Winter 11-2016

# FLEX-RO: DESIGN, IMPLEMENTATION, AND CONTROL OF SUBASSEMBLIES FOR AN AGRICULTURAL ROBOTIC PLATFORM

Jared Patrick Werner
*University of Nebraska - Lincoln*, wernerj09@gmail.com

# FLEX-RO: DESIGN, IMPLEMENTATION, AND CONTROL OF SUBASSEMBLIES FOR AN AGRICULTURAL ROBOTIC PLATFORM

by

Jared Patrick Werner

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master of Science

Major: Agricultural and Biological Systems Engineering

Under the Supervision of Professor Santosh Pitla

Lincoln, Nebraska

November, 2016

# FLEX-RO: DESIGN, IMPLEMENTATION, AND CONTROL OF SUBASSEMBLIES FOR AN AGRICULTURAL ROBOTIC PLATFORM

Jared Patrick Werner, M.S.

University of Nebraska, 2016

Advisor: Santosh Pitla

Automation technology in agriculture is growing, making agricultural robotics viable. Innovative field usable multi-purpose robotic platforms are needed for the successful progression of agricultural robotics. Furthermore, the field of agricultural robotics would benefit from a robotic platform design allowing for variable height, thus accommodating navigation throughout various crop growth stages. A variable height machine, Flex-Ro was developed to accommodate this feature. Multiple sub-assemblies were designed and implemented for Flex-Ro. An electronic control unit (ECU) enabled engine was used to power Flex-Ro. An embedded application program was developed to control engine speed using proprietary Controller Area Network (CAN) messages in conjunction with J1939 standard messages. It was observed that the maximum standard deviation from the requested set speed was 8.423 rpm. A hydrostatic transmission system was designed and implemented on Flex-Ro. An embedded application program was developed for controlling the pumps and motors of the hydrostatic transmission utilizing proprietary CAN messages. Additionally, the embedded controller operated a Control Cut Off (CCO) that regulated flow to the spring applied motor brakes and the Electronic Displacement Control (EDC) which controls pump flow. Based on the motor speed data collected, it was observed that an increase in deviation occurred as the operational speed increased with a minimum standard deviation of 6.98 rpm at 50 RPM and a maximum standard

deviation of 36.13 rpm at 156 RPM. The measured data should be used in developing further higher level control programs. A vertically adjustable frame was developed to allow Flex-Ro to enter crops at various growth stages. A steering system was developed and programmed to allow for steering control through the CAN bus again using proprietary CAN messages. Lastly, a remote control program was developed to allow messages to be created and wirelessly transmitted to Flex-Ro's CAN bus for teleoperation. Sub-assemblies of Flex-Ro will be further developed for fully autonomous navigation, and performing various field operations.

**Acknowledgements**

I would like to thank my advisor Dr. Santosh Pitla for offering me the privilege of working on this project. Thanks for providing me guidance, knowledge and direction while also allowing me freedom to manage the project how I saw fit.

Thank you to Danfoss for donating the hydraulic pump, and drive motors implemented on the robotic platform. Thank you to Anderson Industrial Engines CO. and Kubota Engine America Corporation for donating the engine, various engine components, and technical documentation used on this robotic platform. Thank you additionally to Magnetek for donating the Wireless CAN Modules, Goodyear for donating tires, and Surplus Center for donating various hydraulic fittings.

Furthermore thank you to undergraduate workers Nick Jarecki and Jeremy Blackford who helped assemble the engine and frame components, graduate student John Evans who assisted in hydraulic data collection, and all other UNL students and faculty who have given support on this project.

Thanks also go to my committee members, Dr. Joe D. Luck, Dr. Michael F. Kocher, and Dr. Roger Hoy for their support.

Above all, thank you my family to whom I dedicate this thesis. To my parents who never got too angry when I did not put the things I disassembled back together, and have always supported the decisions I have made in my pursuit of knowledge and education. My wife who has been there supporting me every step of this master's journey; providing guidance, love, and encouragement.

# Table of Contents

**List of Figures**

**List of Tables**

## Chapter 1 Introduction

The field of agricultural robotics is growing to automate agricultural processes essential for increased production to support the needs of a growing population. One of the next steps in increasing production, while reducing production costs, is to develop machinery that can operate extensive hours without the constant physical control of an operator. Many large scale machinery manufacturers have semi-automated navigation, with the exception of headland turning. It is logical that fully autonomous field machinery will be the next advancement in agriculture.

Many autonomous ground vehicles have been developed for various applications. These vehicles can provide safety from harsh environments, allowing sensing and research to be performed in areas previously too dangerous for researchers (G. Muscato et al., 2003). They can also be used as a luxury item, as seen with self-driving cars, to allow unfit drivers to travel safely, or just alleviating the need to attentively operate a vehicle (Urmson et al., 2008). However, most of the current agricultural robots have some disadvantages that are limiting field deploy-ability. In many instances of agricultural production, one of the best features an automated machine can possess is extended operating time. However, many of the agricultural robots that have been developed rely on battery power sources which limits the field operating time (Jinlin and Liming, 2010; Slaughter et al., 2008). Batteries are a finite energy source that when used require recharging off site, as many rural operations do not have direct access to power sources. While some machines rely on solar panels to supplement battery usage, this is an unreliable method, considering sunlight limiting cloud coverage. Furthermore, the best

time to charge these batteries is during non-operating hours, which is usually when there is no daylight. Using batteries as a stand-alone power source limits the number of electronics you can have on a machine. Current production machinery uses a vast amount of technology. Typical sensors calculate spatial location, yield during harvest, and monitor most important operations that occur within the machine. With an agricultural robot all these would be required, with the exception of operator display and comfort. Moreover, the sensors required for maintaining safe operating conditions for autonomous machinery, computing requirements for intelligent operation, and end user communication are important and all require extensive electrical power.

Much of the focus in agricultural robotics is in computer vision for autonomous navigation (Bulanon et al., 2004; Chen et al., 2003; Slaughter et al., 2008; Tanigaki et al., 2008), and not much information is available on drivetrain selection for field robotic machinery. Hybrid vehicles, while prevalent in automobiles, have not been extensively tested in an agricultural setting. The hybrid system would use a combustion engine to convert mechanical power to electrical, such that it can be stored in a battery for continued use.

Traditional agricultural machinery is constrained by its ability to navigate through crops at various growth stages. Field robotic platforms for Midwestern row crop operations would benefit from variable height and width designs for working in the crop at different growth stages. A system of this nature needs to be developed as a full machine that can be implemented into agricultural fields with varying terrain. This type of system would allow for navigation to various portions of the field to perform controlled weeding. It

would also allow for automation in field sampling and scouting that could improve crop management practices. Independently steered and driven wheels need to be implemented on this type of a robotic platform. This is due to the limitations of mechanically linked powertrain systems.

This thesis covers the design, implementation, and control of various subassemblies implemented on an agricultural robotic platform referred to as flexible structure robotic vehicle (Flex-Ro). Chapter 2 consists of engine selection, implementation, and control for use on Flex-Ro. Chapter 3 examines the hydrostatic transmission configuration, implementation, and control of the wheel speeds for the Flex-Ro. Chapter 4 highlights the design of Flex-Ro's frame assembly. Chapter 5 discusses the steering system component selection and design, and the embedded application developed. Chapter 6 highlights the programming and communication for a remote control that allows for teleoperation of Flex-Ro. Chapter 7 brings the work of this thesis to a conclusion and details future work to be implemented on Flex-Ro. The appendices of this thesis contain important component drawings, programming, and manual documentation used when developing Flex-Ro, and hence forms significant section of this thesis.

# Chapter 2 Embedded Engine Control Application Program Development and Implementation on a Flex-Ro via CAN Bus and SAE J1939 Messages

## 2.1 Engine Introduction

With the field of robotics expanding, it is imperative for agriculture to embrace highly automated systems for efficient crop production. Some mobile agricultural robotic machines currently being developed use batteries as their primary power source (Tabile et al., 2011). With many actual field operations requiring long hours to complete and substantial power requirements, this power source is unsustainable for continuous field work. A solution to address this problem is the use of an internal combustion (IC) engine as the primary power source. Some researchers have used hydrostatic transmissions (Godoy et al., 2012; Oksanen, 2015) or have transformed existing conventional machines (Bergerman et al., 2012) into autonomous machines.

For agricultural robots where engines are used, engine speed control is an important aspect for automation. Small engines allow for electric starters, but mostly have mechanical throttle controls. These mechanical controls can be automated through the use of various actuators; however, they are less accurate than other methods, and have higher cost and other fabrication needs. Additionally, these engines must be implemented with sensors to collect important performance data. Manual engine control methods are prone to improper operation leading to inefficiencies and safety concerns. Current technologies implemented on electronically fuel injected gasoline and diesel engines allow for extensive data collection, control, and safety measures. An Electronic Control Unit (ECU) onboard allows for diagnostic procedures, as well as providing embedded safety

measures that will de-rate the engine, or shut the engine down if operating conditions become unsafe. Further, engine speed can also be controlled via CAN (Controller Area Network) messages sent to the engine ECU.

CAN systems are used extensively on modern equipment. While instrumenting electronics on agricultural field machinery occurred in the 60s (Stone et al., 2008), the CAN bus protocol was first implemented by Robert Bosch GmbH in 1986 through cooperation of automobile manufacturer Mercedes Benz (Voss, 2005). The CAN protocol standard, SAE J1939, was proposed for use in agriculture in the early 1990s. The SAE J1939 defines parameters in a public database for manufacturers to use as a guide while programing ECUs (Voss, 2008). CAN bus is currently implemented on almost all agricultural machinery, and often allows communication among multiple ECUs in a distributed control architecture protocol.

The SAE J1939 standard, as well as others publicly available standards in place, allow for machinery performance data monitoring. Research has centered on collecting data through the use of these standardized messages (Marx et al., 2015; Pitla et al., 2016; Pitla et al., 2014). Given the rate of technological advancements, it is challenging to address the need of these new advancements with current standardized messages. To use these technologies and control methods, manufacturers create messages that do not conform to existing standards, and are considered proprietary. Companies often do not release information on these messages for public use. CAN bus based engine control, through both public and proprietary messages, is important for both autonomous and manned machinery. CAN communications occur over a two wire bus reducing the amount of wiring required to operate the engine from the operator station of modern agricultural

equipment. Additionally, it removes the use of mechanical linkages to the engine, saving space for other components, as well as allowing conditional safety features to be programmed into the machine. The overall goal of this research work was to develop a CAN based embedded control application for engine control to be implemented onto the agricultural robotic vehicle, Flex-Ro.

## 2.2 Engine Objectives

The specific objectives of this project were to:

(1) Develop a CAN based embedded application program for controlling the engine speed using a proprietary CAN message.

(2) Validate the application program's control of engine speed using the CAN equipped engine installed onto the agricultural robotic vehicle, Flex-Ro.

## 2.3 Engine Methods and Materials

### *2.3.1 ECU equipped Gasoline Engine*

A medium sized gasoline engine (approx. 40 HP) with liquid cooling and an onboard ECU was used as a power source for the robotic vehicle under development. The drivetrain of Flex-Ro consisted of a hydrostatic transmission with a pump that drove four hydraulic motors. Ease of engine data collection and the ability to control the engine through the use of an onboard ECU without the need for additional actuators reduced the complexity of the overall system. A gasoline engine was preferred to a diesel engine because of the high capital costs and the additional hardware requirements to address the Tier 4 final emission requirements of the diesel engine. A 57 HP Kubota Industrial

Engine (WG1605-GL-E3-KEA-1, Kubota Engine America Corporation, Lincolnshire, Illinois) (Figure 2.1) which can be operated using either gasoline or natural gas was selected as the power source for the robotic platform.

**Figure 2.1: Kubota WG1605-GL-E3-KEA-1 bare engine used as the power source for the Flex-Ro robotic platform.**

The engine was equipped with a manufacturer provided pre-programmed ECU (Figure 2.2). Specifically, this ECU allowed for the safety requirements, engine speed controllability, and the setting of the governor speed. The governor speed setting was used to restrict the maximum engine speed to 3450 rpm. This engine speed was chosen not to exceed the maximum allowable pump shaft speed of 3500 rpm (Danfoss, 2015). This setting ensured that the engine speed was always within safe operating conditions of the pump that was coupled to the engine output shaft.


**Figure 2.2: Kubota 4G custom programmed engine ECU for the Flex-Ro robotic platform.**

The engine ECU accepted a J1939 standard CAN message that consisted of the desired engine speed or torque. This message named TSC1 (Torque/Speed Control 1), defined in the CAN standard, (SAE, 1994) is an 8 byte message transmitted at 100 Hz. The engine speed is encoded in two data bytes, D1 and D2 within the message. This message has a resolution of .125 rpm/bit and allowed for inputs of 0 to 8031.875 rpm (SAE, 1994). The communication among the system components is depicted in Figure 2.3. Note that all CAN messages published to the bus may be accessed by any node of the bus. However, the embedded controller (Plus+1 Controller, Danfoss) was programmed to receive the computer generated proprietary message and send the TSC1 control message to the engine ECU. A control box consisted of relays and connections communicated via a hardwire connection with the Plus +1 controller, through digital inputs and outputs, and sent power to the engine's starter and run inputs.

