commandMessage.buf[1] = 0;
406   commandMessage.buf[2] = 0;
407   commandMessage.buf[3] = 0;
```

```
408   commandMessage.buf[4] = directionCommand;
409   commandMessage.buf[5] = (speedModeEnable << 2) + (inverterDischarge << 1) + inverterEnable
          ;
410   commandMessage.buf[6] = 0;
411   commandMessage.buf[7] = 0;
412 }
413
414 /* ------------------------------*/
415
416 void sendTorque(void) {
417   appsOutGated = appsOutRaw * appsTrail * appsPlaus * inverterRTD; // turned off for brake
          sensor issues
418
419   // Anti-creep functionality
420   if (appsOutGated <= APPS_FLOOR) {
421     appsOutGated = 0;
422   }
423
424   // Torque limit
425   if (appsOutGated > APPS_CEIL) {
426     appsOutGated = APPS_CEIL;
427   }
428
429   // Vehicle speed calculation
430   if (motorSpeed > 10000) {
431     motorSpeedRPM = 65536 - motorSpeed; // Unsigned bit rollover?
432   } else {
433     motorSpeedRPM = -motorSpeed;
434   }
435   speedKph = (motorSpeedRPM * 60 / (46/14)) * (3.14 * 16 * 0.0254 / 1000); // WheelsRevs/
          Hour * Kilometres/Rev
436
437   if (inverterRTD == 0) {
438     scaledTorqueCommand = 0;
439   } else {
440     // Calculate scaledTorqueCommand based on appsOutGated
441     if (appsOutGated >= ZERO_TORQUE_APPS) {
442       // Positive torque
443       scaledTorqueCommand = ((appsOutGated - ZERO_TORQUE_APPS) / (APPS_CEIL -
              ZERO_TORQUE_APPS)) * MAX_MOTOR_TORQUE * 10;
444     } else if (appsOutGated < ZERO_TORQUE_APPS && appsTrail != 0) {
445       // Regenerative braking torque
446       if (speedKph > 5) { // Compliance rules
447         scaledTorqueCommand = ((appsOutGated - ZERO_TORQUE_APPS) / ZERO_TORQUE_APPS) *
              MAX_REGEN_TORQUE * 10;
448
449       } else {
450         scaledTorqueCommand = 0;
451       }
452     } else {
453       // No torque [ONLY OCCURS IF APPS DIES]
454       scaledTorqueCommand = 0;
455     }
```

```
456   }
457   commandedTorqueLimit = MAX_MOTOR_TORQUE*10;
458   // Convert motor speed from RPM to rad/s
459   motorSpeedRadPerSec = motorSpeedRPM * 2.0 * PI / 60.0;
460   if (motorSpeedRadPerSec > 0) {
461     maxTorque = MAX_POWER*efficiency/motorSpeedRadPerSec;
462     requiredCurrent = (scaledTorqueCommand * motorSpeedRadPerSec) / packVoltage;
463     if (scaledTorqueCommand > maxTorque) { // Maximum power torque reduction
464       commandedTorqueLimit = (maxTorque) * 10;
465     }
466     else if (requiredCurrent > DCL) { // Maximum power current condition
467       //scaledTorqueCommand = (DCL/requiredCurrent)*scaledTorqueCommand;
468       commandedTorqueLimit = (packVoltage*DCL/motorSpeedRadPerSec) * 10; // Torque = Power/
              speed
469     }
470   }
471   if (inverterRTD == 0) {
472     commandedTorqueLimit = 0;
473   } else if (commandedTorqueLimit > MAX_MOTOR_TORQUE*10) {
474     commandedTorqueLimit = MAX_MOTOR_TORQUE * 10;
475   }
476
477   // Increment the rolling counter and wrap around to 0 after 15
478   rollingCounter = (rollingCounter + 1) & 0x0F; // 0x0F ensures rollingCounter stays within
      0-15
479
480   // Pack the control bits for inverterRTD, inverterDischarge, and speedModeEnable
481   uint8_t controlBits = 0;
482   if (speedModeEnable) {
483     controlBits |= (1 << 2); // Set bit 2 for speedModeEnable
484   }
485   if (inverterDischarge) {
486     controlBits |= (1 << 1); // Set bit 1 for inverterDischarge
487   }
488   if (inverterEnable) {
489     controlBits |= (1 << 0); // Set bit 0 for inverterRTD
490   }
491
492   // Manually update the rollingCounter and controlBits in buf[5]
493   // First, clear the upper 4 bits (4-7) and lower 3 bits (0-2) of buf[5]
494   commandMessage.buf[5] = 0x00; // Start with a clean byte
495
496   // Set the rolling counter in bits 4-7
497   commandMessage.buf[5] |= (rollingCounter << 4); // Set bits 4-7
498
499   // Set the control bits in bits 0-2
500   commandMessage.buf[5] |= controlBits; // Set bits 0-2
501
502   // Ensure scaledTorqueCommand fits in the range of a signed 16-bit integer
503   short signedTorqueCommand = (short)scaledTorqueCommand;  // Convert to signed 16-bit
      integer
504
505   // Assign the values to the buffer in little-endian format
```

```
506   commandMessage.buf[0] = signedTorqueCommand & 0xFF;           // Low Byte (LSB)
507   commandMessage.buf[1] = (signedTorqueCommand >> 8) & 0xFF;    // High Byte (MSB)
508
509   // Ensure scaledTorqueCommand fits in the range of a signed 16-bit integer
510   short signedTorqueLimit = (short)commandedTorqueLimit;  // Convert to signed 16-bit
          integer
511
512   // Torque limits
513   commandMessage.buf[6] = signedTorqueLimit & 0xFF;           // Low Byte (LSB)
514   commandMessage.buf[7] = (signedTorqueLimit >> 8) & 0xFF;    // High Byte (MSB)
515
516   NUCAN_direct_write(commandMessage);
517 }
518 /* ------------------------------*/
519
520 /* ------------------------------*/
521
522 void serialOut(void){
523     Serial.println("---------------");
524
525     Serial.print("appsOutRaw = "); Serial.print(appsOutRaw);Serial.print("Apps Gated = ");
          Serial.print(appsOutGated);Serial.print(". APPS Trail OKHS = "); Serial.print(
          appsTrail); Serial.print(". APPS Plaus OKHS = "); Serial.println(appsPlaus);
526     Serial.print("Brake Pressures: Front = "); Serial.print(bpsFScaled); Serial.print(" psi.
          Rear = "); Serial.print(bpsRScaled); Serial.println(" psi.");
527     Serial.print("Brake Signal = "); Serial.println(brkSig);
528     Serial.print("TPS Sensors: Sensor 1 = "); Serial.print(tps1Scaled); Serial.print("%.
          Sensor 2 = "); Serial.print(tps2Scaled); Serial.println("%.");
529     Serial.print("TPS DIFF: "); Serial.print(tps1Scaled - tps2Scaled); Serial.println("%.");
530     Serial.print("Torque Scaled = "); Serial.print(scaledTorqueCommand); Serial.print("Limit
          Scaled = "); Serial.println(commandedTorqueLimit);
531     Serial.print("APPS Trail OKHS = "); Serial.print(appsTrail); Serial.print(". APPS Plaus
          OKHS = "); Serial.print(appsPlaus); Serial.print(". Plaus Flag "); Serial.println(
          plausFlag);
532     Serial.print("BSPD OCSC OKHS = "); Serial.print(bspdPressureOCSC); Serial.print(". BSPD
          THRESH OKHS = "); Serial.println(bspdPressureThresh);
533     Serial.print("SD START OKHS = "); Serial.print(sdStart);
534     Serial.print(". SD BOTS OKHS = "); Serial.println(sdBots);
535     Serial.print("SD TSMS OKHS = "); Serial.print(sdTSMS);
536     Serial.print(". rtdState = "); Serial.println(rtdState);Serial.print("Inverter Enable =
          "); Serial.println(inverterEnable);
537     Serial.print("CAN Inverter Motor Speed = "); Serial.println(motorSpeed);
538     Serial.print("Inverter Motor Speed = "); Serial.print(motorSpeedRPM); Serial.print(" rpm
          . Wheel speed = "); Serial.print(speedKph); Serial.println(" kph.");
539     Serial.print("SOC%: "); Serial.println(SoC);
540     Serial.print("DCL = "); Serial.print(DCL);Serial.print(". CCL = "); Serial.println(CCL);
541 }
```

J. Cascadia Key Information Document

Cascadia CM200DZ Key Information Document

Written by Josh Hayward
for NU Racing 2024
Version 0.4 – 26/03/2024

Table of Contents

Setup without HV: ***(Pg.3 – Start Guide, Cascadia Master File)***

- Create directory structure on computer hard drive. *(Pg.12 – Software)*
- Wiring for 12V Power, RS232, CAN (optional) and program switch to RMS unit.
- RMS GUI; check firmware version, and date-code. Save EEPROM data as "EEPROM Factory Data".
- Program up-to-date firmware to ensure Program-enable switch and C2Prog application are both working.

Setup requiring HV: ***(Pg.4 – Start Guide, Cascadia Master File)***

- Connect HV Cabling to the inverter from DC and motor phase cables.
- Program the correct motor type in EEPROM.
- Resolver calibration for PM motors, not necessary for induction motors.
- Select EEPROM settings for your system.

Functional Overview: ***(Pg.4 – Hardware, Cascadia Master File)***

By default, parameters are setup in Torque Control Mode. These parameters must be changed to match the load motor and operating characteristics before running for the first time.

Liquid Cooling Connections: ***(Pg.7 – Hardware, Cascadia Master File)***

- **Coolant Type:** 50/50 Mix ethylene glycol.
- **Coolant Temperature:** -30C to 45C full power, 45C to 80C de-rated output.
- **Coolant Flow Rate:** 12 LPM minimum.
- **Pressure Drop:** 0.3bar (4.3psi) @ 12 LPM @ 25C.
- **Port Size:** SAE ORB -06, comes with 5/8" hose and -8 AN fitting.

For proper operation of the inverter, the coolant must flow at a rate equal to or above the minimum specified flow rate at all times that the motor is enabled. It is possible to adjust the fan speed on the coolant radiator as needed depending on the operating conditions of the inverter. The best practice is to measure the actual coolant flow after the system has been assembled. The power module temperature sensors are located in such a way that they are much closer to the temperature of the coolant than they are to the temperature of the transistors and diodes used inside the power module.

CM200 Signal Connector: ***(Pg.25 – Hardware, Cascadia Master File)***

A Molex CMC 48-way connector is used for all low voltage signals.

- **Mating Housing:** Molex 64320-3311
- **Strain Relief:** Molex 64320-1301
- **Contact, CP 0.6, 0.5mm2/20AWG, Wire OD 1.4-1.7mm:** Molex 64322-1239
- **Contact, CP 0.6, 0.75mm2/18AWG, Wire OD 1.4-1.7mm:** Molex 64322-1219
- **Contact, CP 1.5, 0.5-1mm2/18AWG, Wire OD 1.4-2.15mm:** Molex 64323-1319
- **Blind plug, CP 0.6:** Molex 643251010
- **Blind plug, CP 1.5:** Molex 643251023

Note: *A1 – K4 are CP 0.6 size, L1 – M4 are CP 1.5 size*

1

External Power Connections: *(Pg.29 – Hardware, Cascadia Master File)*

DC/DC wires:

DC/Battery Power is located on the REAR/FRONT? The DC power must be run through an external pre-charge circuit to safely charge the capacitors inside the controller before the main contactor engages. The CM200 uses Rosenberger HPK family connectors for making connection to the DC bus input of the inverter.