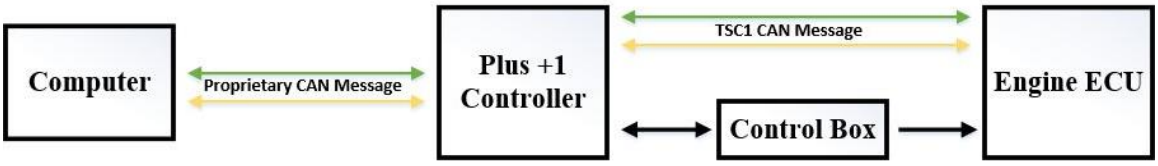



**Figure 2.3: Block diagram of the system and the control message flow between the user computer, embedded controller (Plus+1) and the engine ECU.**

### *2.3.2 Plus+1 Controller and Instrumentation*

A Plus +1 controller module (Model: MC024-010, Danfoss, Ames, Iowa) shown in Figure 2.4 was programmed for engine speed control. This module was selected based on the number of analog, digital I/O channels, CAN capabilities, and the weatherproof packaging (Danfoss, 2008).


**Figure 2.4: Danfoss MC024-010 programmable Plus +1 controller for engine control.**

For manual engine startup, the controller read two inputs, an ignition run command, and a

start command, both of which were physical inputs on the machine. The start command was not executable unless the ignition run command was positive, which forced the machine to be in run mode prior to starting. The digital inputs on connector 1, pin 7 (C1p07) and pin 10 (C1p10) receive signals from the physical run and start switches, respectively (Figure 2.5). After passing through the logic these commands are sent out of the controller through multifunction digital output pins 9 (C2p09) and 10 (C2p10) on connector 2 to relays, which draw electricity directly from the battery. A wiring diagram of the physical inputs as well as the relays is illustrated in Figure 2.6.

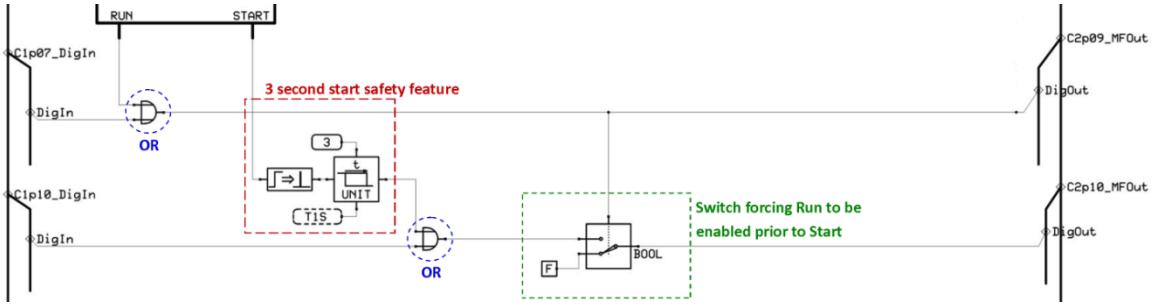



**Figure 2.5: Plus +1 engine program allowing for physical button or CAN message controlled inputs for the start and run sequence on the engine.**

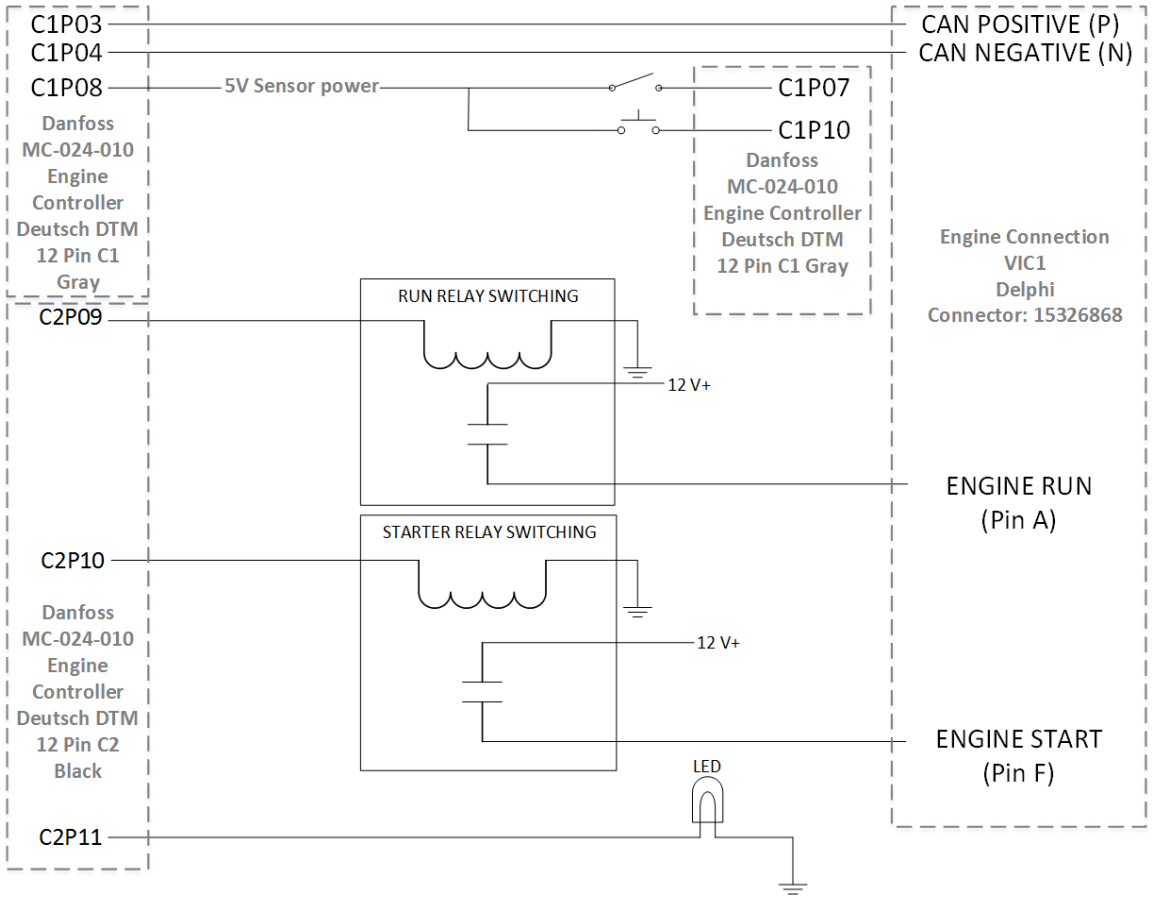



**Figure 2.6: Wiring diagram for control box that bridges the Danfoss Plus +1 controller and the Engine ECU.**

### *2.3.3 Embedded CAN Application Program*

A set of J1939 compatible function blocks were downloaded for use within Plus +1 Guide application program (Danfoss, 2012). Included in the library was a TSC1_Tx function block, which compiles and broadcasts the TSC1 message based on user set variables (see Figure 2.7).

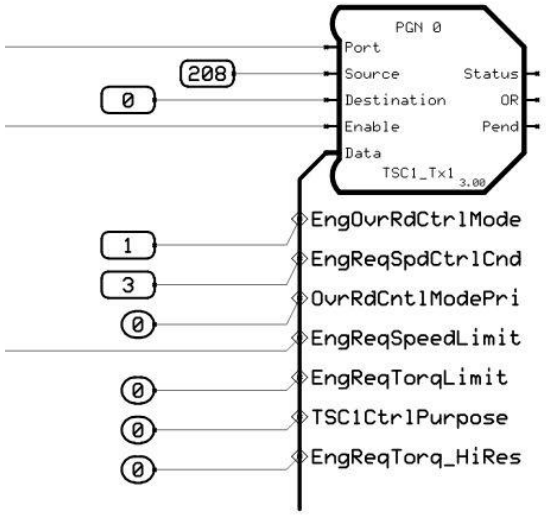



**Figure 2.7: Plus +1 J1939 TSC1 function block.**

Using a combination of the function library literature, as well as the manufacturer CAN data, the user set inputs were configured (Danfoss, 2012; Kubota, 2012). The input port was connected to the CAN_0 port to allow for the message to be sent across the CAN bus. The source and destination, respectively 208 and 0, can be found in the manufacturers data, and the "Enable" input was set to true to broadcast the message. Setting EngOvrRdCtrlMode to 1 allowed the message to act as the speed control type for desired engine speed. The EngReqSpdCtrlCnd was set to 3, such that the ECU controlled the engine speed for stability optimized for driveline engaged condition (being a PTO driveline). EngReqSpeedLimit is adjusted to the desired engine speed, and all other inputs are 0 for default operation.

As previously stated, the run, start, and engine speed can all be controlled via a specifically formatted CAN message. This message is an 8 byte proprietary message (Figure 2.8); therefore it does not fall under any standardized message sequence. This

message was broadcast on the CAN Bus with both the Mask and ID set to 104.

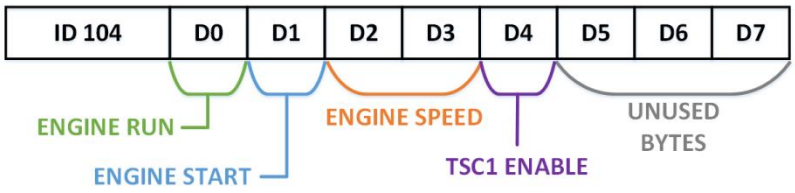



**Figure 2.8: Proprietary CAN message arrangement for engine control.**

Additionally, only five of the eight available bytes within the message carry pertinent information. The byte in position D0 is for control of the run command. This byte is filled with a 0 if the command is to be off or a 1 to activate the command. The Start command follows the same nomenclature as the Run; however, the byte used for the Start command is in the D1 position. Additionally, the D1 start command byte will allow power to flow to the starter relay for three seconds allowing the byte to be sent a single time to enable the starter. If the start command needs to be re-sent, the byte must first be sent as zero, and then be re-sent as 1. Enabling the TSC1 J1939 command is also performed in the same way but using the D4 byte position. Last, the desired engine speed is also set in this message. The size of the engine speed requires the use of 2 bytes within the message to adequately control the speed. The quotient of the desired speed was taken using 256 as the denominator, as 255 is the maximum number allotted in an 8 byte message, this value was then entered into the D3 byte position. The remainder of the division was stored in the D2 byte position. The use of this method to send the desired engine speed to the controller results in a resolution of 1 rpm per bit in the proprietary message, forcing desired engine speed to be sent as a whole number. This was a reasonable resolution for control of the engine, as a desired engine speed increment of engine speed below 10 rpm was not needed. The engine speed was adjusted in increments of 50 or 100 rpm for this

application.

In order to implement this proprietary message a functional page called Engine CAN Decoder was created within the Plus +1 Guide program. This page connected the CAN_0 bus as an input and sent control output Boolean values of 0 or 1 for the Run, Start, and enabling of the TSC1 command as shown in Figure 2.9. Additionally, the desired engine speed was output as an S16 number (signed 16-bit integer).

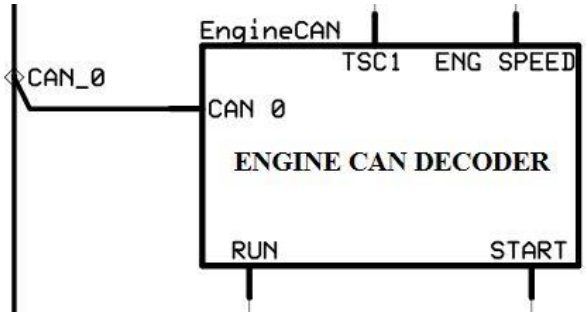



**Figure 2.9: Shell containing the decoding program for the Engine Control Message.**

Within the page, the CAN message with the mask and ID as both 104 are taken from the bus using the CAN_Rx1 function block found in the Danfoss Basic Function Library (Danfoss, 2005). Following the nomenclature for the engine control proprietary message the Can_Data_0 was taken to switch the Run command Boolean control. This same method was used to allow the CAN_Data_1 to control the Start command, as well as the CAN_Data_4 to enable the TSC1 command. When the start command activated via the CAN message, the Plus +1 controller sent power to the starter relay located in the control box for three seconds, and then turned off. The programming of the three second delay feature is shown in Figure 2.10.

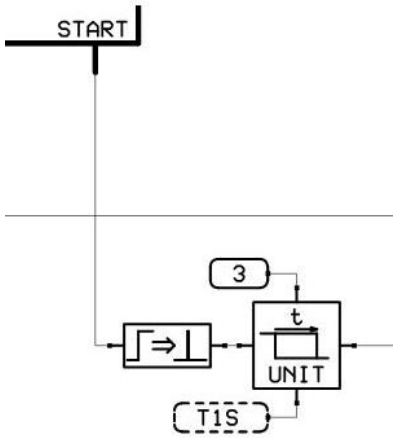



**Figure 2.10: Three second start signal safety feature.**

Additionally, the CAN_Data_3 was multiplied by 256 then added to CAN_Data_2 to output the desired engine speed control signal (EngReqSpeedLimit) for the TSC1 function block. These programmed processes are all illustrated in Figure 2.11.

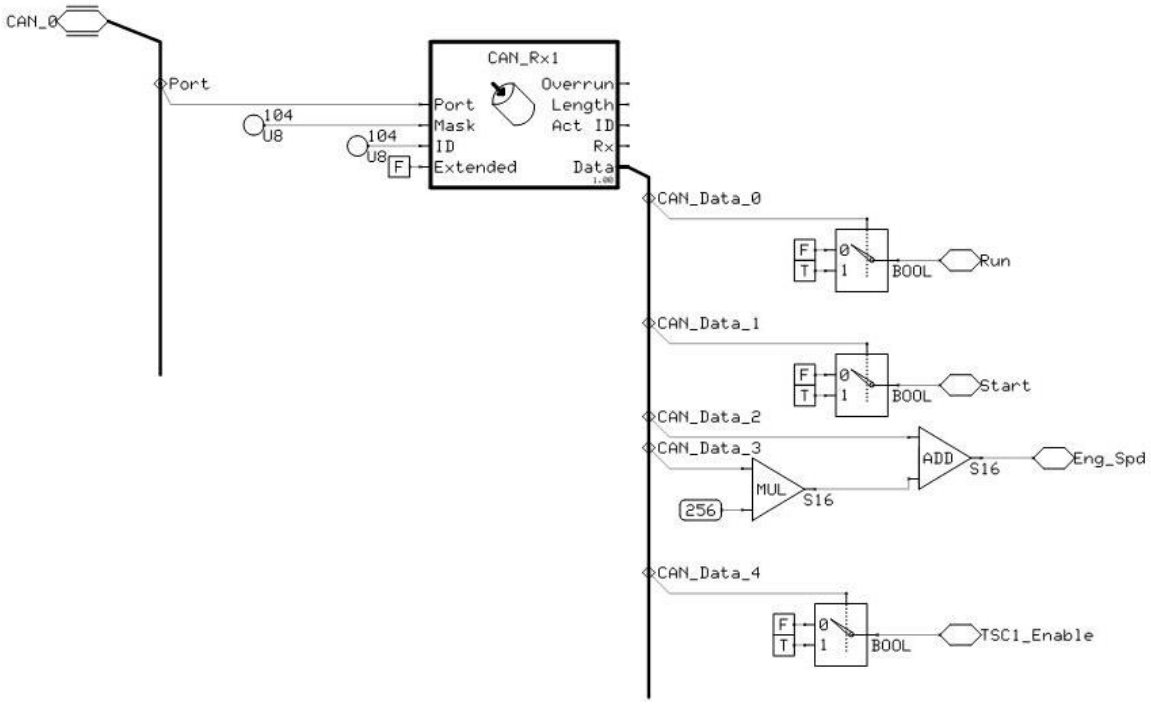



**Figure 2.11: Program to receive and decode CAN Engine Control Message.**

An additional failsafe was programmed for the desired engine speed to protect the equipment from operator error. As the two byte combination for desired engine speed allowed the user to submit extremely large values, a check point only allowed values above 850 and below 3401 rpm to be submitted. If values outside of this range were submitted to the TSC1 command it reverted back to the 850 rpm idle state. The programming for this failsafe is shown in Figure 2.12.

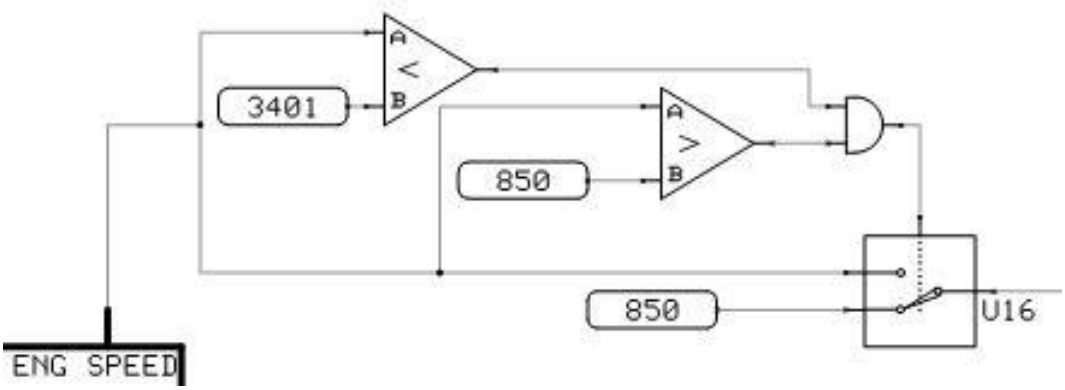
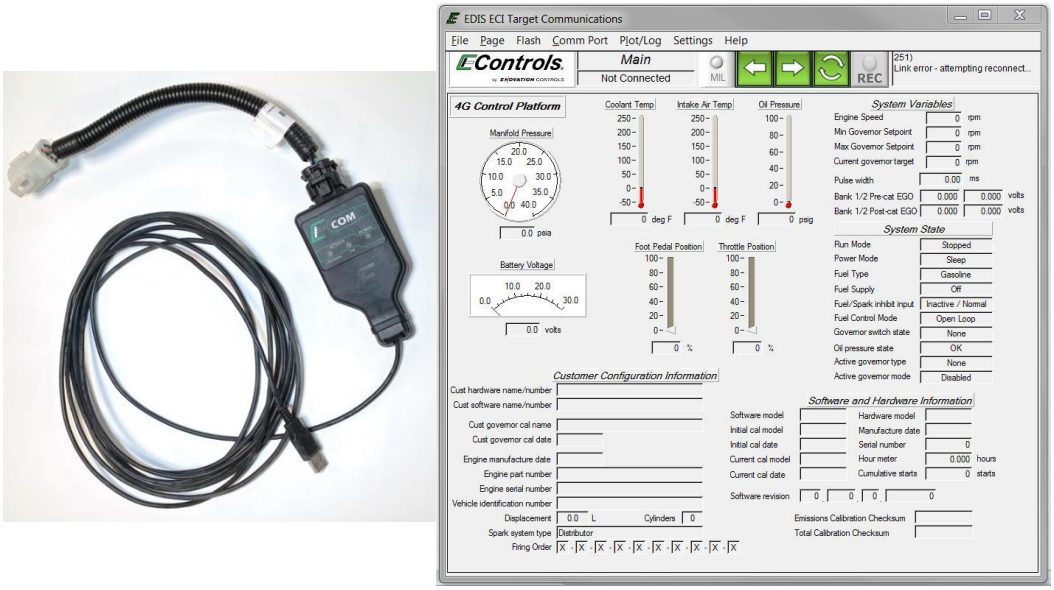



**Figure 2.12: Engine speed safety feature that reverts to idle speed when invalid engine speeds are sent.**

### *2.3.4 Engine Data Collection*

Engine data were collected for analysis using diagnostic software (E-Controls 4G Display) connected to the engine via a diagnostic Universal Serial Bus (USB) device (ECOM E2046000, EControls by Enovation Controls, San Antonio, Texas) (EControls Inc., 2008) both shown in Figure 2.13.




**(a)** **(b)**

**Figure 2.13: (a) The ECOM E2046000 diagnostic tool used to connect to the engine. (b) E-Controls 4G Display diagnostic software interface used to view engine diagnostics**

## 2.4 Engine Results and Discussion

To evaluate the working of the Plus+1 guide application program, different methods were used. First, the TSC1 command message was collected directly from the controller, which was sending TSC1 commands with a total of five desired engine speed values (1000 rpm, 1500 rpm, 2000 rpm, 2500 rpm, and 3000 rpm). These engine speeds represent the majority of the normal operating engine speeds. This was done with the controller disconnected from the engine to prevent an incorrect message from damaging the engine ECU. The messages were collected using Plus +1 Guide CANKing (Kvaser Inc., Mission Viejo, California) depicted in Figure 2.14. The messages were then decoded via an Excel spreadsheet to determine if the message was broadcasting the correct desired

speed as shown in Table 2.1. Desired engine speeds outside the range of the speed failsafe feature were also sent to verify the TSC1 message reverted to the 850 rpm idle speed.

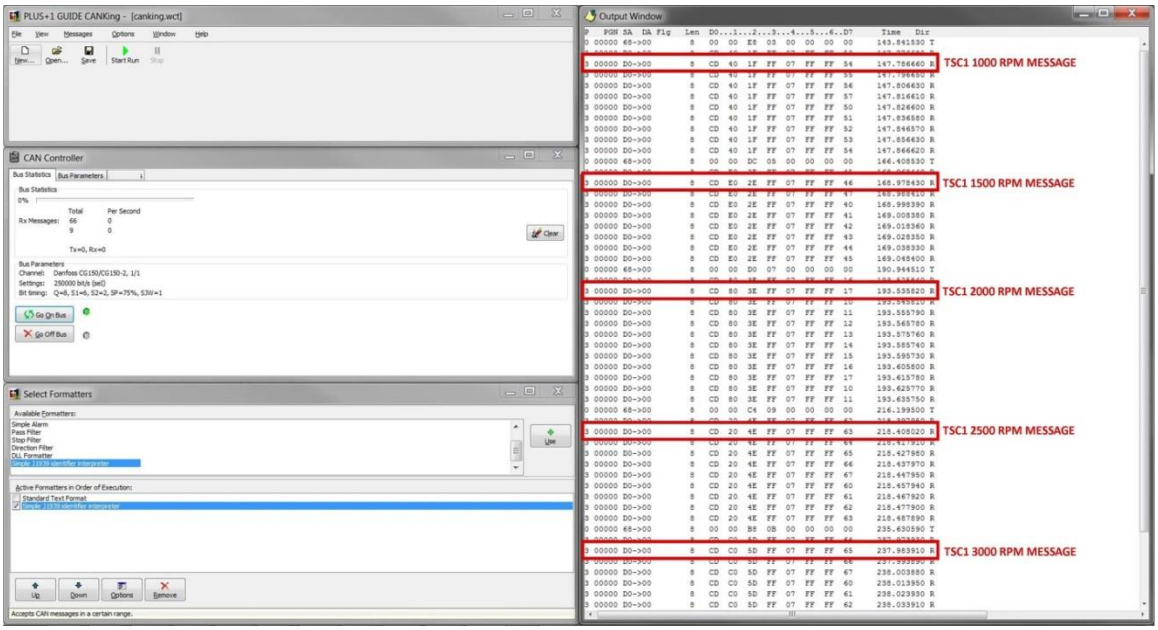



**Figure 2.14: Plus +1 Guide CANKing software interface with collected TSC1 messages.**

**Table 2.1: Decoded TSC1 CAN messages sent from the controller to the engine.**

| P | PGN | SA->DA | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | Time | Dir | Raw Bit Value | RPM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 0 | D0->00 | CD | 40 | 1F | FF | 7 | FF | FF | 36 | 349.323 | R | 8000 | 1000 |
| 3 | 0 | D0->00 | CD | E0 | 2E | FF | 7 | FF | FF | 47 | 383.802 | R | 12000 | 1500 |
| 3 | 0 | D0->00 | CD | 80 | 3E | FF | 7 | FF | FF | 13 | 416.192 | R | 16000 | 2000 |
| 3 | 0 | D0->00 | CD | 20 | 4E | FF | 7 | FF | FF | 62 | 441.141 | R | 20000 | 2500 |
| 3 | 0 | D0->00 | CD | C0 | 5D | FF | 7 | FF | FF | 64 | 459.41 | R | 24000 | 3000 |

Upon verification of the correct desired engine speeds within the TSC1 message, the Plus +1 controller was instrumented onto the engine. Diagnostic data were collected for 60 seconds at each desired engine speed using an engine diagnostic USB device (ECOM E2046000) to observe the actual engine speed within the diagnostic software (E-Controls

4G Display).

The engine speed data that were collected and analyzed are summarized in Table 2.2. As shown, the average trial speeds for all trials were within 1.5 rpm of the desired speed commanded by the embedded controller. The largest of the average standard deviations for each desired speed was approximately 8.25 rpm which occurred at a speed of 1500 rpm. It was also observed that larger trial standard deviations of engine speed occurred at lower speeds (e.g., 1000 and 1500 rpm in Table 2.2).

**Table 2.2: Engine speed diagnostic data**

| Desired Speed rpm | Trial | Average Trial Speed rpm | Overall Average Speed rpm | Trial Standard Deviation rpm | Average Std. Deviation rpm |
|---|---|---|---|---|---|
| 1000 | Run 1 | 1000.69 | 1000.62 | 7.646 | 7.785 |
| | Run 2 | 1000.58 | | 7.762 | |
| | Run 3 | 1000.58 | | 7.947 | |
| 1500 | Run 1 | 1501.15 | 1501.09 | 8.423 | 8.250 |
| | Run 2 | 1501.02 | | 7.985 | |
| | Run 3 | 1501.10 | | 8.341 | |
| 2000 | Run 1 | 2001.08 | 2001.25 | 5.555 | 5.461 |
| | Run 2 | 2001.45 | | 5.566 | |
| | Run 3 | 2001.21 | | 5.261 | |
| 2500 | Run 1 | 2500.84 | 2501.02 | 4.788 | 4.586 |
| | Run 2 | 2501.15 | | 4.387 | |
| | Run 3 | 2501.08 | | 4.584 | |
| 3000 | Run 1 | 3000.66 | 3000.60 | 4.807 | 4.955 |
| | Run 2 | 3000.56 | | 4.809 | |
| | Run 3 | 3000.59 | | 5.248 | |

The histogram in Figure 2.15 represents all collected engine speeds when the desired speed was set at 3000 rpm. All frequency distributions for the desired speeds were normal distributions that centered very closely to the desired speed.