DANGER: *Before changing the wiring make sure that the internal DC bus capacitors are discharged. The voltage should be measured at the terminals before disconnecting. If there is any doubt about the safety wait at least 1 hour after power has been removed before touching the terminals.*

Motor Phase wires:

Phase A, Phase B and Phase C are wired to the motor such that they give the proper direction of rotation. They are most likely to generate high EMI and carry a higher average current than the DC wires. They should be kept as short as possible and shielded to the chassis of the inverter and chassis of the motor. The CM200 uses Rosenberger HPK connectors for the AC connections.

Pre-Charge Circuit:

An external pre-charge circuit must be used with the controller. The pre-charge circuit adds a resistor, relay and fuse in parallel with the main contactor. When the controller is powered on the controller will first engage the pre-charge relay to charge the capacitors internal to the controller. If the capacitors charge properly then the main contactor will engage. The pre-charge resistor needs to be sized to rapidly charge the capacitor, but not dissipate too much power in a fault condition It must be sized that if the controller had a short on its input the pre-charge resistor would not fail. The pre-charge relay will only remain closed for about 3 seconds. The pre-charge sequence must complete before this time or the inverter will declare a fault condition, opening the pre-charge relay. The pre-charge circuit should be fused with a small fuse appropriate to the wire used. Below 5 amps is typical, hence 18 AWG and a 5 amp fuse would be suitable.

Model	Internal Capacitance	Maximum Pre-charge resistor	Cascadia Motion Part Number
CM200DZ	255uF	2000 Ohms	53-0008

(A sizing example is provided on Pg.32 of the Hardware document, consult Electrical Engineer for calculations.)

Main Contactor:

The main contactor is the switching element between DC High Voltage and the controller. It must be sized to handle the operating currents of the controller. It must be able to open under a fault condition. It may be in series with the positive path from the battery to the controller. *(An example is the Gigavac GX14BA, Cascadia Motion p/n 77-0035).* It must be rated to handle DC voltage and AC-only rated contactors and relays must not be used. Note that DC rated contactors are usually polarity sensitive.

Main Fuse:

The DC input to the controller must be fused. This fuse must be rated for the voltage of the battery and must open under the short circuit current the battery may produce. This may be part of the battery pack, if not, a semiconductor type fuse is recommended.

2

Passive Discharge of the High Voltage DC Bus:
The inverter contains a large amount of DC bus capacitance. If provisions have not been made for discharging these capacitors, they should not be touched for at least 5 minutes after the high voltage has been disconnected.

Model	Passive Discharge Resistance	DC Link Capacitance	3 Time Constants	Y - Capacitance
CM200DZ	70K ohms	255uF	54 s	136nF

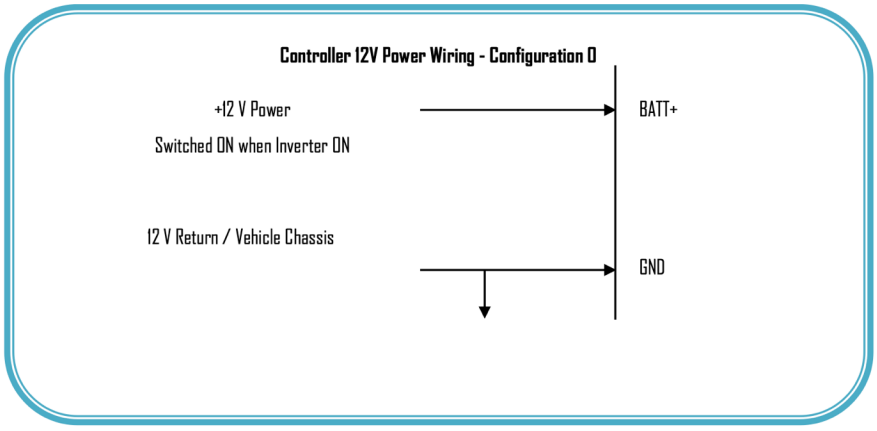
The 3 time constants represents the value at which the voltage is less than 5% of its initial value. At this point it should be safe to touch. The passive discharge resistance is connected to the high voltage DC bus at all times. For example. If the CD200DZ is being used at 450V it would draw 450/70K = 6.4mA even when the inverter is disabled. The Y-Capacitance is the total amount of capacitance connected between the DC bus and the chassis of the inverter.

12V Power:
The switched 12V power is connected to the BATT+ terminals. The ground return for the 12V power is connected to the GND terminals. For normal applications only one pin is necessary.

Grounding:
The inverter housing has a location for connecting the case to ground. The inverter housing must be connected to the motor case. It must also be connected to the vehicle chassis, and this assumes that the vehicle chassis is at the same potential as the 12V GND.

Controller 12V Power Wiring: *(Pg.39 – Hardware, Cascadia Master File)*

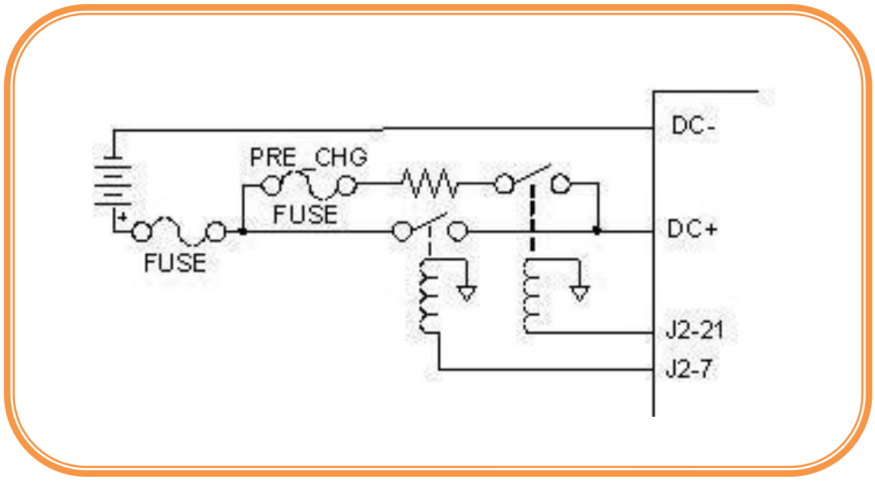
Simple ON/OFF Configuration. Using an external switch, the 12V Power is supplied to the controller. This has a less controlled shutdown process as power could be removed whilst motor is actively being controlled.



(When using this configuration set the EEPROM parameter Key_Switch_Mode_EEPROM to 0)

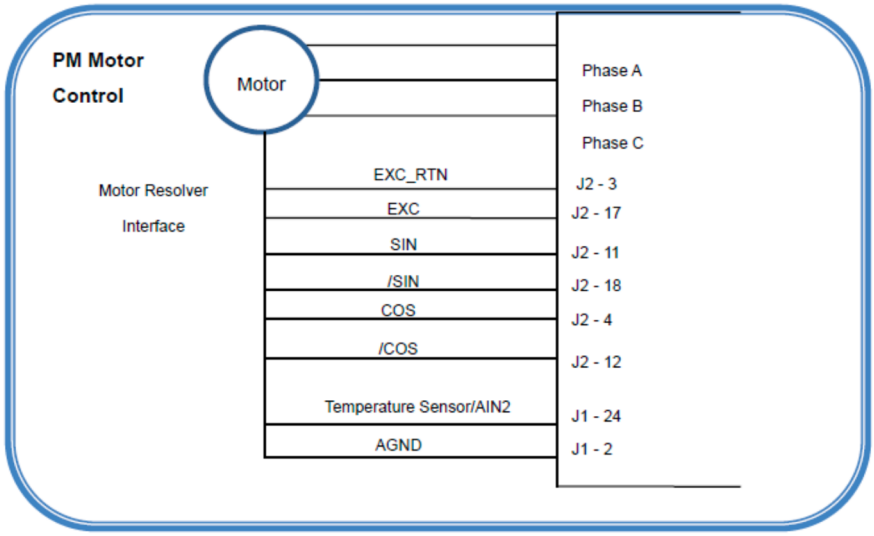
3

Pre-charge Circuit: ***(Pg.41 – Hardware, Cascadia Master File)***



(The connections shown are from the PM100/PM150)

PM Motor Control: ***(Pg.43– Hardware, Cascadia Master File)***



(The connections shown are from the PM100/PM150)

4

CAN Interface: *(Pg.44 – Hardware, Cascadia Master File)*

The controller has one active CAN interface *(CAN A)*. It has multiple purposes:

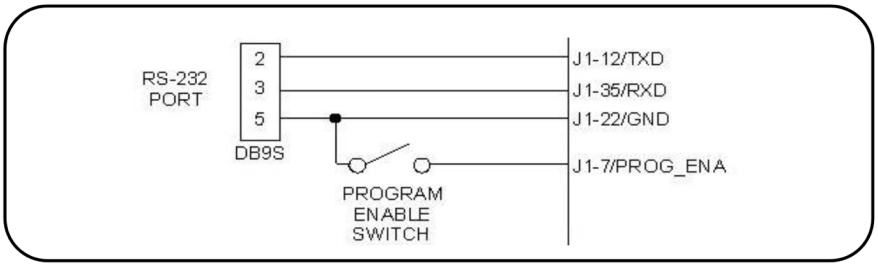
- Provides direct control of the motor.
- Provides diagnostic and monitoring capabilities.
- Provides user-adjustable configuration.

The user can change the following hardware related configuration parameters.

- Inverter Command Mode: Setting this parameter to 1 allows the CAN mode to become active.
- CAN Bus Speed: Allowed speeds are 125 Kbps, 250 Kbps, 500 Kbps or 1 Mbps. Enter the respective number to program the configuration parameter.
- CAN Terminator Resistor *(Potentially PM Family Only).*

RS-232 Interface: *(Pg.44 – Hardware, Cascadia Master File)*

The RS-232 serial interface is used to set up and tune the controller, as well as downloading controller software updates from a PC. RMS GUI is a simple Windows based software package used for monitoring and changing parameters. The drive can also be placed in data-logging mode and used with a PC or other serial device to broadcast datasets at 3Hz of a number of parameters including performance and energy consumption data.
(For more information, refer to SCI Data Acquisition manual).

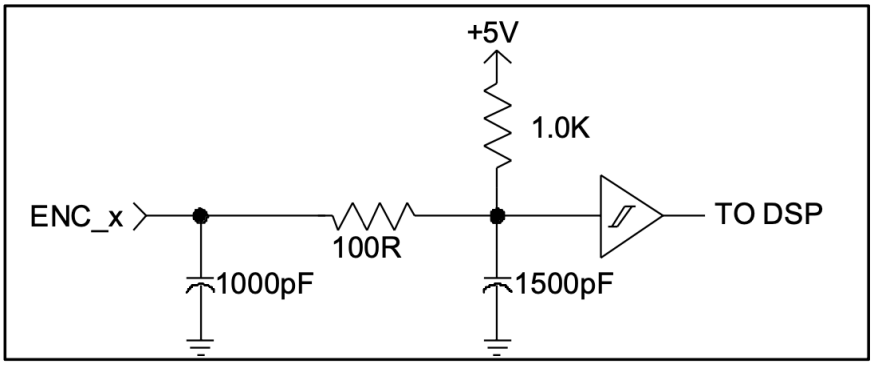


(The connections shown are from the PM100/PM150)

5

Encoder Interface: ***(Pg. 45 – Hardware, Cascadia Master File)***

The controller provides a 5V interface to power the external encoder and to receive, level translate, and filter the signals from A, B and INDEX channels. It is connected internally to the TI DSP QEP Module (Quadrature Encoder Peripheral). This has special hardware for wide dynamic range speed and angle calculation from the encoder data. The drive has internal pull-up resistors for these inputs and works with encoders that have either bi-polar or open-collector outputs.