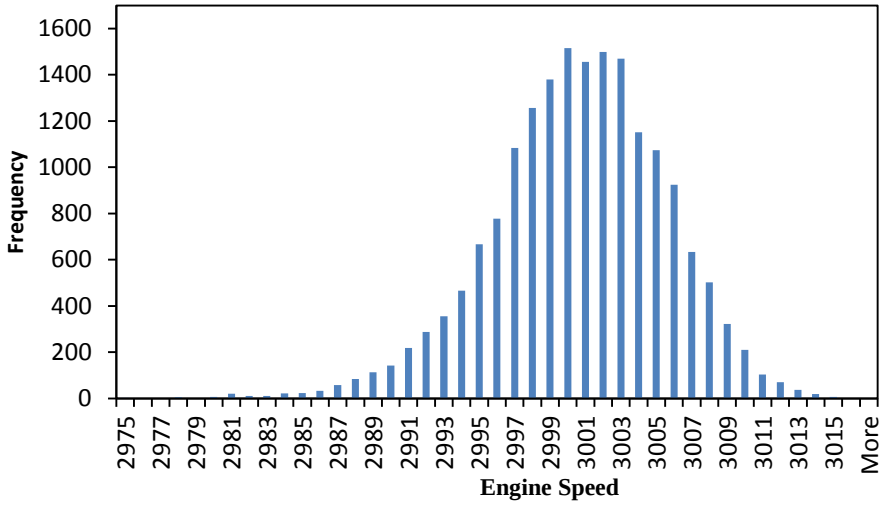



**Figure 2.15: The frequency of all collected engine speeds with the desired speed of 3000 rpm.**

Figure 2.16 shows the first 5 seconds of each run at the 3000 rpm desired speed in three trials. All of the data presented appeared to follow an oscillating pattern as the engine controller compensated for variability in fuel mixtures. The data presented showed that the measured engine speed was very consistent and close to that of the desired speed requested by the CAN message at all times.

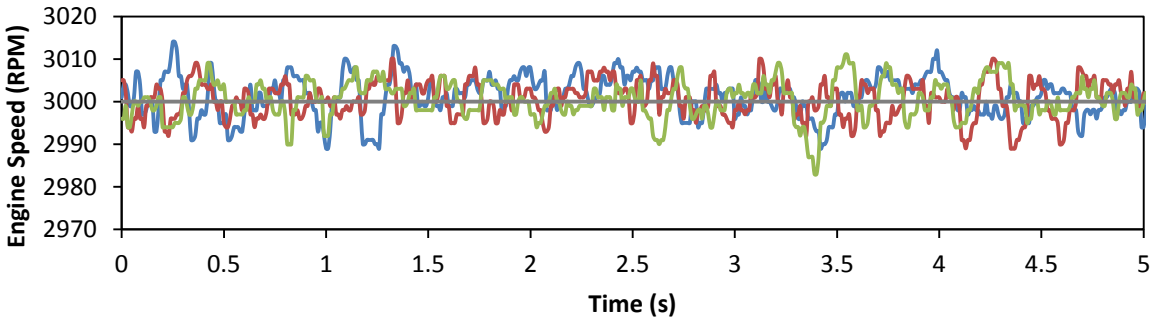



**Figure 2.16: Actual engine speeds for all three trials at a desired speed of 3000 rpm for the first 5 seconds.**

Figure 2.17 presents regression analysis between all recorded engine speed data and the corresponding desired engine speed requested by the embedded controller. The fit of the line is nearly perfect ($R^2$ =0.9999) showing that the actual engine speed aligned with the desired speed throughout the speed range tested. This indicated that there was very little variability at any point between the desired speed requested and the measured engine speed.

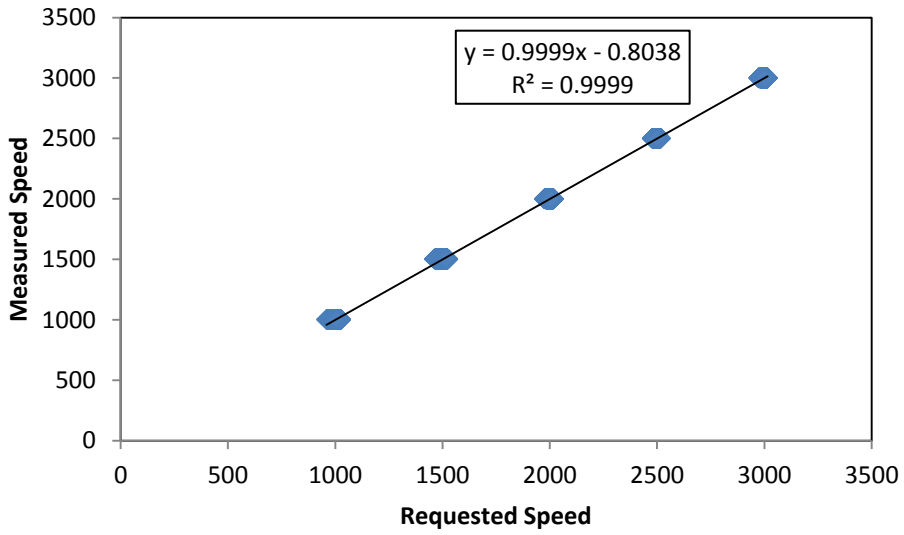



**Figure 2.17: Regression plot of all measured engine speeds plotted against corresponding desired speeds**

## 2.5 Engine Conclusions

The embedded application program was successfully programmed using the Plus +1 guide software, and implemented on the Danfoss Plus +1 controller. The controller with the embedded program was integrated onto the CAN bus of the robotic platform receiving messages from a computer and sending messages to the engine ECU.

All requested engine speeds sent in conjunction with the embedded CAN application program were verified to be correct using Plus +1 CANKing and Excel to collect and decipher the TSC1 message. Both the TSC1 message and the proprietary control message were further verified to be working when the engine speed data were collected. When collecting the diagnostic data, the standard deviation of all engine speed data collected during the trial runs was within 10 rpm of the desired engine speed provided by the TSC1 message for all speeds. The added safety feature to protect the engine from invalid engine speeds was also verified to work correctly. Using industry proven controllers and implementing this control system onto a CAN bus network was very effective.

# Chapter 3 Hydrostatic Transmission Selection and Control via an Embedded Control Application Program for Flex-Ro.

## 3.1 Hydrostatic Transmission Introduction

In mobile robotic applications where it is too expensive or difficult to transfer power to drive wheels via a mechanical linkage, there are limited options. When transmitting power to multiple remote drive components on a machine there are three viable options that can be utilized. The first option is direct current electricity from a stored power source, such as a battery. This method is commonly implemented on small robotic platforms (Bak and Jakobsen, 2004; Bawden et al., 2014; Hansen et al., 2013). The second option is using alternating current electricity produced by a generator, or using a combination of power generation and storage (Bangert et al., 2013; Gonzalez-de-Soto et al., 2016). This method provides significant torque to drive the machine. However, the components and control required can be difficult to implement. The third option for transferring power to the wheels is using a hydraulic drive system. Hydrostatic drives have been industry proven and heavily used in agriculture (Dieter Kutzbach, 2000). Multiple robotic platforms have implemented hydrostatic drives (Bakker et al., 2010; Godoy et al., 2012b; Zhang et al., 2016). These systems produce sufficient drive torque while allowing multiple control options.

## 3.2 Hydrostatic Transmission Objectives

The specific objectives of this project were to:

(1) Implement a hydrostatic transmission system that can adequately control the pumps and motors to operate the drive wheels of Flex-Ro.

(2) Develop and evaluate an embedded controller application that can generate CAN messages for operating the hydrostatic transmission system.

## 3.3 Hydrostatic Transmission Implementation

The hydrostatic transmission design is important for the Flex-Ro to maintain adequate operation in varying field conditions. The major components of the hydrostatic transmission system that are controlled by the CAN messages, auxiliary hydraulic components and the CAN controller application are discussed in detail in this section.

### *3.3.1 Pump*

The type of pump selected for the system was of utmost importance as it is the heart of the hydrostatic transmission. While there were many different types of pumps available for hydraulic systems, a closed loop design between the pump and motors was desired for mobile applications. This system pulls hydraulic fluid from the return side of the system, while supplementing this with reservoir fluid when needed. A closed loop system allows the bulk of the hydraulic oil to be stored within the system itself, permitting reduced reservoir sizes. The smaller hydraulic fluid reservoir is important for weight and space requirements on a vehicle platform. However, these systems circulate mostly the same fluid causing the heat being produced in the circuit to be retained and compounded, which can result in a rapid fluid temperature increase. This is alleviated through the use of a heat exchanger cooling the fluid that has passed through the relief valves and case drains. Furthermore, a bidirectional variable displacement pump obviates the need for additional valves which produce unneeded friction and heat within the system, as well as increasing the number of components that must be controlled. Lastly, the pump should be

a variable displacement axial piston pump, in series with a charge pump, in order to handle high oil pressures while maintaining control of the flow rate through adjustment of the swash plate.

A transmission with a single pump running four parallel drive motors would have significant limitations. The main limitation is that when one or more motors encounters a significantly greater load than the others, the flow rates to the individual motors will adjust so the pressure drop across all four motor branch circuits is the same. This will result in a gradation of flow rates with the greatest flow rate going to the motor with the smallest load and the smallest flow rate (possibly no flow) going to the motor with the greatest load. An example would be if one tire becomes stuck the fluid power delivered to the other wheels would increase, possibly to the point where they would exceed wheel slip and turn freely, stopping machine travel. Another example is that if one or more wheels experiences reduced load (e.g. loses contact with the ground) all flow could go through that motor, rather than the motors that require more torque. In this scenario a wheel that is suspended in air (over a low spot or channel) would spin freely, without load, taking power away from the remaining wheels.

The pump selected for the robotic vehicle application (shown in Figure 3.1) was a tandem variable displacement axial piston pump (Model: H1 Series Pump H1-T-045-R-A-N-A4-C1-N-D1-F-G1-TN-20-20-20-20-M-P-22-PN-NNN-S02, Danfoss, Ames Iowa). This hydrostatic pump is built with two in-line axial piston pumps and one charge pump, all driven by a single shaft. The charge pump feeds both axial piston pumps, thus reducing chances of cavitation. Each axial piston pump in the system has a maximum displacement of 45 $cm^3$ per revolution. The closed loop design of the hydrostatic transmission allowed

most of the fluid to remain within the loop; however the fluid losses from the loop were compensated for with a 24 $cm^3$ per revolution charge pump.

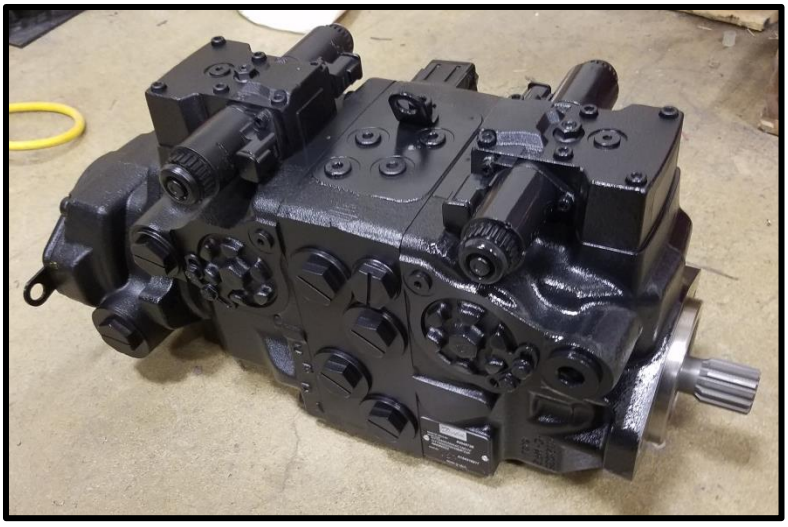


**Figure 3.1: Danfoss H1 series tandem variable displacement axial piston pump used for the hydrostatic transmission.**

### *3.3.2 Motors*

The two types of hydraulic motors that were considered for the drive wheels of this robotic platform were fixed and variable displacement. Variable displacement motors have some advantages. First, they can be individually controlled allowing for increased traction control as well as reduced stopping distance. Unfortunately, in order to take advantage of these improvements an advanced program must be created. This can be both time consuming as well as difficult to implement. Additionally, variable displacement motors can be supplied with fluid from a fixed displacement pump. However, there is added complexity and cost when implementing these motors.

Fixed displacement motors have the advantage of being simple to implement, as well as being more economical. These motors are very robust, and will not succumb to electronic

failure. Additionally, they do not need to be supplied with electricity which reduces wiring on the machine. This is important for the motors in particular as they will have the longest electric wire lengths and hydraulic hoses on the platform.

It was decided that an orbital hydraulic motor would be ideal for this application. This motor requires little maintenance, and can be equipped with both a wheel hub and a lock out brake for use when the robotic platform is inactive. Four hydraulic motors (Model: OMT 315 FX, Danfoss, Ames, Iowa) (Figure 3.2) were used as drive motors for the robotic vehicle. These motors have a displacement of 326.3 $cm^3$/rev, a maximum speed of 380 rpm, and a maximum continuous torque of 950 N·m.

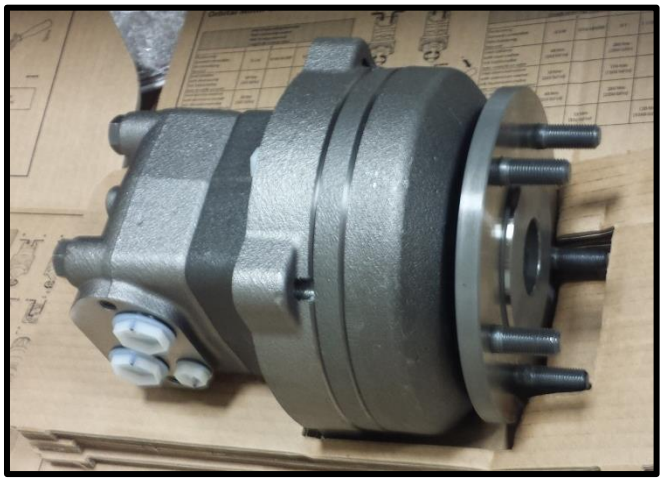


**Figure 3.2: Danfoss OMT 315 FX orbital hydraulic motor with wheel hub and brake.**

### *3.3.3 System*

The hydraulic system schematic (Figure 3.3) was composed of the tandem axial piston pump and four orbital motors. The axial piston pump was coupled with a charge pump that was directly coupled and mounted to the engine. The charge pump drew oil from a hydraulic reservoir, and then forced it through a filter prior to it entering the tandem axial

piston pump. This allowed the pump to compensate for leakages and case drains in the closed loop system. Each of the pumps provided flow to two motors in parallel, positioned at opposite ends of a diagonal pattern on Flex-Ro. All case drains were connected back to a heat exchanger that cooled the oil. Finally, the outlet of the heat exchanger sent fluid back to the hydraulic reservoir. It should also be noted that the H1 tandem pump was equipped with a common case drain port, which included the high pressure reliefs as well as charge pressure relief, which was connected into the hydraulic lines connected to the motor case drains.

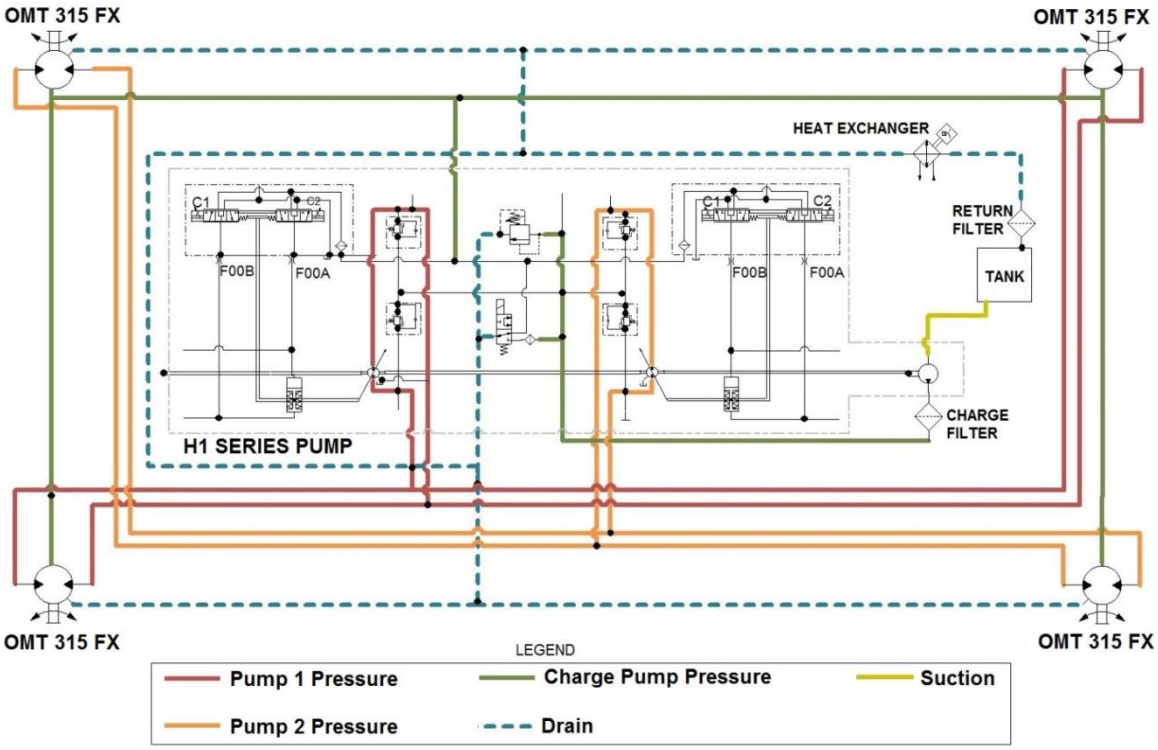



**Figure 3.3: Flex-Ro Hydrostatic transmission schematic diagram.**

With the charge pump discharging 24 $cm^3$ per revolution of hydraulic fluid, the flow rate at maximum speed (3500 rpm) was calculated to be around 84 L/min. The reservoir had a total fluid capacity of around 114 L, which could contain approximately 1.35 minutes of

maximum charge pump flow rate. It is also important to note that normal pump operating speed is expected to be approximately 2400 rpm, providing approximately 58 L/min of flow from the charge pump, so the reservoir could contain approximately 2 minutes of normal pump flow rate. The additional volume of fluid and tank surface area aids the heat exchange capabilities of the hydraulic system. A custom reservoir (Model: SS3297UNE, American Mobile Power, Fairmount, Indiana) (Figure 3.4) was created based on the size constraints of the platform.

**Figure 3.4: Hydraulic oil reservoir**

A heat exchanger was equipped with an electric fan (MAR-32-209341, Thermal Transfer Products, Racine, Wisconsin) that operates from a 12 Volt DC power source was sized and added to the system (Figure 3.5). This heat exchanger included a 414 kPa bypass, which was especially important to avoid building excessive pressure at the case drains of the pump and motors.

**Figure 3.5: Heat exchanger for the oil in the hydrostatic transmission.**

Additionally, the heat exchanger was equipped with a sensor port to allow for a stand-alone fan controller (MagHex Fan Controller, Thermal Transfer Products, Racine, Wisconsin) that was added to the system (Figure 3.6). This fan controller measured the fluid temperature exiting the heat exchanger, and turned the fan ON and OFF accordingly. The controller had various modes and the temperature was set with a magnetic adjustment tool.

**Figure 3.6: Controller for the electric fan of the heat exchanger in the hydrostatic transmission.**

There were four components in the system to remove particulates from the fluid before it enters the pump. A charge filter head (Model: HMK03 P179460, Donaldson,

Bloomington, Minnesota) with an 11 micron filter (Model: HMK03 P170311, Donaldson, Bloomington, Minnesota) and a 345 kPa bypass and visual blockages indicator was positioned after the charge pump (Figure 3.7). This is referred to as a remote full flow filter as the full flow of the charge pump flowed through the filter to remove particulates before the fluid entered the axial piston pumps.


**Figure 3.7: Remote full flow filter assembly to filter all fluid before it enters the axial piston pumps of the hydrostatic transmission.**

An in-tank filter housing (Model: WL15 P574231, Donaldson, Bloomington, Minnesota) with blockage indicator (Model: X011059, Donaldson, Bloomington, Minnesota) with a 5 micron element (Model: WL15 P566270, Donaldson, Bloomington, Minnesota) (Figure 3.8) was fastened to the top of the reservoir, with the return line from the heat exchanger feeding into it. All fluid returned through the system drains flowed into the reservoir through this filter.

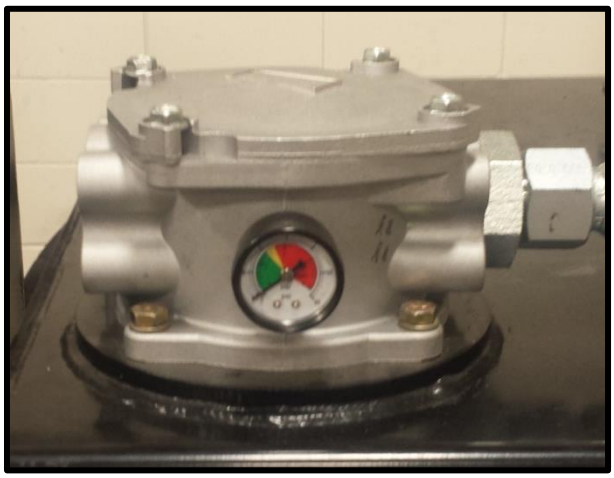


**Figure 3.8: Tank Return Filter**

Furthermore, a suction filter (Model: SS-023P104/P200-0-1, Magnaloy, Alpena, Michigan) was installed on the outlet of the reservoir preventing larger particulates from entering the charge pump, while the reservoir itself had a small strainer in the fill port to prevent large debris from being added to the system while filling the reservoir.

The pump selected was factory set with pressure limiters at 20,000 kPa (Danfoss, 2015). This condition required all hydraulic hoses and tubing to be rated for a minimum operating pressures of at least 20,700 kPa. In addition, it is important to note that the main lines from the axial piston pumps sized at hydraulic dash size 16 were reduced to dash size 12 hoses when dividing the flow to both motors. This practice is permissible when dividing flow from a single pump between two motors. Dash size is the measure of inner diameter of the hose in sixteenths of an inch (Cundiff, 2001). The brake lines were sized at dash 4 as they are only required to build brake releasing pressure. The drain lines were sized at dash 4 from the motors and dash 12 from the pump, these increased in size when coupled together to accommodate the combined flow ending in a dash 16 size running to the heat exchanger. All were sized based on maximum flow and velocity in the

lines (Goering, 1992).

### *3.3.4 Hydrostatic Transmission Control Components*

The Danfoss hydrostatic pump was controlled electronically. One important component of the pump is a built in Control-Cut-Off (CCO) valve. This valve highlighted in Figure 3.9 is a spring return, solenoid actuated, 2 position, 3 way cartridge valve installed on the side of the pump that governs flow to the Electronic Displacement Control (EDC) valve sets. A 12 Volt digital signal must be implemented to control this valve. In its normal de-energized state, the valve connects the EDC line to the case drain, thus relieving all pressure in the line. When energized the valve directs flow from the charge inlet to the EDC line building pressure used for controlling the swash plate. Additionally, a line is ported into the pump that connects to the controlled EDC line to be used for the spring applied brake release. Again, this is very useful as the de-energized state permits the pressure in the brake line to bleed to the return line, thus allowing the spring to reapply the brake to the wheel hub on the hydraulic motor.

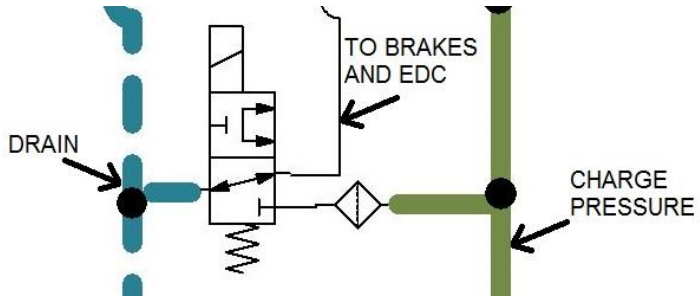



**Figure 3.9: Schematic diagram of the Control Cut Off (CCO) valve portion of the Danfoss H1 series hydrostatic transmission pump.**

Each of the axial piston pumps is equipped with an EDC (Figure 3.10), allowing for separate, dedicated control of each pump. The EDC consists of two 12 Volt solenoids on

both sides of a three position, four-way valve.

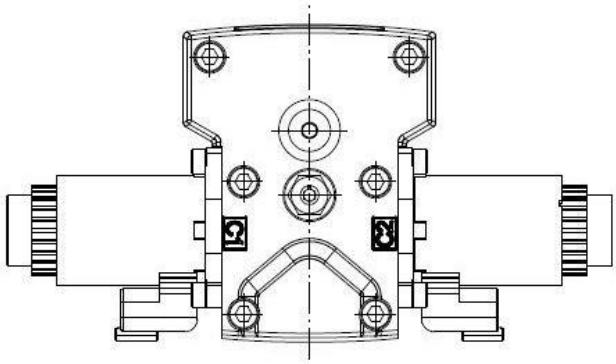


**Figure 3.10: Electronic Displacement Control unit used for control of each pump. (Danfoss, 2015)**

Each solenoid was supplied with a Pulse Width Modulation (PWM) signal resulting in varying current from 640 mA to 1640 mA (Danfoss, 2015) controlling the amount of actuation force of the solenoid, in turn varying the displacement of the pump (Figure 3.11). This PWM signal was able to proportionally adjust the solenoid via the resulting current output created by the varying duty cycle of the signal. However, when a signal was sent to one of the solenoids on the EDC the opposite solenoid was required to receive an input of 0 volts to ensure proper control. Switching which solenoid was receiving the control PWM signal reversed the direction of flow through the pump. As a result of each pump being equipped with a designated EDC, both pumps were able to be controlled individually; however, during most operation, the pumps receive the same signal to ensure the same flow rate is sent to each pair of parallel-connected motors.