Resolver Interface: ***(Pg.46 – Hardware, Cascadia Master File)***

A resolver is a position sensor often used with Permanent Magnet type motors. It requires an excitation voltage and provides a SIN and COS feedback. The CM200 inverters have an excitation frequency of 10kHz that is not synchronised with the PWM frequency. There is a dedicated Resolver to Digital Converter (RDC). The target SIN/COS voltage is dependent on how the RDC is configured.

Analog and Digital Inputs: ***(Pg.47 – Hardware, Cascadia Master File)***

The CM200 has 4 available 0-5V analog inputs which are assigned by default as follows:

1. AIN1 – ACCEL
2. AIN2 – Motor Thermistor
3. AIN3 – BRAKE
4. AIN4 – Unassigned by default

AIN1, AIN2 and AIN3 have programmable 1K pull-up resistors. AIN4 has an always active 10K pull-down resistor.

There are also 4 digital inputs which are Switch To Ground type (STG).

1. DIN1 – FWD_ENA
2. DIN2 – REV_ENA
3. DIN3 – BRAKE
4. DIN4 – REGEN Disable

More information on these can be found on Pg.47 and Pg.48 of the Hardware manual.

Digital Outputs: ***(Pg.53 – Hardware, Cascadia Master File)***

The CM200 has 4 available digital outputs.

1. RLY1 (HSD) – PRECHARGE DRIVE
2. RLY2 (HSD) – MAIN DRIVE
3. RLY3 (LSD) – OK INDICATOR
4. RLY4 (LSD) – FAULT INDICATOR

Firmware: ***(Pg.6 – Software, Cascadia Master File)***

The firmware is a single file in hexadecimal format which can be downloaded and programmed into the inverter over the serial port. CM200 Gen 5 firmware follows the filename convention 'CM_GEN5_yyyymmdd_nnnn_option.hex'. The 'option' can be either :

1. 'Group_1' – motor types 0-59
2. 'Group_2' – motor types 60 and onwards.

The Firmware can be found online at www.cascadiamotion.com in the Documentation/Support page.
To upload new firmware to the controller, we use the RS-232 serial port and C2Prog.

Serial Communication Interface: ***(Pg.8 – Software, Cascadia Master File)***

The serial communication interface (SCI) has 3 purposes:

- Firmware download.
- Graphic User Interface (GUI) communication.
- SCI Data acquisition.

On regular powerup the inverter enters SCI data acquisition mode. SCI data is transmitted in hexadecimal format where it can be captured on a PC using Real-term. This data can be used to plot graphs and understand vehicle performance. When RMS GUI is started the inverter will automatically switch to communicating with the GUI. From here the GUI can be used to reprogram EEPROM parameters and monitor data using Windows platform. It will automatically search all open COM ports for the inverter. It is important that the COM port is not being used by another software. If the GUI is not able to find a defsyms (symbol defining active parameters for the firmware) with a date code that matches the inverter firmware date code it will flag an error.

RMS GUI: ***(Pg.9 – Software, Cascadia Master File)***

RMS GUI must be installed upon the first use. This is done by running the GTK environment (gtk+-2.8.9-setup-1.exe) file. After this the PC must be rebooted. The RMS GUI will automatically setup its baud rate to suit the inverter COM port. Whenever loading or saving a file using the RMS GUI please locate that file in the same folder that the RMS GUI.exe exists in. It will create a 'conf' folder that contains a record of any EEPROM changes that have been made.

7

C2Prog: *(Pg. 10 – Software, Cascadia Master File)*

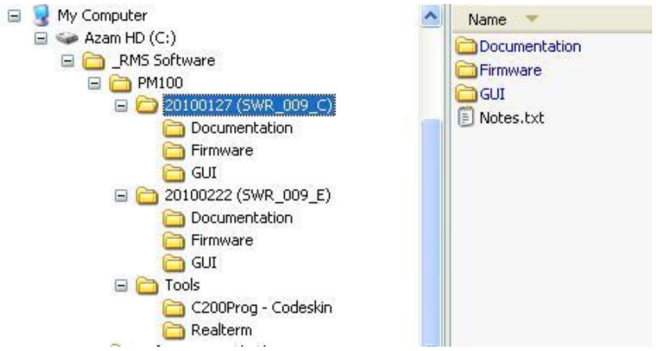
C2Prog is a flash-programming tool that utilises the RS-232 and CAN protocols. It uses the boot=loader feature of the MCU for rapid flash programming over the serial line. The inverters are designed to be reflashed using C2Prog and the SCI Port. The inverters will enter a boot mode in the Program Enable input is grounded when low voltage power is applied to the inverter. Reflashing the inverter should not change any EEPROM settings (if firmware is from the same family). It is good practise to save your existing EEPROM settings before reflashing an inverter.

Realterm: *(Pg.10 – Software, Cascadia Master File)*

Realterm is a terminal program specifically designed for capturing, controlling and debugging binary and other data streams. It is especially useful for bench testing the inverter. Realterm should be configured for 57600 baud rate, 8 data bits, no parity.

Firmware Folder Structure: *(Pg.12 – Software, Cascadia Master File)*

The suggested folder structure for the RMS GUI and Cascadia software is as follows:

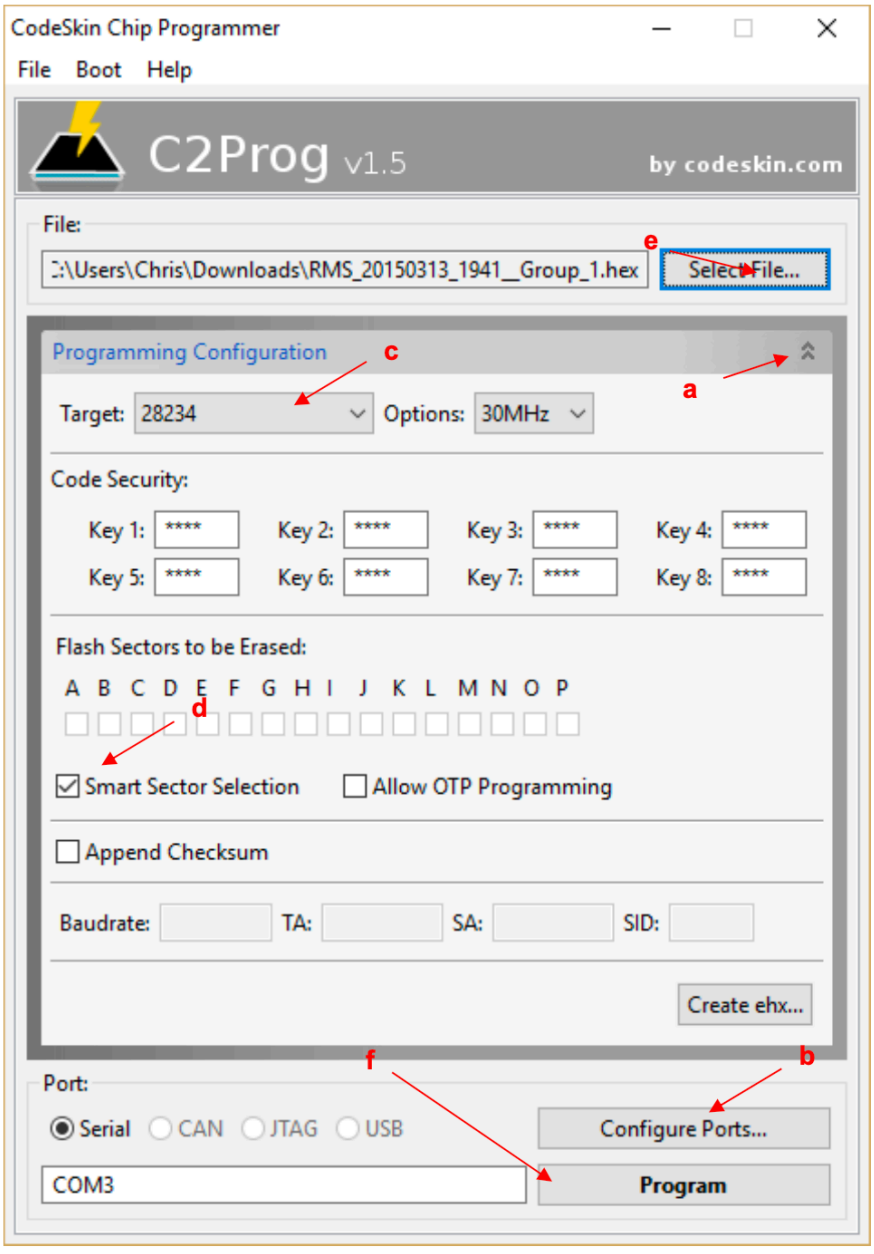


C2Prog Firmware Programming Guide: *(Pg.14 – Software, Cascadia Master File)*

First connect the RS-232 Serial connector between the motor inverter and turn on low voltage power. Then complete the following:

a) Expand the “Programming Configuration” tab.
b) Click “Configure Ports” then “Scan Ports” to select the correct COM port.
c) Pull-down the “Target” menu and select the correct target.
(28375,7,9D-CPU01 Options: 20MHz (Gen 5 / CM200)
d) Select “Smart Selector Selection” box.
e) Click “Select File…” and browse to current firmware file. *(This will have a .hex extension)*
f) Turn off the low voltage power to the inverter. Ground the Program Enable input to the inverter. Turn on the lower voltage power to the inverter.
g) Click the “Program” Button
h) Programming will begin.

i) When the programming is completed, click OK to close the status screen. Turn off the low voltage power to the inverter. Disconnect the Program Enable input from ground. When the inverter is repowered, the new firmware will be operational.



9

SCI Data Acquisition Guide: ***(Pg. 18 – Software, Cascadia Master File)***

Open Realterm and configure the software for the COM port for the inverter with the following RS232 settings:

Baud Rate	57600
Parity	None
Data Bits	8
Stop Bits	1
Hardware Flow Control	None

Data Acquisition Parameters:

Count	Parameter
1	The low word of the Power On Timer (increments every 3ms)
2	Filtered Accel-pot input voltage (V) times 100
3	Motor Torque feedback (Nm) times 10
4	Vehicle Torque Command (Nm) times 10
5	DC Voltage (V) times 10
6	DC Current (V) times 10
7	Motor Speed (rpm)
8	Flux Weakening Regulator Output (Apk) times 10
9	Motor Voltage Magnitude (Vpk) times 10
10	IQ Command (Apk) times 10
11	IQ Feedback (Apk) times 10
12	ID Command (Apk) times 10
13	ID Feedback (Apk) times 10
14	Modulation times 10000
15	Module A Temperature (C) times 10
16	Motor Temperature (C) times 10
17	Run Fault Low Word
18	Run Fault High Word
19	Torque Shudder (Nm) times 10
20	Filtered Brake pot (V) times 100

Once the data is captured in a text file, import it into an Excel spreadsheet as space delimited data. After importing all data, it can be copied into *SCI Template.xls* spreadsheet which provides conversion formulae for each data record and allows the user to plot graphs to analyse the vehicle performance in more detail.
(For information of the Data Formats visit Pg. 20 – Software, Cascadia Master File)

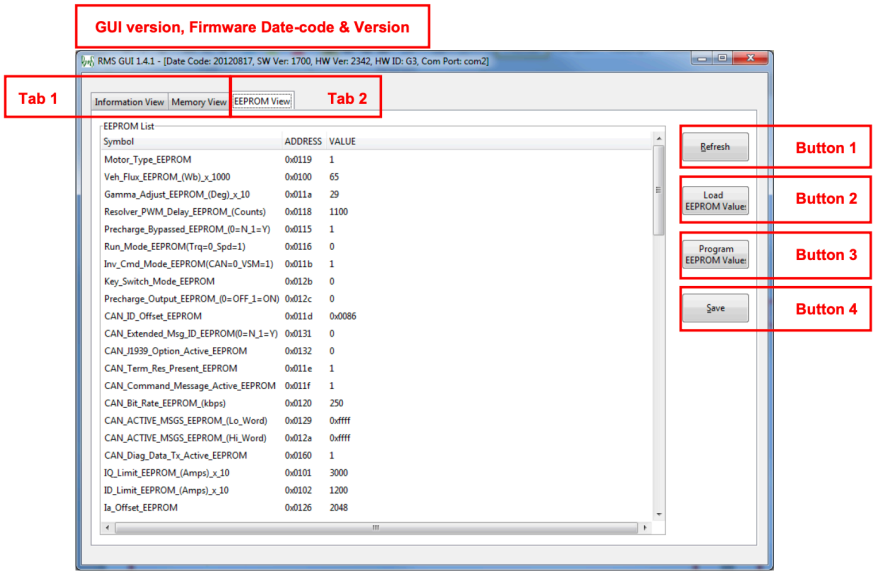
RMS GUI EEPROM Parameters Guide: *(Pg. 22 – Software, Cascadia Master File)*

RMS GUI provides the ability to program certain EEPROM parameters. These are specific to each motor and system setup with the Cascadia. EEPROM parameters **MUST** be configured correctly before the controller is operated. Before beginning the following files are required:

- RSM GUI Application
- Default symbols file *(e.g. defsyms_yyyymmdd.txt)*
- Firmware file

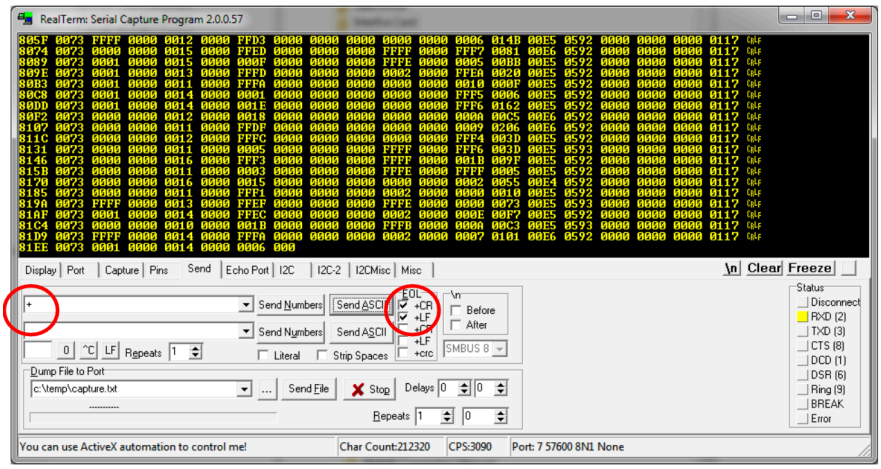
Programming Procedure:

a) Start the GUI. Confirm firmware date code and version on the title of GUI window.
b) Click on EEPROM view (Tab 2) to display all EEPROM parameters that can be programmed by the user.
c) To change a value, click on the value under the VALUE column, enter a new value and hit ENTER.
d) When finished, click Program EEPROM (Button 3)
e) Status message will confirm the programming completion. Follow any instructions presented.



EEPROM values can be saved using the Save button (Button 4).
EEPROM values can be loaded using the Load EEPROM values button (Button 2).

SCI Data acquisition mode can be accessed by completely shutting down the RMS GUI application. In Realterm, open the serial port and click anywhere in the window where the serial data appears. Press '+' then 'Enter' to restart broadcast. Furthermore, enter a '+' in the first box, then enable +CR and +LF and click 'Send ASCII' to send ASCII broadcast data.



EEPROM Parameter Setup: *(Pg. 25 – Software, Cascadia Master File)*

The EEPROM contains the internal parameters or 'calibrations' required for the inverter. These values must be adjusted to suit the vehicle and motor being used. These parameters will highly dictate the drivability and vehicle dynamics of the car. Parameter groupings for the EEPROM settings are found in the Software manual appendices:

- **Appendix A:** Motor Configuration Parameters
- **Appendix B:** System Configuration Parameters
- **Appendix C:** CAN Configuration Parameters
- **Appendix D:** Current Parameters
- **Appendix E:** Voltage & Flux Parameters
- **Appendix F:** Temperature Parameters
- **Appendix G:** Accelerator & Torque Parameters
- **Appendix H:** Speed Parameters
- **Appendix I:** PID Regulator Parameters
- **Appendix J:** Shudder Compensation Parameters
- **Appendix K:** Brake Parameters

Monitored Parameters View: *(Pg. 26 – Software, Cascadia Master File)*

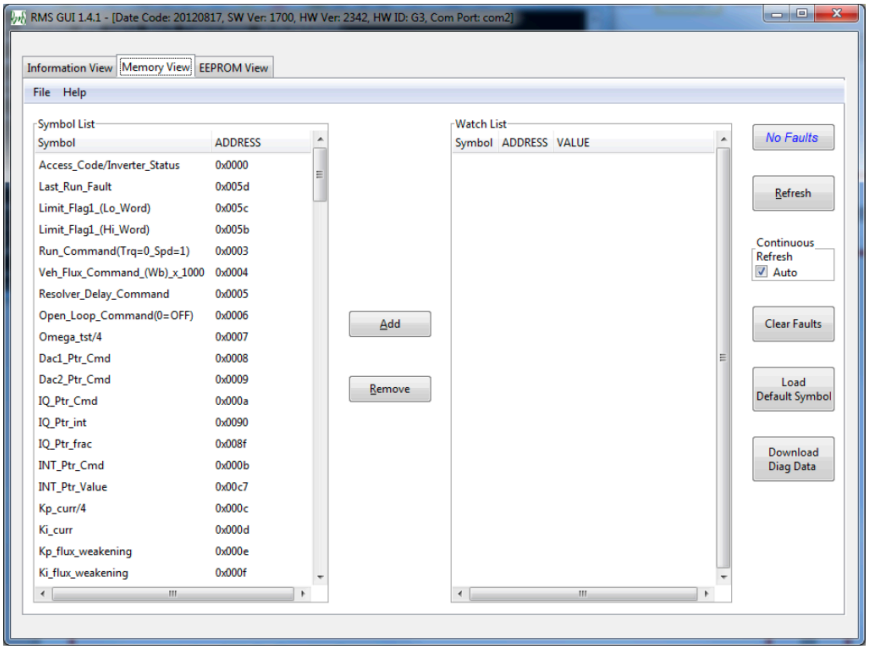
RMS GUI provides the ability to monitor several operation parameters of the controller. This helps for checking connections to the controller as well. The parameters that can be monitored are found in the Software manual appendix:

- **Appendix L:** GUI Display Parameters

1. **No Faults/Check Faults button:** Allows the user to check the fault status when 'Continuous Refresh' is set to Auto. The 'Check Faults' status indicates the presence of one or more faults. To identify faults, click this button.
2. **Clear Faults button:** This will clear all faults with the exception of those in:

- **Appendix: N**

3. **Download Diagnostic Data button:** Refer to the manual 'Download Diagnostic Data' for details. When requested, allow RMS GUI several minutes to complete the download.
4. **Load Default Symbols button:** Load the default symbols file for the firmware.



13

Calibration Process: *(Pg. 29 – Software, Cascadia Master File)*

The inverter must be calibrated properly before it can be used successfully. The following table describes the calibration processes involved with the inverter:

Calibration Process	User Manual (PDF format)	Factory Calibrated?
Current Offset	Current Offset Calculation (Only used on certain Gen 2 units)	No, not necessary
DC Voltage	DC Voltage Calibration Process (Factory calibrated)	Yes
Hall Sensor Encoder	Encoder Hall Sensor Calibration (not normally needed)	No
SIN/COS Encoder	Encoder Calibration for SIN_COS Encoder	No
Resolver	Resolver Calibration Process (Necessary for motors that use a resolver or SIN/COS Encoder)	No
RTD	RTD Calibration Process (factory calibrated thus not normally needed)	Yes

Vehicle State Machine: *(Pg. 31 – Software, Cascadia Master File)*

The drive has in internal state machine that it steps through during all stages of operation. The particular state that the drive is in can be tracked via the RMS GUI software. This is monitored via the VSM_State symbol which can take on the following values:

VSM_State	Name
0	Start State
1	Pre-charge sequence initial state – Turn on the pre-charge relay
2	Pre-charge sequence active state – Waiting for capacitor to finish charging
3	Pre-charge sequence finish state – Completes the final checks before proceeding to Wait State
4	Wait State – waiting for activation of forward or reverse
5	Ready State – Activates the inverter state machine to begin energizing the motor
6	Motor Running State – Normal motor running
7	Fault State – The controller has faulted
14	Shutdown in Process – In key switch mode 1, user has turned key switch to off position
15	Recycle Power State – This indicates that the power to the controller needs to be recycled after EEPROM Programming is complete

(For information on the fault codes visit Pg. 37 – Software, Cascadia Master File)

CAN EEPROM Parameters Overview: *(Pg. 3 – CAN Protocol, Cascadia Master File)*

Inverter Command Mode:
Gives the option to operate the inverter in VSM or CAN mode. In CAN mode, both GUI and CAN interfaces are active and can be used to monitor and modify parameters. In CAN mode the torque and speed commands come from the Command message.
0 = CAN Mode
1 = VSM Mode (Default)

CAN ID Offset:
Allows the user to set contiguous CAN message identifiers starting with the value in CAN ID Offset.

a) 11-bit CAN: Range = 0 – 0x7C0, Default = 0x0A0 (Hence, Range = 0x0A0 – 0x7C0)
b) J1939 CAN: Range = 0 – 0xC0
c) 29-bit CAN: Range = 0 – 0xFFC0

CAN Extended Message Identifier:
Allows for switching between CAN standard and extended message identifiers:

a) 0 = Standard CAN Messages (11-bit identifiers)
b) 1 = Extended CAN Messages (29-bit identifiers)

CAN J1939 Option Active:
Allows switching to the J1939 CAN format.

a) 0 = CAN ID defined as above (In CAN Extended Message Identifier)
b) 1 = Extended CAN Messages in SAE J1939 Format

CAN diagnostics are available through the RMS GUI, codes in (Pg. 5 – CAN Protocol, Cascadia Master File)

CAN Termination Resistor Present:
The CAN Termination resistor can be enabled remotely via CAN or RMS GUI software. This will be specific to the system and wiring schematic of the CAN network.

a) 0 = Term. Resistor not active
b) 1 = Term. Resistor active (Default)

The CM controllers have the ability to activate a CAN terminator, but only by making an external wiring change.

CAN Command Message Active:
The CAN Timeout feature requires a 'heartbeat' CAN Command message to be sent at some regular interval. The CAN Command message controls the inverter, motor direction and torque or speed. If a CAN command message is not received within the CAN Timeout time, the inverter will declare a Run Fault of CAN Command Timeout. The inverter will then disable the motor. If this is not enabled, when CAN Communications are lost the inverter continues to hold the last received CAN Command.

a) 0 = The CAN command message Timeout feature is disabled. The controller will hold last received CAN Command.
b) 1 = The CAN command message Timeout feature is enabled. A CAN Command message should be sent at some regular interval.

15

CAN Bit Rate:
The default bit rate is 250Kbps. Bus speed can be changed; however, this requires a power resent on the controller as bus speed is setup upon initialisation. The 4 options for baud rate are:

a) 125 = 125Kbps
b) 250 = 250Kbps (Default)
c) 500 = 500Kbps
d) 1000 = 1Mbps

CAN Active Messages Word:
Used to enabled/disable CAN Broadcast Messages. Can be of either type, CAN Active Messages (Low Word) and CAN Active Messages (High Word) in RMS GUI.