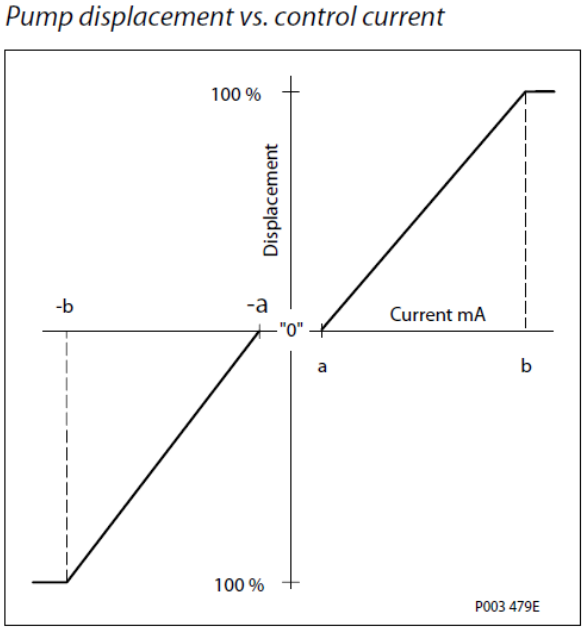



**Figure 3.11: Pump Displacement vs Current Output sent to the Solenoid for control of the H1 series pump displacement (Danfoss, 2015)**

Furthermore, this EDC has a manual override control that allows manual adjustment of the control valves during troubleshooting. The EDC varies the swash plate angle by allowing fluid to flow through the control valve to the spring-centered servo which is mechanically linked to the swash plate. This servo is centered at the zero degree neutral position, and allows variation in both directions resulting in bi-directional pump flow. Additionally, a rod is connected to the servo that provides feedback to the solenoids allowing for better control (Figure 3.12). F00A and F00B in the figure are flow restrictive orifices, that smooth the actuation of the swash plate.

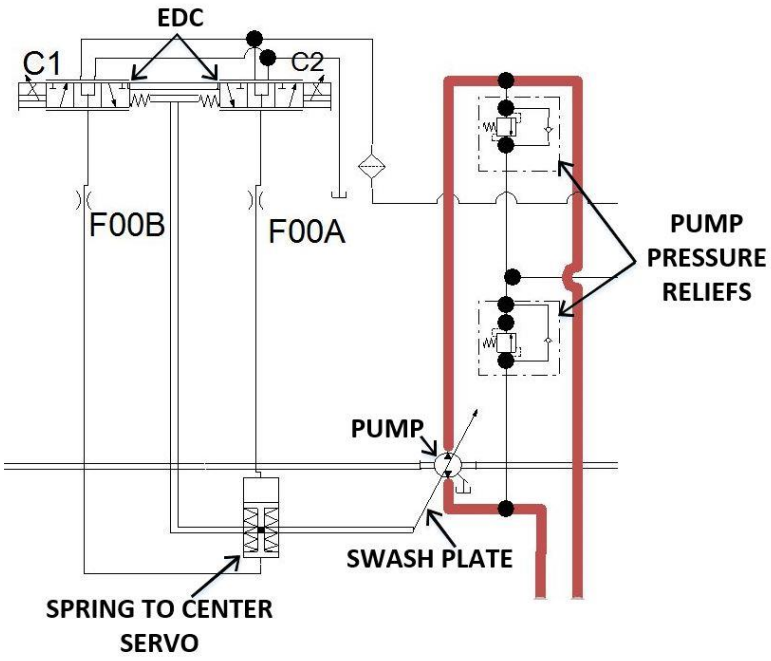



**Figure 3.12: Pump Control Schematic displaying all components used for pump displacement control.**

### *3.3.5 Embedded CAN Application Program*

A program was developed in Danfoss Plus +1 Guide to be imbedded on a Danfoss controller for full control of the EDCs, and consequently, the pumps. An 8 byte CAN message (Figure 3.13) with a mask and ID of 333 was created to allow for full control of the EDC, consequently controlling the pump. The D0 byte position controlled the functional mode of the pumps. The function mode for a value of 0 in the D0 byte position transmitted the same PWM for both EDCs. A value of 1 in the D0 byte position resulted in the PWM signal being sent only to the EDC on pump 1 while a PWM signal of zero was sent to the EDC on pump 2, effectively stopping the flow from pump by setting the swash plate angle to zero. A value of 2 in the D0 position resulted in the PWM signal for

pump 2 to be sent to the EDC on pump 2 and a PWM signal of zero was sent to the EDC on pump 1. The D1, D2, and D3 bytes control the direction and speed of pump 1. A value of 0 in the D1 byte position results in the pump outputting flow in the forward direction. A value of 1 in the D1 byte position results in the pump outputting flow in the reverse direction. Additionally, the D2 and D3 byte positions form a 2 byte integer with D3 as higher-order byte allowing values of 0 to 10,000. Bytes in the D4, D5, and D6 positions worked in the same manner as the D1 through D3 bytes respectively, but controlled pump 2. It is important to note that the bytes concerning pump 2 were only used when the D0 or mode byte was set to 2, therefore only controlling the second pump. When the D0 mode was set to 0 (both pumps receiving the same PWM signal) the speed and direction for both pumps were set using the values for pump 1 (D1, D2, and D3). The D7 byte position was populated with a 0 or 1 to actuate the CCO valve, with a value of 0 de-energizing the CCO solenoid (directing brake and EDC lines drain to the reservoir), and a value of 1 energizing the CCO solenoid (brake and EDC lines receiving pressure and flow from the charge pump).

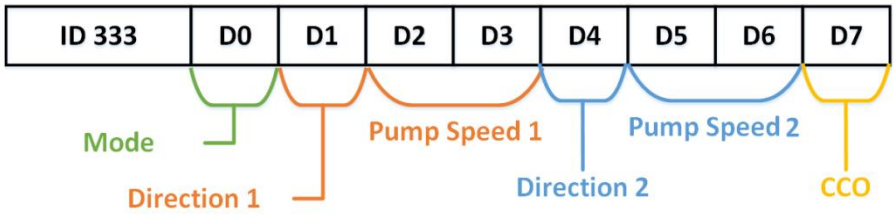



**Figure 3.13: Proprietary hydraulic control CAN message arrangement**

A CAN function block within Plus +1 Guide allowed the Danfoss controller (MC024-110, Danfoss, Ames Iowa) (Figure 3.14) to receive the hydraulic control CAN message. The function block received and parsed the message (Figure 3.15), then sent the

individual byte values of the message into a programmed Pump CAN Controller block that was created to implement the control messages.

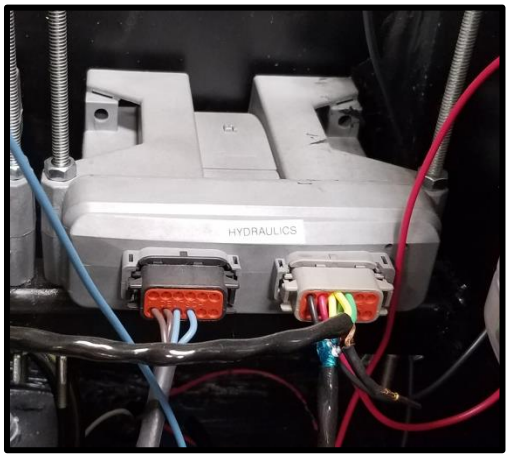



**Figure 3.14: Danfoss MC024-110 programmable Plus +1 controller.**

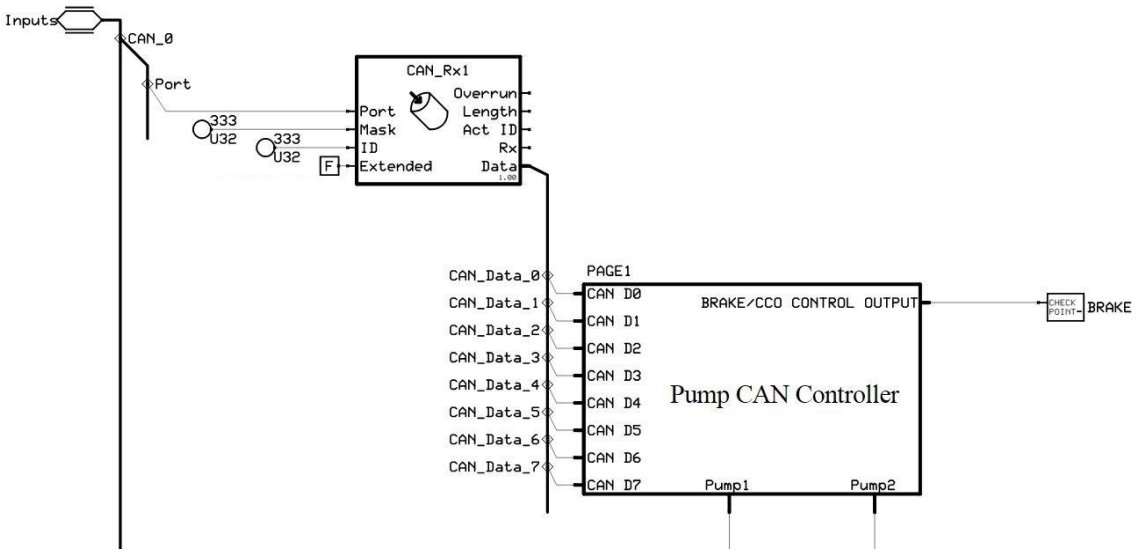



**Figure 3.15: Shell page for receiving CAN message.**

As previously explained, the functional control mode byte controlled the mode the system was in, therefore controlling which PWM message was sent to which pump EDC. Additionally, simple arithmetic was performed to recombine the two byte speed message

into a 0-10,000 value integer. A positive or negative sign was assigned to this value based the value in the respective pump direction control byte (Figure 3.16).

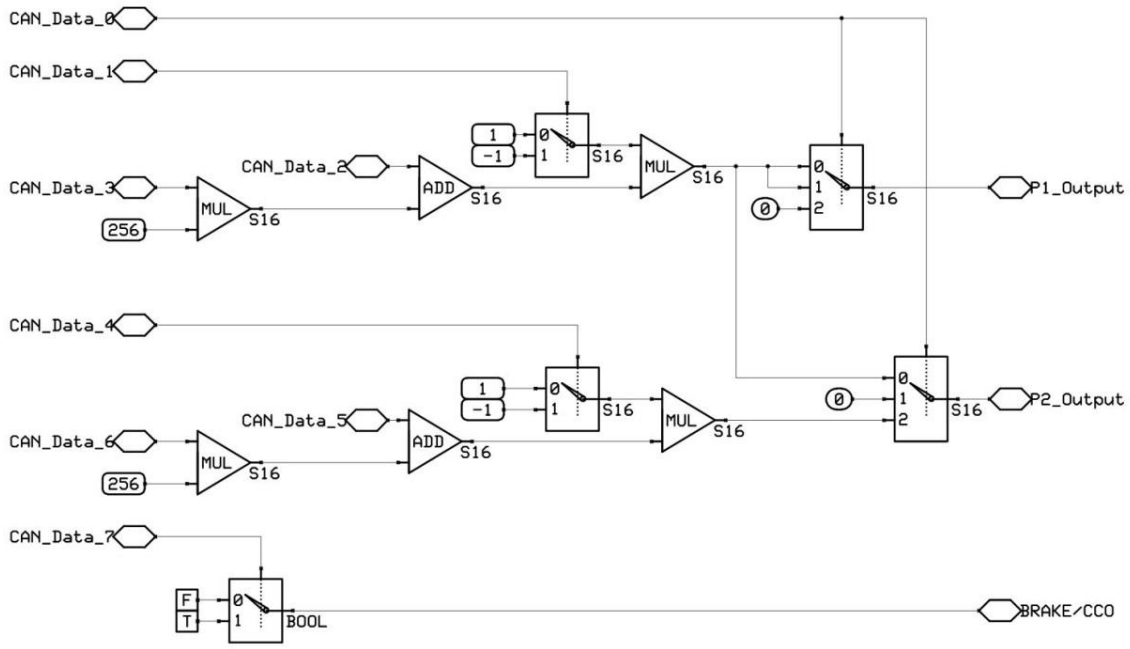



**Figure 3.16: Program to decode hydraulic control CAN message.**

Once the CAN message was deciphered and converted into a command signal to control the PWM output to the pump, each value was routed into an H1 EDC pump hardware compliance block (Figure 3.17). This compliance block ensured that only one of the two solenoids on each EDC was receiving a PWM signal (Danfoss, 2005). It also contained a feedback loop allowing the program to respond to changes in the system. This program was developed to be implemented onto the MC24-110 Danfoss controller. The inputs for the compliance block were power and status of the output pins for the feedback. This program was set up such that the first pump Out_A was connected to C2p09 and Out_B to C2p10. Additionally, the second pump used the C2p11 pin for Out_A and C2p12 for Out_B. Additionally; digital inputs were connected to the Enable pin on the compliance

block. These inputs were connected with an AND statement and correlated to emergency stop buttons on the machine. Since E-Stops are normally closed switches, once any E-Stop was pressed there was no longer be a Boolean value of "True" at the Enable pin, thus forcing the swash plates to the neutral position.