a) 0 = CAN Messages broadcast disabled
b) 1 = CAN Messages broadcast enabled (Default)

Individual messages can be enabled/disabled as seen in (Pg. 17 – CAN Protocol, Cascadia Master File)

CAN Diagnostic Data Transmit Active:
Used to enable/disable the broadcast of the diagnostic data.

a) 0 = CAN Diagnostic Data broadcast disabled
b) 1 = CAN Diagnostic Data broadcast enabled (Default)

CAN Inverter Enable Switch Active:
This can be used in CAN mode only.

a) 1 = DIN1 digital input is taken into consideration and the inverter will only be enabled if both DIN1 and inverter command are active
b) 0 = DIN1 will have no effect on enabling or disabling the inverter (Default)

CAN Timeout:
This parameter sets the CAN Timeout time. It is defined as the maximum time between the CAN Command messages that will not generate a fault. This is a multiple of 3 milliseconds. For example, the default value is 333 which is equivalent to the actual timeout value of 999ms. This parameter delays setting the CAN Timeout fault for the amount of time it represents.

CAN Slave Cmd ID:
Not implemented, see (Pg. 8 – CAN Protocol, Cascadia Master File)

Can Slave Dir:
Not implemented, see (Pg. 8 – CAN Protocol, Cascadia Master File)

CAN Fast Msg Rate / CAN Slow Msg Rate:
There are two programmable EEPROM parameters that define the CAN fast messages (Default 10ms) and CAN slow messages (Default 100ms). The software loop responsible for sending can messages run on a 3ms period. The messages will be sent out on the next 3ms increment (i.e. a 10ms increment will be sent closer to 12ms). The CAN controller will send according to the availability of the CAN bus and priority of the messages. The values:

a) CAN_Fast_Msg_Rate_EEPROM_(ms) = 100 (Default)
b) CAN_Slow_Msg_Rate_EEPROM_(ms) = 10 (Default)

CAN Diagnostic Parameters Overview: ***(Pg. 13 – CAN Protocol, Cascadia Master File)***

CAN Diagnostics are available through the RMS GUI

RMS GUI Watch Item	Description
CAN_Status_Bus_Off	0: Bus is active 1: CAN Bus has been turned off due to excessive errors. Note: The auto CAN Bus restart feature has been enabled. If the CAN bus controller gets turned off it will automatically restart.
CAN_Status_Error_Passive	0: Bus is active 1: CAN Bus controller has gone to the passive state.
CAN_Status_Error_Warning	0: Bus is active 1: The number of CAN errors has reached a warning limit of 96.
CAN_Status_Last_Error_Code	Indicates the last reported error on the CAN bus controller: 0: No Error 1: Stuff Error 2: Form Error 3: Ack Error 4: Bit 1 Error 5: Bit 0 Error 6: CRC Error
CAN_Tx_Error_Counter	Count of errors in sending of CAN messages. The maximum count is 255 before the counter loops back to 0.
CAN_Rx_Error_Counter	Count of errors in receiving CAN messages, maximum count is 127. For each correctly received CAN message the error counter will count down towards 0.

17

CAN Format: *(Pg. 14 – CAN Protocol, Cascadia Master File)*

CAN Protocol conforms to CAN 2.0A (11 bit identifiers). All messages have a data length code (DLC) of 8 bytes and follow little-endian format. This implies that the least significant byte is stored at the lowest address.

Example: A command message is setup to turn the inverter on in CAN Speed mode with a speed command of 500 RPM, the data bytes look like this.

Data Byte 0	Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7
44	1	244	1	0	1	0	0

a) Torque Command: Sent as a value in Nm x 10
30 Nm should be entered as 300 = (1 x 256) + 44
- Data Byte 0 = 44 (Low byte)
- Data Byte 1 = 1 (High byte)

b) Speed Command: Sent as a value in RPM
500 RPM is entered as 500 = (1 x 256) + 244
- Data Byte 2 = 244 (Low byte)
- Data Byte 3 = 1 (High byte)

c) Direction Command:
- Data Byte 4 = 0 (Clockwise = reverse)
- Data Byte 4 = 1 (Anticlockwise = forward)

d) Inverter Run Command:
- Data Byte 5 = 0 (Disable Inverter)
- Data Byte 5 = 1 (Enable Inverter)

Each data frame is 89 bits long.

CAN Database File

A CAN database file stores information for a given CAN network. This decodes information about CAN nodes. These files have a .dbc extension. Cascadia Motion provides a CAN database file.

18

Data Formats: ***(Pg. 15 – CAN Protocol, Cascadia Master File)***

Format	Description	Range
Temperature	Signed integer, actual temperature (in °C) times 10	-3276.8 to +3276.7 °C
Low Voltage	Signed integer, actual voltage (in Volts) times 100	-327.68 to +327.67 volts
Torque	Signed integer, actual torque (in Nm) times 10	-3276.8 to +3276.7 N-m
High Voltage	Signed integer, actual voltage (in Volts) times 10	-3276.8 to +3276.7 volts
Current	Signed Integer, actual current (in Amps) times 10	-3276.8 to +3276.7 amps
Angle	Signed integer, actual angle (in degrees) times 10	0.0 to ±359.9 degrees
Angular Velocity (Speed)	Signed integer, actual velocity (in RPM)	-32768 to +32767 rpm
Boolean	Unsigned byte, 1 = true/on, 0 = false/off	0 or 1
Frequency	Signed integer, actual frequency (in Hz) times 10	-3276.8 to +3276.7 Hz
Power	Signed integer, actual power (in kW) times 10	-3276.8 to +3276.7 kW
Time	Unsigned long integer or Unsigned integer. These are scaled values in counts that can be calculated by using their respective Scale Factors. For each Scale Factor, see the description column for that parameter.	NA
Flux	Signed integer, actual flux (in Webers) times 1000	-32.768 to 32.767 Webers
Proportional Gain	Unsigned integer, actual gain (unit-less) times 100 OR actual gain (unit-less) times 10000	0– 655.35 OR 0– 6.5535
Integral Gain	Unsigned integer, Actual gain (unit-less) times 10000	0– 6.5535
Derivative Gain	Unsigned integer, actual gain (unit-less) times 100	0– 655.35
Low-pass Filter Gain	Unsigned integer, Actual gain (unit-less) times 10000	0– 6.5535
Per-unit Value	These are scaled values that can be calculated by using their respective Scale Factors. For each Scale Factor, see the description column for that parameter.	NA
ADC Count	The value for ADC counts as read directly by the registers of a microcontroller.	0 - 4095
Pressure	Signed integer, actual pressure (in psi) times 10	-3276.8 to +3276.7 psi

19

CAN Messages: *(Pg. 17 – CAN Protocol, Cascadia Master File)*

Broadcast Messages:

Broadcast messages are sent continuously irrespective of the command mode, VSM or CAN. The default addresses are bases on the default CAN ID Offset of 0x0A0. A parameter 'CAN Active Messages Lo Word' with address 148 defined to enable/disable individual CAN Broadcast messages. The parameter 'CAN Active Messages Hi Word' should be kept as 0xFFFF (or 0xFFFE if implementing the High Speed Message). Each bit in the CAN Active Messages Lo Word represents a CAN Message broadcast status as follows:

a) 0 = CAN Messages broadcast disabled
b) 1 = CAN Message broadcast enabled

Visit (Pg. 17 – CAN Protocol, Cascadia Master File) to view all CAN addresses, frequencies, content and bit designation.

Broadcast Message Definitions:

These can be found from (Pg. 22 – CAN Protocol, Cascadia Master File)

POST Faults:

These can be found on (Pg. 28 – CAN Protocol, Cascadia Master File)

RUN Faults:

These can be found on (Pg. 29 – CAN Protocol, Cascadia Master File)

Command Message:

This is used to transmit data to the controller. It can be sent from a user-supplied external controller. The 8 byte control message (0x0C0) is available on (Pg. 32 – CAN Protocol, Cascadia Master File).

Inverter Enable Lockout:

This protects the invert from accidently being enabled on powerup. Before sending out an Inverter Enable command, the user must send out an Inverter Disable command. After the inverter sees a Disable command, it will begin searching for an Enable command.

Sudden Reversal of the Direction Command:

If the direction command is changed suddenly when the inverter is still enabled, inverter is disabled without triggering any faults. The inverter must be disabled if a direction change is required.

Can Message Sequence Example: *(Pg. 34 – CAN Protocol, Cascadia Master File)*

An example is shown on the following page regarding a 'CAN' mode with run mode set to 'Torque'. The following EEPROM parameters are assumed.

GUI EEPROM Parameter	Default Value	Description
Inv_Cmd_Mode_EEPROM(CAN=0_VSM=1)	0	CAN Mode
Run_Mode_EEPROM(Trq=0_Spd=1)	0	Torque mode
CAN_ID_Offset_EEPROM	0xA0	Default CAN ID Offset
CAN_TimeOut_(/3ms)_EEPROM	333	1 second timeout period

Message Type	CAN ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Description
Rxd	0xAA	4	0	9	0	**0**	0	**128**	0	Torque mode Is active. Lockout is enabled.
Txd	0xC0	0	0	0	0	0	**0**	0	0	Send out inverter disable command to release lockout. **Note** that lockout will not disable if the inverter is faulted. This command should have been set up to be transmitted at a rate sufficient to prevent the CAN Timeout fault. To prevent a fault at startup start sending before the inverter is powered up.
Txd	0xC0	**100**	**0**	x [8]	x	**1**	**1**	0	0	Enable the inverter with a torque command of +10 Nm in forward direction.
Txd	0xC0	**200**	**0**	x	x	1	1	0	0	Set the torque to +20 Nm (motoring) in forward direction.
Txd	0xC0	**156**	**255**	x	x	1	1	0	0	Set the torque to -10 Nm (regenerative) in forward direction.
Txd	0xC0	X	x	x	x	1	**0**	0	0	Disable the inverter before changing the direction. If the direction is changed without disabling the inverter first. The inverter will be automatically disabled as a safety precaution.
Txd	0xC0	**100**	**0**	x	x	**0**	**1**	0	0	Set the command to +10 Nm (motoring) in reverse direction.

'x' indicates a DON'T CARE value, send a zero if unsure.

Sign Convention for Torque and Speed: *(Pg. 36 – CAN Protocol, Cascadia Master File)*

When the controller is in CAN mode, command inputs must be formatted to provide the desired output. Following descriptions provide details on speed command, speed feedback, torque command, torque feedback and direction command and the possible outcome in each scenario.

CAN Speed Command:

Not used for the NU Racing application, documentation available on (Pg. 36 – CAN Protocol, Cascadia Master File)

CAN Torque Command:

For a forward direction command:

- Positive torque command will give a positive torque feedback and is motoring for positive speed.
- Negative torque command will give a negative torque feedback and is regen for positive speed.

Information on negative direction command is omitted as it is out of the rules of the FSAE Competition.

Parameter Messages: *(Pg. 37 – CAN Protocol, Cascadia Master File)*

Parameter messages (0x0C1 and 0x0C2) are used to read and write parameters. Some contain EEPROM data, some are used to change functionality and some are used to monitor parameters not available in the broadcast message.

- To write a parameter; use message 0x0C1 with byte #2 set to 1 (write). The controller will respond with 0x0C2 and upon success, byte #2 will be set to 1.
- To read a parameter; use message 0x0C1 with byte #2 set to 0 (read). The controller will respond with message 0x0C2 containing the requested data.

Both parameter messages contain 4 bytes for the data that is read of written. Some parameters will only occupy a single byte. If the data occupies less than 4 bytes it will be loaded into byte #4 first, followed by #5 and so on. If the parameter address is not recognised, 0x0C2 will contain 0 in both bytes 0 and 1 of the return data.

- **Parameter message format & Address ranges:** *(Pg. 38 – CAN Protocol, Cascadia Master File).*
- **Command parameters:** *(Pg. 39 – CAN Protocol, Cascadia Master File).*
- **Relay command information:** *(Pg. 40 – CAN Protocol, Cascadia Master File).*
- **EEPROM parameters:** *(Pg. 41 – CAN Protocol, Cascadia Master File).*
- **System Configuration:** *(Pg. 42 – CAN Protocol, Cascadia Master File).*
- **CAN Configuration:** *(Pg. 44 – CAN Protocol, Cascadia Master File).*
- **Current:** *(Pg. 47 – CAN Protocol, Cascadia Master File).*
- **Voltage & Flux:** *(Pg. 47 – CAN Protocol, Cascadia Master File).*
- **Temperature:** *(Pg. 48 – CAN Protocol, Cascadia Master File).*
- **Accelerator Pedal:** *(Pg. 49 – CAN Protocol, Cascadia Master File).*
- **Torque:** *(Pg. 50 – CAN Protocol, Cascadia Master File).*
- **Speed:** *(Pg. 52 – CAN Protocol, Cascadia Master File).*
- **Shudder Compensation:** *(Pg. 53 – CAN Protocol, Cascadia Master File).*
- **Brake Pedal:** *(Pg. 54 – CAN Protocol, Cascadia Master File).*

Orion BMS Support: ***(Pg. 57 – CAN Protocol, Cascadia Master File)***

The BMS CAN message can be used to limit the maximum torque commands to a level that approximates the amount of DC (battery) current that will be flowing. The BMS CAN Message must be configured as follows:

- **CAN Message ID:** 0x202 (514 decimal) – *This cannot be changed.*
- **CAN Message Format**:
  Byte 0 and Byte 1 contain the maximum discharge current in Amps.
  Byte 2 and Byte 3 contain the maximum charge current in Amps.

Data format is Little Endian.

Currents can be transmitted as positive or negative as the code will correctly interpret them based on the CAN message format.

Example:

0x02, 0x01, 0x04, 0x02, 0x00, 0x00, 0x00, 0x00 will yield:

- Discharge Limit: (1 x 256) + 2 = 258 A
- Charge Limit: (2 x 256) + 4 = 516 A

The inverter and BMS must run at the same CAN baud rate. To enable the Orion BMS Support, the EEPROM parameter CAN_BMS_Limit_Enable_EEPROM = 1 must be set. The firmware will begin accepting messages from the BMS. If the Inverter is receiving messages from the BMS CAN, BMS_Limit_Msg_Status = 1 will show in the GUI. If the controller stops receiving the BMS messages than the result will return to 0 after one second has elapsed. *There is a flag within the Internal States message indicating when the maximum applied torque level is being limited by the BMS.*

The BMS torque limiting function does not use the inverter measured or estimated DC current. Instead, the following equation is implemented:

- *Maximum torque = DC bus voltage * DC Current Limit / speed*

Where, mechanical speed is in rad/s and torque is in Nm. The equation does not consider motor and inverter efficiency. To prevent the maximum torque from being excessively large the speed used in the above equation has a lower limit *(525rpm for 10pole motor).*

If the inverter is receiving a charge/discharge current limit it will result in a reduced amount of available torque. At lower speeds if the battery current limits are normal, the operator should not notice any difference. If the motor speed is high, a full accelerator application would exceed the battery current limit and the maximum point of the accelerator would be reduced. In CAN mode it will limit the torque command coming from the CAN command message. This also works for 'speed' mode.

Rolling Counter: ***(Pg. 60 – CAN Protocol, Cascadia Master File)***

At this stage this documentation has not been considered, however it should be implemented within the Supervisory system of the CAN network.

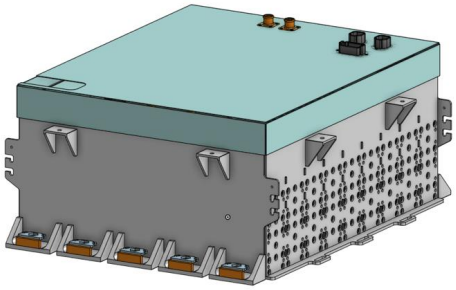
23

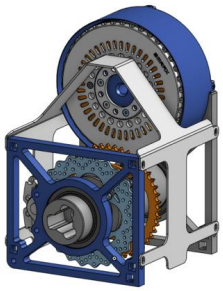
Revision History:

Version	Description of Versions / Changes	Responsible Party	Date
0.1	Initial version	Joshua Hayward	11/03/2024
0.2	• Completed content from Start Guide • Completed content from Hardware Manual • Added Table of Contents • Reformatted Headings and Page borders	Joshua Hayward	13/03/2024
0.3	• Completed content from Software Manual	Joshua Hayward	24/03/2024
0.4	• Completed content from CAN Protocol Manual • Added Revision History and backdated previous versions	Joshua Hayward	26/03/2024

K. EV Tractive System Booklet

EV TRACTIVE SYSTEM





2024


DESIGN OVERVIEW
RACING
DESIGN GOALS
5 YEAR PLAN
2023
• Top 3 in auto-x
• First through tech
• 400 kms testing
• 260 kg car
• Competitive in all static events
2024
• Finish top 5 in endurance
• First traction limited drivetrain output
• Reduction in car weight by 10%
2025
• Design cycle shift earlier
• 80 kW capable accumulator
• Limited slip rear axle
2026
• Monocoque half chassis
• Adjustable anti-roll
• < 215 kg car
2027
• Full monocoque chassis
• < 200 kg
HOW WE WIN
1. Align bottlenecks
2. Prefer simple self-documenting systems
3. Reduce part count
4. Prioritise long-term team performance
ROBOTIC SYSTEMS
JANUS ELECTRIC
AMPCONTROL
SAFEgroup Automation
DSI UNDERGROUND
ResTech
THALES
Building a future we can all trust
Ausgrid
BANLAW
BME TECHNOLOGY
VARLEY
THE UNIVERSITY OF NEWCASTLE AUSTRALIA
NU TEAMS

DESIGN OVERVIEW

POINTS GOAL

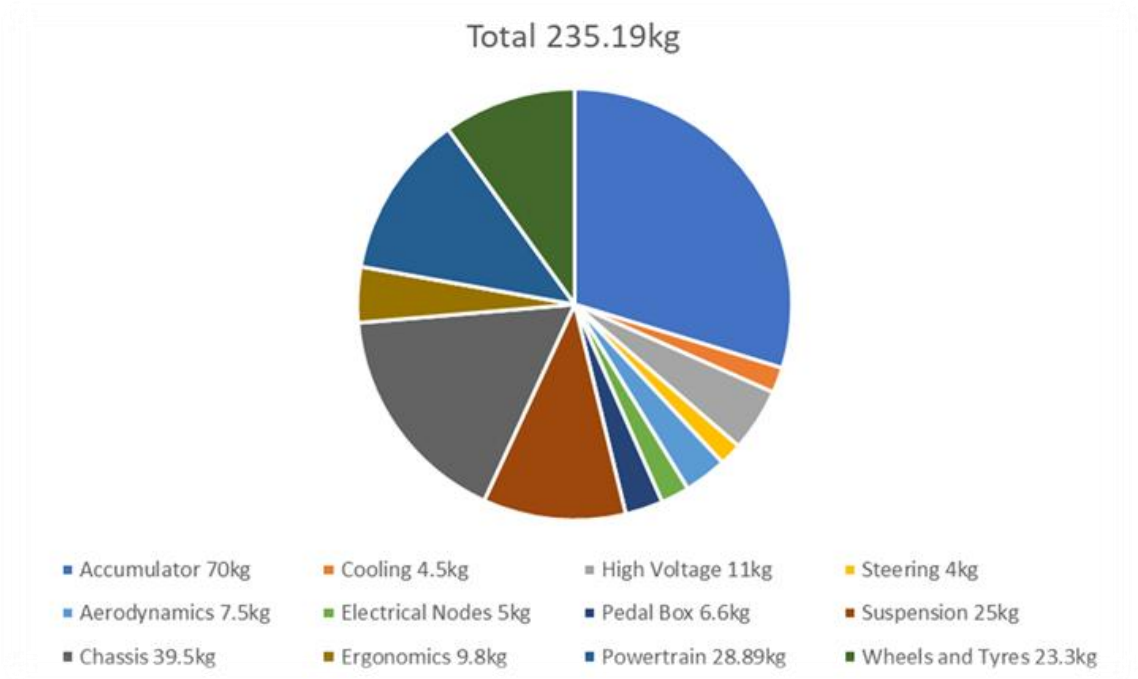
Event	2023 (7th)	2024 Goal	Delta	Out of
Presentation	72.55	65	-7.55	75
Cost	51.06	70	+18.94	100
Design	130.33	135	+4.67	150
Skid pad	65.77	66	+0.23	75
Acceleration	50.92	65	+14.08	100
Autocross	120.99	121	+0.01	125
Endurance	9*	260	+251	275
Efficiency	52.52	65	+12.48	100
Total	**553.14**	**847**	**+293.86**	**1000**

The total points goal of 847 points would result in first place (by 1.16 points) based on the 2023 Results.

* Projected Points from Endurance 2023 based on the average lap time would be approximately 255.