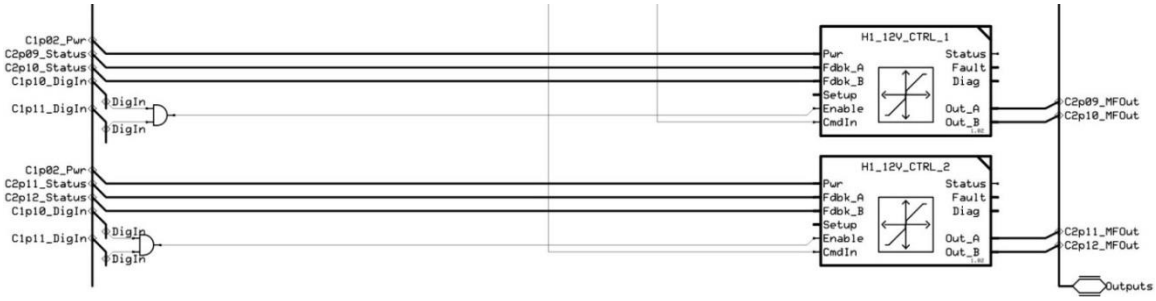



**Figure 3.17: H1 pump function block control.**

Manufacturer data were given in terms of swashplate angle in degrees; however our control messages were set to a 0 to 10,000 input value corresponding to 0 to 100% of swashplate actuation. The maximum angular value of the swash plate was 18 degrees with the minimum being zero. A proportional relationship was used to determine theoretical control values that corresponded to each of the manufacturer's provided swash plate angle values (Table 3.1).

**Table 3.1: Theoretical Control Messages Derived from the Swashplate Angle.**

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Swashplate Angle (deg) | 2.89 | 4.95 | 6.37 | 7.41 | 9.43 | 11.83 | 13.80 | 16.10 | 18.00 |
| Theoretical Control Message | 1607 | 2751 | 3537 | 4114 | 5238 | 6570 | 7663 | 8950 | 10000 |

### *3.3.6 Data Collection*

Data were collected to verify operating speed conditions declared by the manufacturer. Hall Effect gear tooth sensors (GS1007, Cherry, Pleasant Prairie, Wisconsin) were implemented to collect rotational speed data from each wheel. These sensors were activated by an metal sprocket with 65 teeth, and a #35 pitch, with 5/8 inch holes drilled to permit fastening to the wheel lugs of the motor hub. Based on manufacturer guidelines (Cherry, 2016) the radial air gap from the tip of a sprocket tooth to the sensor was set to be roughly 1.5 mm. It received a 12 V DC supply from the data acquisition system. A 2.4 kΩ pull-up resistor was installed between the supply voltage and output lines as recommended by the manufacturer. The sensor produced an output sinking current of 20 mA, maximum. The sensor was secured to the motor using a simple mount fabricated from light gauge sheet metal, connected to one of the mounting bolt holes on the motor (Figure 3.18).

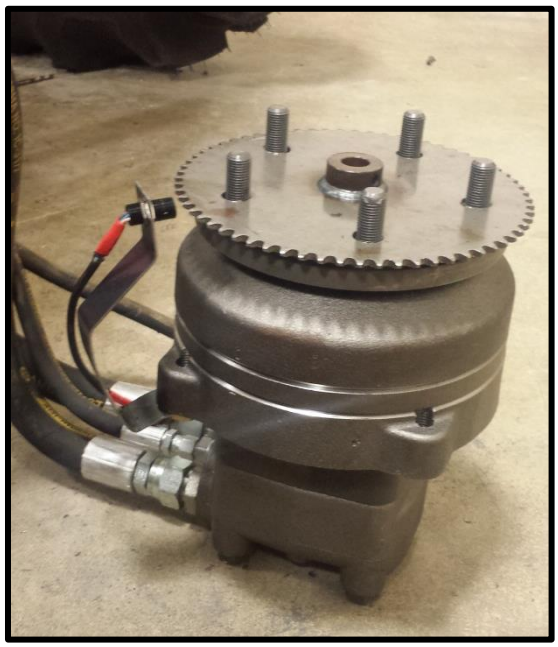


**Figure 3.18: Gear tooth sensor, mount, and sprocket.**

### *3.3.7 Data Collection Embedded Program*

A dedicated Danfoss controller (MC024-010, Danfoss, Ames, Iowa) (Figure 3.19) was used to implement a data collection program. This platform was selected based on the ease of use, as well as the number of inputs that were compatible with the sprocket tooth sensors.


**Figure 3.19: Danfoss MC024-010 programmable Plus +1 controller.**

A simple embedded program was developed to collect the data (Figure 3.20). This program read frequency values from the sprocket tooth sensors and processed them in a frequency to rpm function block. Pulses per revolution were set to be 65 based on the number of teeth on the sprocket attachment. The output value of the function block was an unsigned 32 bit integer (U32) value equal to measured rpm. These outputs were sent to value checkpoints so they could be directly read and logged using the Danfoss Plus+1 Guide Service tool.

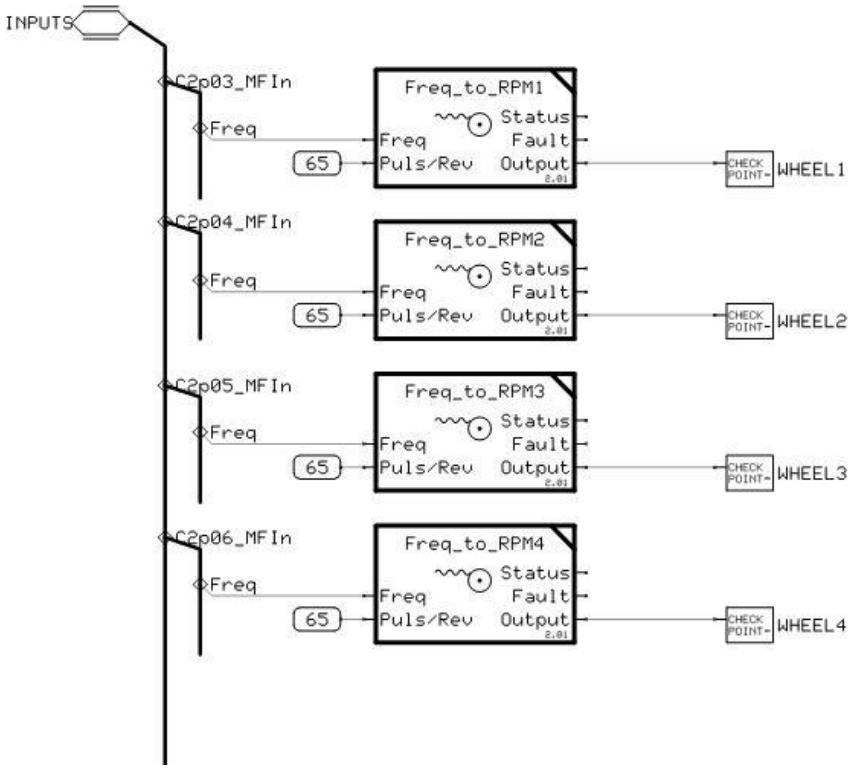



**Figure 3.20: Program to output wheel speed (RPM) based on the read gear tooth sensors.**

### 3.4 Hydrostatic Transmission Results and Discussion

The message sent to the hydraulic control application program during data collection was configured for mode 0, such that both pumps received the same control values for direction and speed. With this command all four wheels theoretically should have been rotating at the same speed. The pump control value was incrementally increased to each of the theoretical control message values provided by Danfoss, while wheel speed data were collected at 10 Hz. At each control message value, 10 seconds of steady-state wheel speed data were obtained after the end of the transient periods during which the pump swash plates were being adjusted.

Figure 3.21 shows that there was variation among the speeds of the different wheel motors. It is important to note that since each pump circuit had two motors connected in parallel the flow from each pump was divided between those motors, and not necessarily divided equally, especially if there were differences among the motor loads. Unequal division of the flow between motors would have resulted in one motor spinning faster than the other, especially since the test was performed with no load applied to the motors.

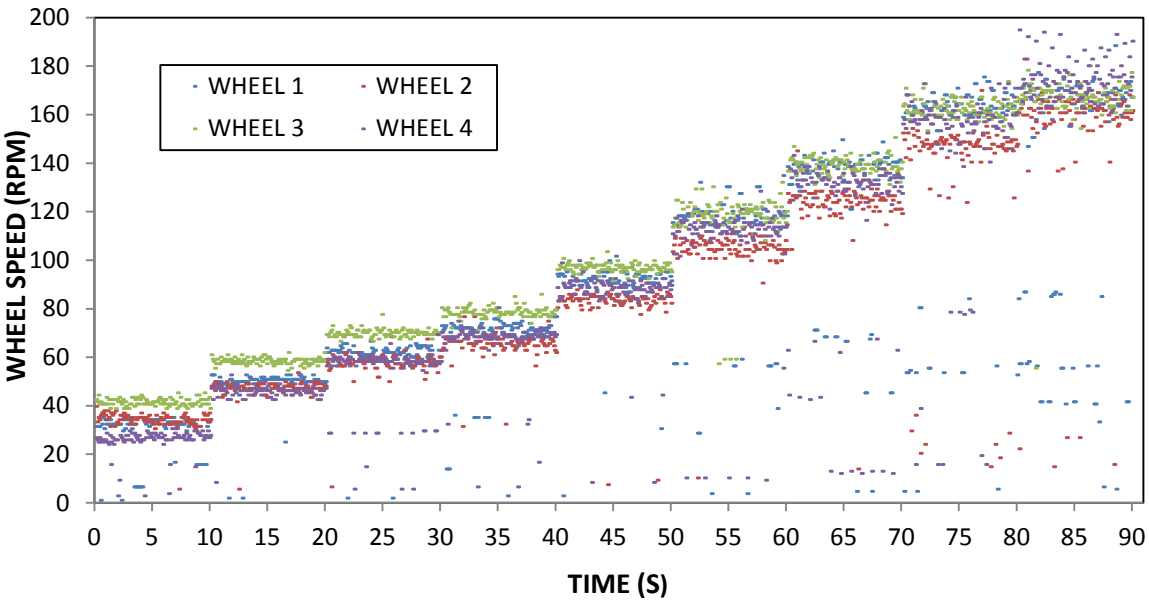



**Figure 3.21: Time series data of the measured wheel speeds**

The average wheel speed and standard deviation of the measured speed data from all four wheels for the duration of each 10 second intervals are shown in Table 3.2. The wheel speed standard deviation increased with the increase in speed. The increased speed caused higher deviations when the system was not under constant load. One reason for these increased variations can be attributed to the fact that the speeds of the two wheels driven by each pump tended to pulsate slightly from the division of fluid between the motors.

**Table 3.2: Measured wheel speed data**

| Control Message | Average Measured Wheel Speed (rpm) | Wheel Speed Standard Deviation (rpm) |
|---|---|---|
| 1607 | 32.38 | 8.24 |
| 2751 | 50.00 | 6.98 |
| 3537 | 60.26 | 10.57 |
| 4114 | 69.01 | 11.61 |
| 5238 | 88.83 | 11.76 |
| 6570 | 108.78 | 20.76 |
| 7663 | 125.72 | 26.47 |
| 8950 | 147.28 | 34.56 |
| 10000 | 156.03 | 36.13 |

The manufacturer also provided theoretical wheel speeds based on the swashplate angles. These speeds were shown in Table 3.3 with the corresponding theoretical control message that was created using the manufacturer's swash plate angle data.

**Table 3.3: Manufacturer wheel speed at theoretical control messages**

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Swashplate Angle (deg) | 2.893 | 4.952 | 6.367 | 7.405 | 9.429 | 11.83 | 13.79 | 16.11 | 18.00 |
| Theoretical Control Message | 1607 | 2751 | 3537 | 4114 | 5238 | 6570 | 7663 | 8950 | 10000 |
| Manufacturer Theoretical Wheel Speed (rpm) | 13.42 | 29.39 | 40.76 | 52.91 | 74.36 | 98.83 | 118.6 | 141.9 | 161.0 |

The manufacturer theoretical wheel speed was plotted alongside the average measured speed (Figure 3.22). Additionally, the plot includes error bars (in grey) for the average wheel speed plus or minus one standard deviation of the measured speed. The measured wheel speeds were noticeably greater than the manufacturer's wheel speeds for average

measured wheel speeds below 88 rpm, but the difference decreased as speed increased. The manufacturer provided data fell within one standard deviation of the measured data at the control message values of 6570 (average measured wheel speed of 109 rpm) and greater.