DESIGN OVERVIEW

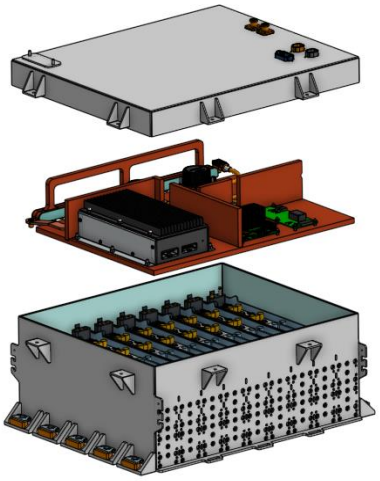
MASS BREAKDOWN

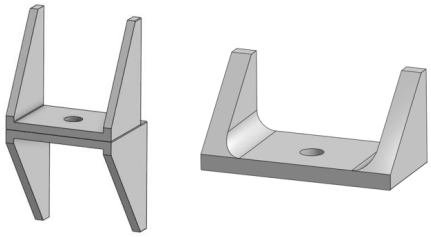


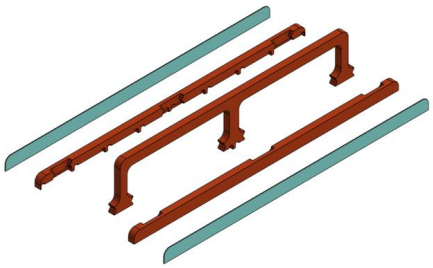
EV TRACTIVE SYSTEM

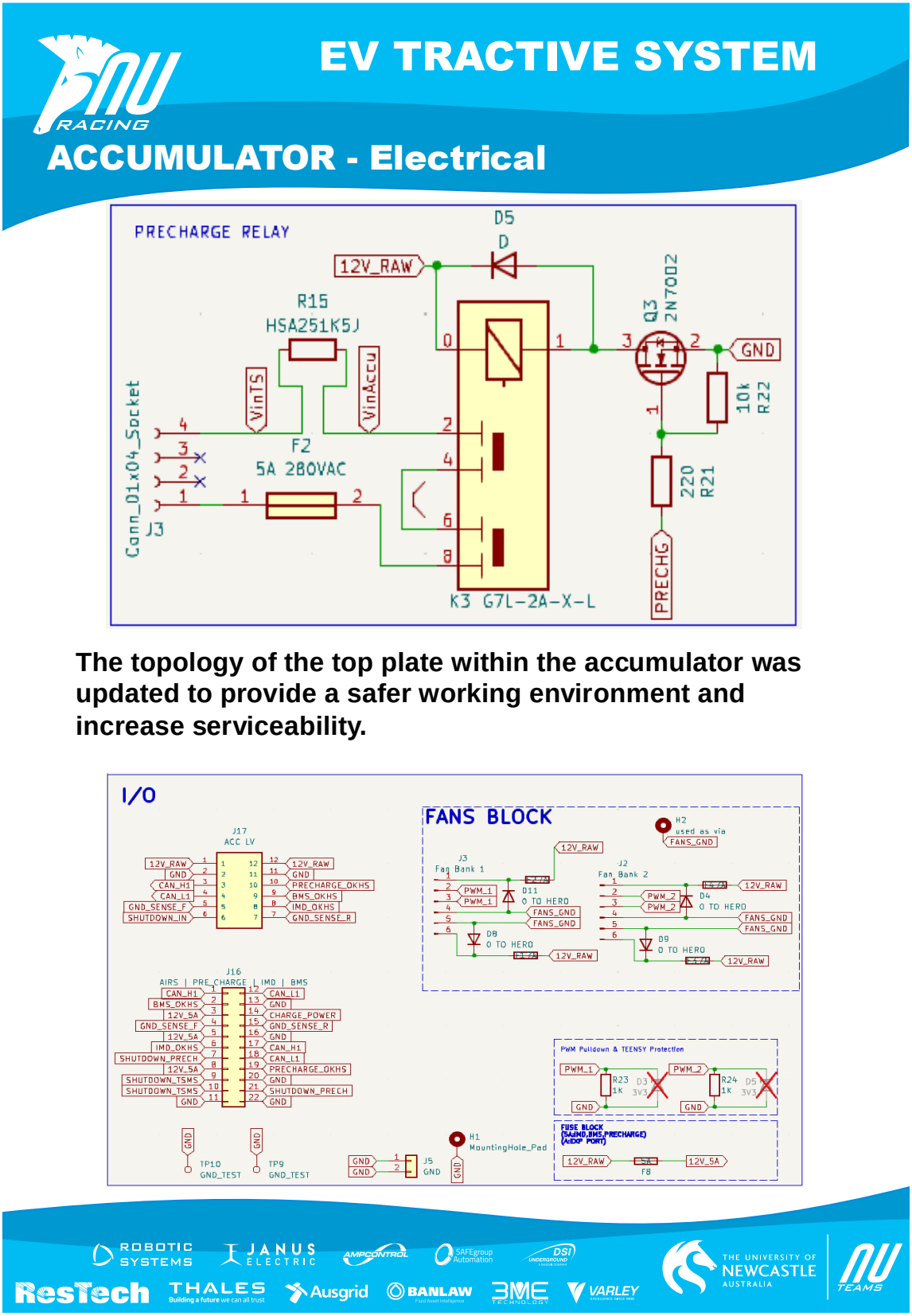
ACCUMULATOR - Structural

- Made from aluminium
- 5kg weight reduction
- Mounts and tabs made from existing extrusion profile
- Use thinner sheets of FR4 to line the container
- Lighter maintenance plug









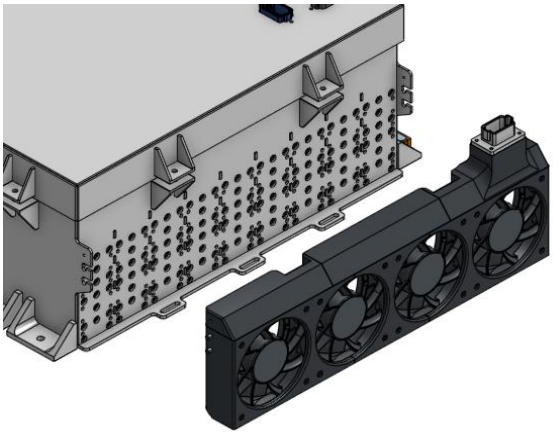
The topology of the top plate within the accumulator was updated to provide a safer working environment and increase serviceability.

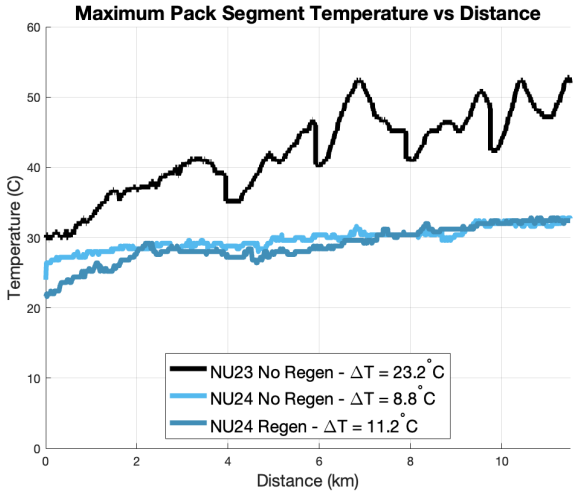
EV TRACTIVE SYSTEM

ACCUMULATOR - COOLING

Accumulator cooling design

- Retains previous reliable fan configuration
- New aluminium construction
- Better heat transfer





Accumulator thermal performance increase

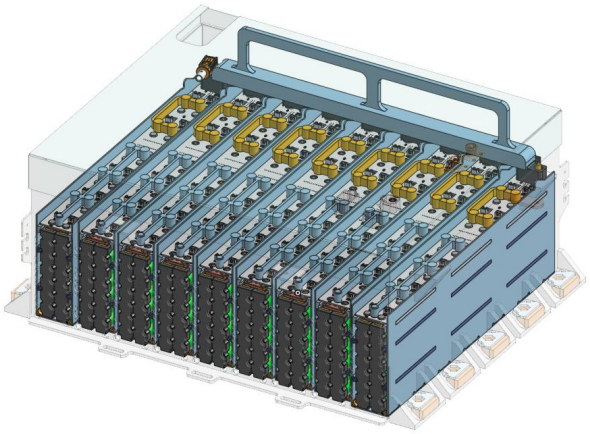
- Maximum cell-thermistor value shown
- Temperature above ambient decreased by over 50%
- Much more stable temperatures improve reliability
- Reduces driver involvement in managing cell temperatures

EV TRACTIVE SYSTEM

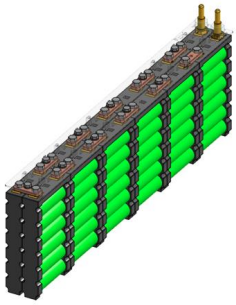
ACCUMULATOR – Cell arrangement

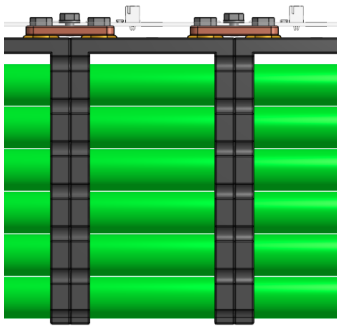
ENEPAQ VTC6 Cell module

- 3.6V Nominal
- NU24 uses these modules in a 108S-6P configuration
- 388.8V Nominal pack voltage
- 18Ah rating results in total capacity of 7kWh
- Maximum TS voltage of 453.6V



Service Handle and cell arrangement





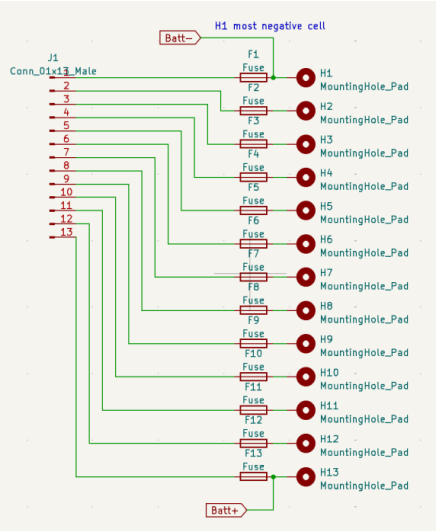
Cell busbars

EV TRACTIVE SYSTEM

ACCUMULATOR – BMS

Orion BMS2

- 108 Cell monitoring
- Reads all cell-voltages and handles cell-balancing
- Causes hard faults when cells are performing weak to avoid risk of danger
- Integration with Cascadia CM200DX inverter

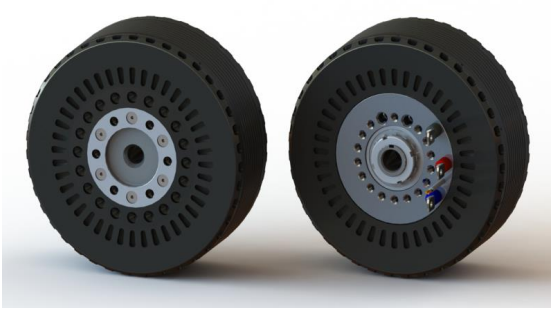


CANaMons Cell Monitors

- Measures and reports all cell voltages and temperatures to the BMS
- Secondary CAN network for BMS communications
- PCB Design reduces wiring harness complexity

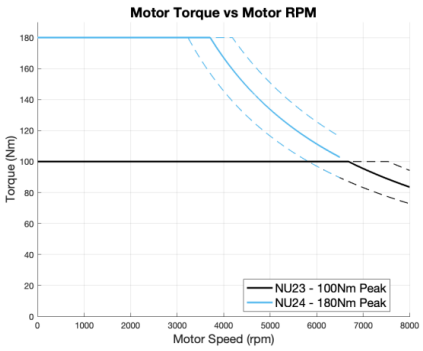
EV TRACTIVE SYSTEM

MOTOR & INVERTER



EMRAX 228 MV
- Maximum 630V
- 230Nm Peak Torque
- 75kW Continuous Power
- Up to 98% Efficiency

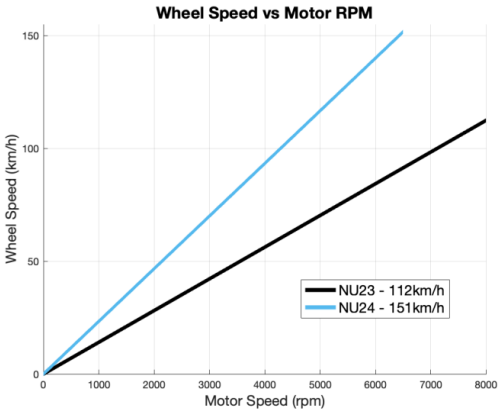
CASCADIA MOTION CM200DX
- 480V peak
- 300A Continuous Current
- Direct support with EMRAX motors

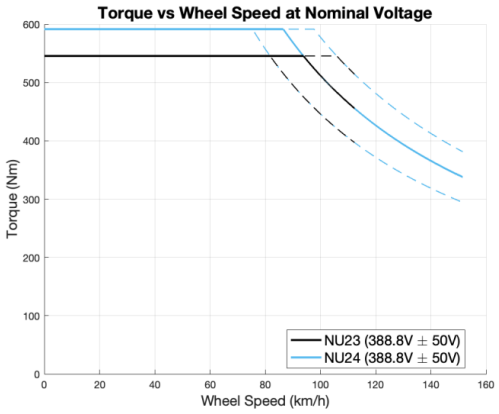


Increased Motor Torque
- 180Nm imposed limit
- 180A imposed limit
- Decreased RPM limit

EV TRACTIVE SYSTEM

MOTOR & INVERTER



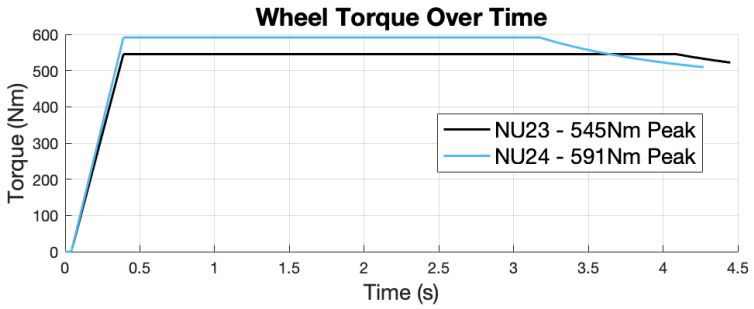


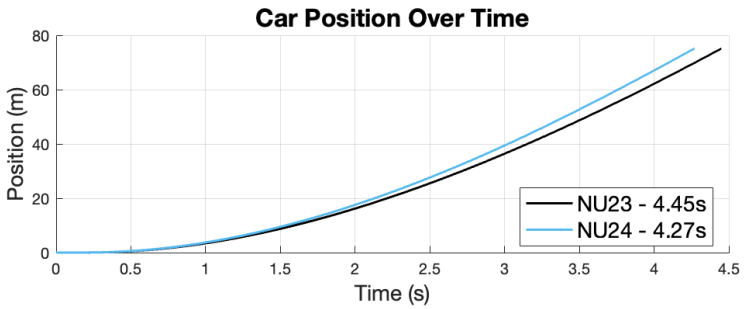
Decreased gear reduction

- Reduced from 60:11 to 46:14
- Increased top speed by 39km/h
- No decrease of wheel torque given lower ratio
- 8.44% increase in peak wheel torque

EV TRACTIVE SYSTEM

MOTOR & INVERTER



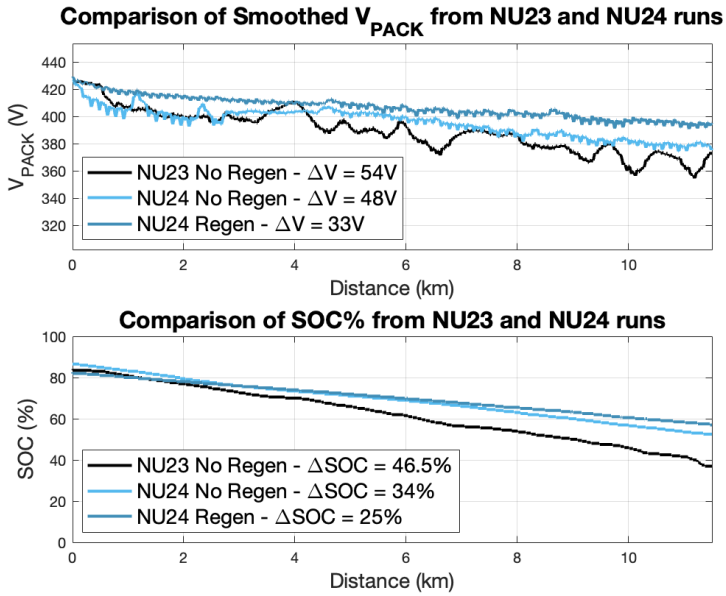


Acceleration simulation

- Modelled based on the performance of NU23
- Assumed no traction loss as validated by testing
- 5km/h increased trap speed
- Greater torque available to driver during acceleration
- 4.1% decrease in acceleration time

EV TRACTIVE SYSTEM

MOTOR & INVERTER

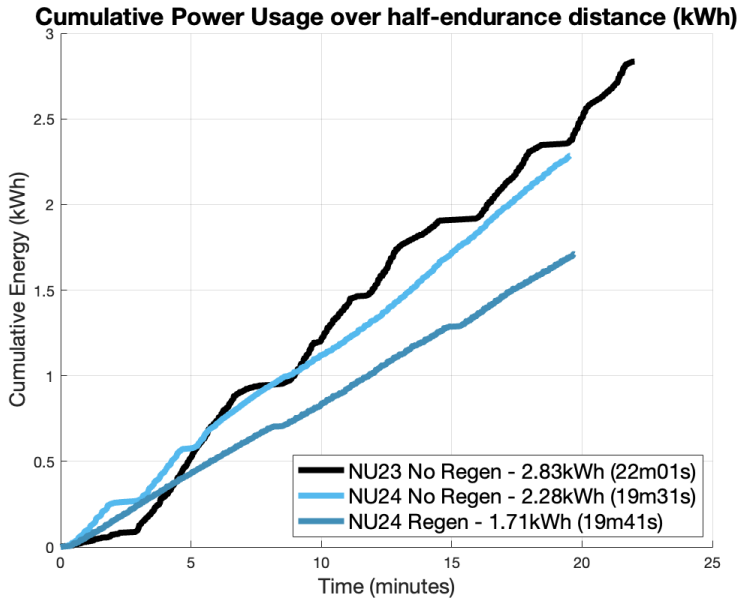


Regenerative Braking

- Implemented to recover kinetic energy
- Improved driving efficiency during endurance
- Allows the drivers to drive harder for longer
- Reduced battery usage by 40% with more powerful motor when compared to NU23
- Single-pedal driving allows fewer braking applications
- Multivariate adjustable torque response including; Maximum commanded regen torque and APPS percentage regen crossover threshold

EV TRACTIVE SYSTEM

MOTOR & INVERTER

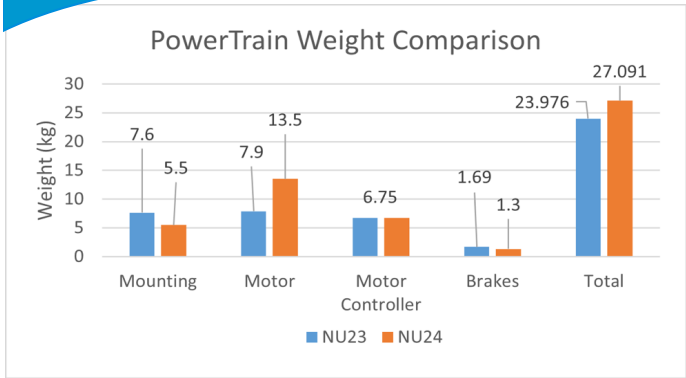


Tractive system efficiency

- New EV Tractive system design allows for less energy usage when compared to NU23 (20% reduction for non-regen based runs)
- Implementation of regenerative braking allows for a further reduction in energy usage (additional 20% reduction when enabling regen)
- Despite 40% total energy reduction, time was improved by 11%

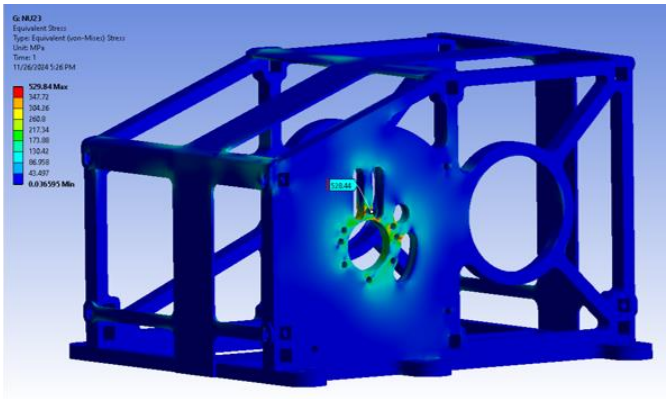
EV TRACTIVE SYSTEM

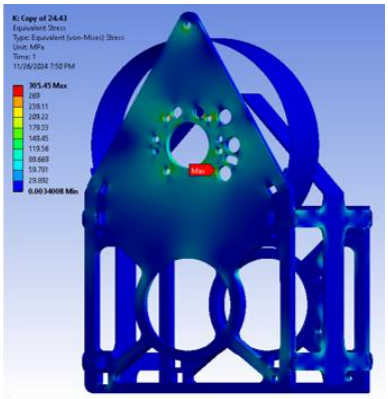
POWERTRAIN – "POWERBOX"



Ways in which we worked towards our goals:

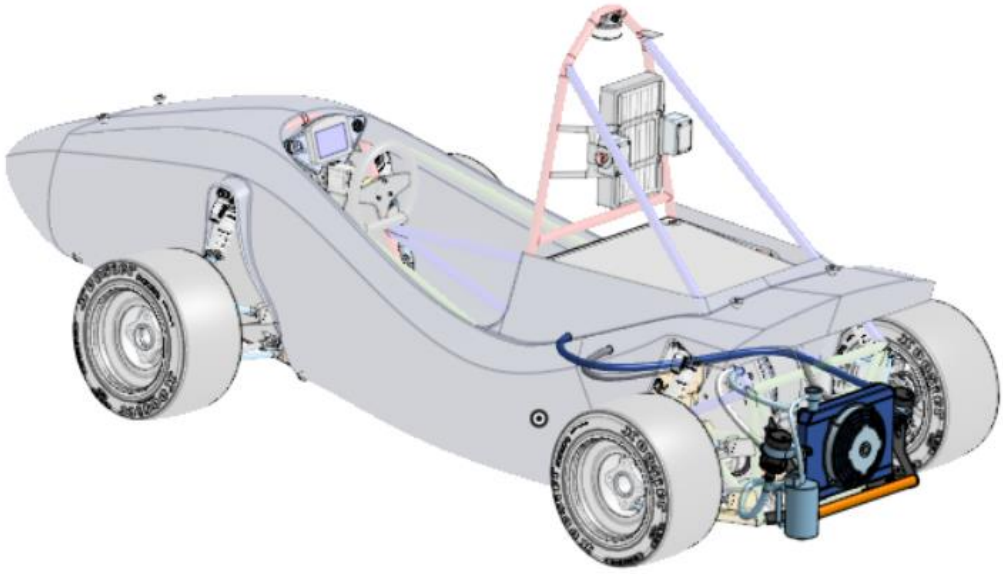
- Modular design
- Reduce part count
- Increase in torque with an acceptable increase in weight
- Reduced footprint to fit in new chassis





EV TRACTIVE SYSTEM

COOLING



Simple and Reliable

- Dual Cooling Loops using a single radiator
- Space Efficient
- Great serviceability
- Lightweight

EV TRACTIVE SYSTEM

COOLING

PWR Radiator
- 328x175mm
- 16FPI
- Dual Pass converted to separate loops

EBP40 Water Pumps
- Max 35LPM Flow
- Max 0.85 Bar Head Pressure

Spal Puller Radiator Fan
- 190.5mm (7.5inch) diameter
- Max 730 m^3/h Flow

References

[1] A. Brewer, "Flexcan_t4." GitHub repository, 2019. Accessed: 2025-01-15.

[2] M. Whitworth, "Nucan." Github Repository, 2023. Accessed: 2025-01-15.

[3] J. Bush, "Lead mechatronics engineer." Technical Report, 2024. Accessed: 2025-01-15.

[4] EMRAX, "Emrax 228 datasheet v1.5." PDF document, 2025. Accessed: 2025-01-15.

[5] J. Hollier, "2023 chief engineer and mechanical lead." Technical Report, 2024. Accessed: 2025-01-15.

[6] L. Fisher, "2024 chief mechanical engineer, vehicle dynamics, cost event and technical inspection." Technical Report, 2025. Accessed: 2025-01-18.

[7] R. Mathuria, "2024 powertrain and business presentation." Technical Report, 2025. Accessed: 2025-01-18.

[8] A. Chapman, "2024 chief mechatronics engineer." Technical Report, 2024. Accessed: 2025-01-18.

[9] Cascadia-Motion, "0a-0151-01 - manual - resolver calibration process." PDF document, 2011. Accessed: 2025-01-29.

[10] Cascadia-Motion, "0a-0163-01 - sw user manual." PDF document, 2011. Accessed: 2025-02-05.

[11] Kvaser-User:Dynamic, "Why do i get error frames?," 2017. Accessed: 2025-02-05.

[12] Ewert-Energy-Systems, "Dtc p0a80 - weak cell fault," 2020. Accessed: 2025-02-14.