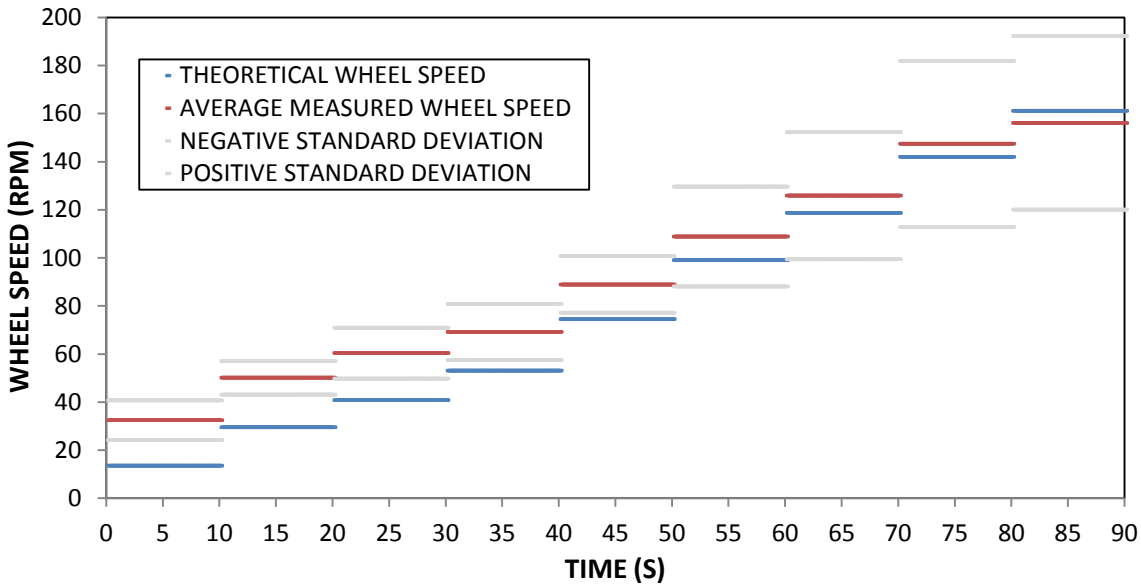



**Figure 3.22: Average measured wheel speed vs manufacturer theoretical speed**

The relationship between wheel speed and control message was plotted for both the manufacturer provided speeds, as well as the measured speeds (Figure 3.23). Again, the difference in the speeds reduces as the speed increases.

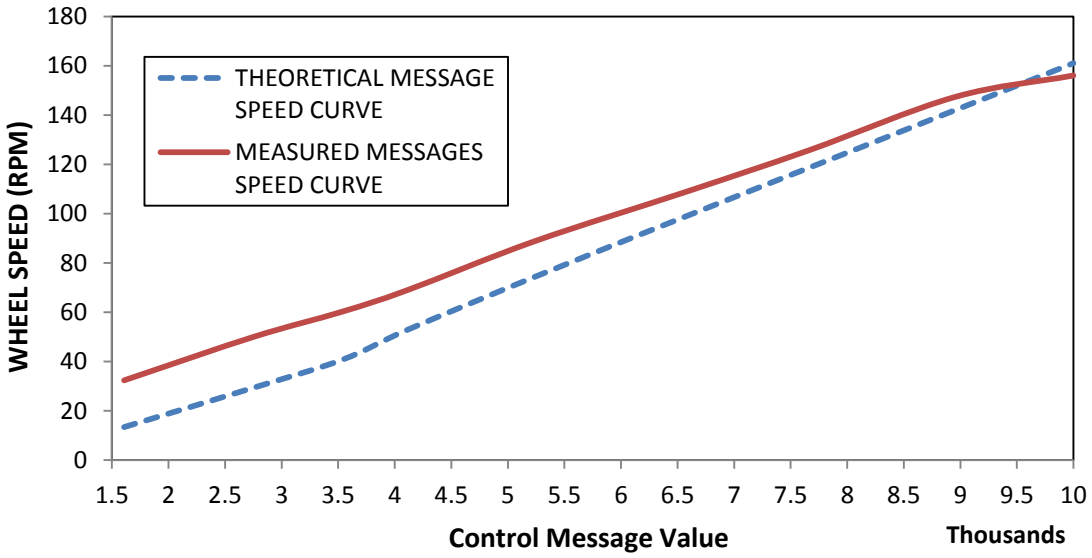



**Figure 3.23: Theoretical and measured control speeds**

The difference between the manufacturer supplied data and the measured data can be caused by a variety of sources. The manufacturer data assumes a perfect engine speed of 2400 rpm, therefore a constant flow rate from the pump. However, the engine speed, while robust and mostly steady, does vary some during operation. Additionally, the Danfoss PWM output pump controller block is proprietary to Danfoss, therefore the full extent of the solenoid variation is not known. Most important is that the control message was developed using the assumption that 0-18 degrees of swashplate angle directly correlated to 0-10,000 message input. However, it could also be that the control message input does not follow that trend directly given the mechanical forces required for swash plate actuation.

The robotic platform drive tires (Goodyear Titan 7.2-16 8 GY Power Torque TL, Quincy IL) (Figure 3.24) were mounted on steel rims. These tires were filled with solid setting foam that added considerable weight for improved traction. This foam is fairly rigid resulting in a stiff wheel with an overall diameter of 29.5 inches.

**Figure 3.24: Goodyear Tire**

Given the rigid state of the tire, it was assumed there would be limited tire deformation during operation, thus resulting in a rolling diameter close to the 29.5 inch standard diameter. Using this diameter, as well as the relationship between the control message and measured motor speed, the relationship between control message value and vehicle speed was determined (Figure 3.25).

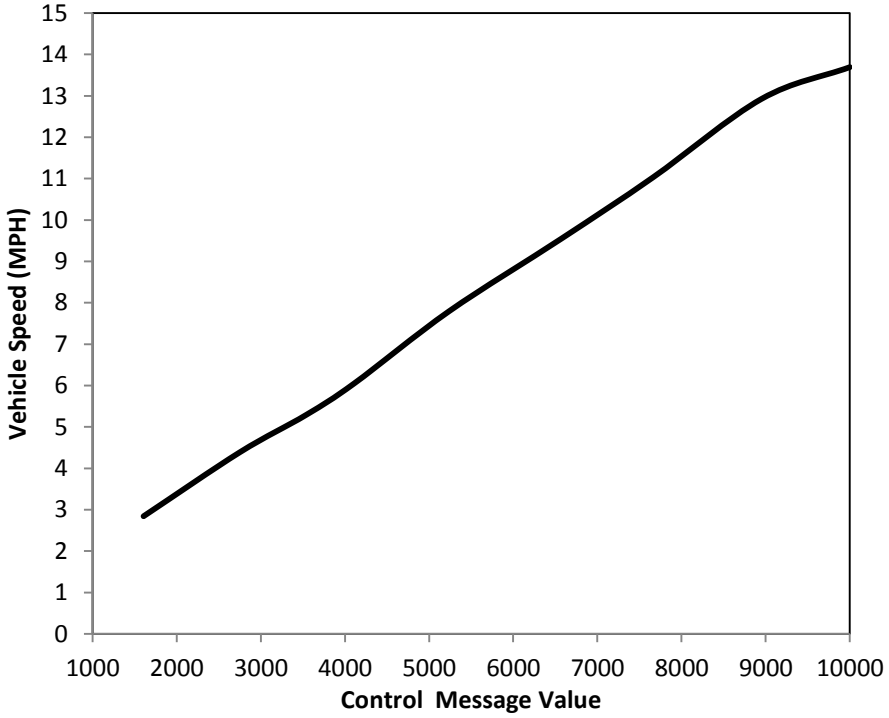



**Figure 3.25: Vehicle speed vs control message, calculated using measured wheel speed data**

### 3.5 Hydrostatic Transmission Conclusions

The robotic platform's hydrostatic transmission system was fairly complex with many electric-over-hydraulic components that required careful programming and electronic interfacing. The system successfully powered the propulsion of the platform through the use of the variable displacement pumps, and fixed displacement motors. Furthermore, the system was cooled through the use of a heat exchanger, along with a correctly sized hydraulic oil reservoir. The filters in the system protected components from damaging particulates that may be introduced to the oil, while the brakes on the motor provided additional safety for the platform. Most importantly, the system is controlled electronically using an embedded application control program, which is a priority for use on a robotic platform.

The hydraulic control application program worked as intended to control components of the hydrostatic system. The control architecture of sending CAN messages to an embedded controller to operate the system functioned sufficiently. This was verified through the collection of motor speeds during operation. However, it was determined that the theoretical speeds provided by the manufacturer did not correlate directly to the measured speeds at various calculated control inputs. Therefore, measured speeds should be used when developing vehicle speed controls, and more data should be collected under various loading conditions.

# Chapter 4 Frame Design of Flex-Ro.

The structure of Flex-Ro will support all critical components, as well as additional mechanisms used for auxiliary functions. It is important that the frame is able to support all components of the Flex-Ro.

## 4.1 Internal Main Frame

The internal main frame (Figure 4.1) was where all the components of the machine were mounted. It was important that Flex-Ro's frame provided the ability to support a substantial weight load. Additionally, the footprint of this frame should be kept at a minimum while providing required space for components. The absence of an operator on the machine allowed for a dramatic reduction in spatial requirements. The internal frame was supported by two main 2 x 6 x 3/16 inch rectangular carbon steel tubes running horizontally along the edge of the frame. Five structural tubes were welded to connect the main tubing. Additionally, mounting plates and support tubes were used to accommodate various components.

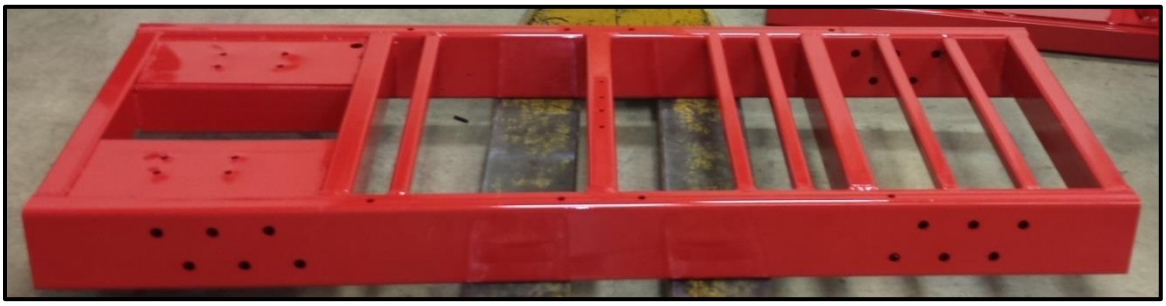


**Figure 4.1: Internal Main Frame**

A simple loading calculation was performed to evaluate the load strength of the system. This was modeled as a dual supported beam with a distributed load (Figure 4.2) and the

area moment of inertia was calculated based on the square tubing dimensions (Figure 4.3). Using the 2 x 6 x 3/16 inch rectangular carbon steel tubing with a yield value of 33,000 psi and a safety factor of 3, the maximum weight that can be supported per steel beam is over 2500lbs. The summation of the loads from two main rails on the internal main frame provide sufficient support for up to 5000 lbs. on that part of the structure, thus negating the need to do further analysis on the smaller internal steel tubes of the Internal Main Frame.

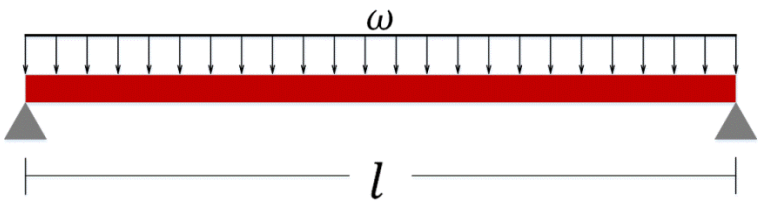



**Figure 4.2: Beam loading of main frame.**

$$M_{max} = \frac{\omega \cdot l^2}{8}$$

Where,

M = Maximum bending moment (in-lbs.)

l = length of beam (in)

ω = load (lb. $in^{-1}$)

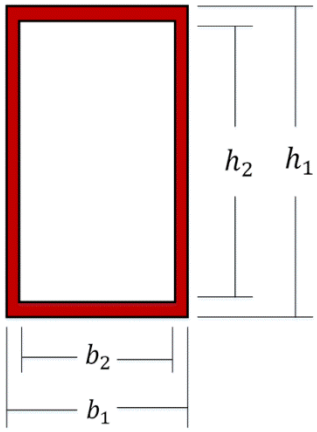



**Figure 4.3: Area moment of inertia value calculation for main frame.**

$$\mathrm{I} = \frac{b_1 \cdot {h_1}^3}{12} - \frac{b_2 \cdot {h_2}^3}{12}$$

Where,

I = area moment of inertia ($in^4$)

$b_1$ = width of tubing (in)

$b_2$ = width of inside of tubing (in)

$h_1$ = height of tubing (in)

$h_2$ = height of inside of tubing (in)

$$\sigma = \frac{M \cdot \mathrm{y}}{\mathrm{I}}$$

Where,

σ = stress (psi)

M = maximum bending moment (in-lbs.)

y = maximum acting distance away from neutral axis (in)

I = Area moment of inertia ($in^4$)

### 4.2 Slide Frames

The internal main frame was designed to be a standalone modular component that would be suitable for various frame configurations for future modifications to the Flex-Ro. To achieve variable width, a slide frame was designed to be attached to the internal main frame. The slide frame was intended to be connected to the internal main frame with bolts, while allowing a tube to slide in and out of the slide frame. This component was evaluated with a 2000 lb. distributed force within SolidWorks (2015, Concord, Massachusetts) to establish stress and deflection. It is important to note that once installed, the internal face of the slide frame aligned against the internal main frame and transmitting force and restricting movement. Figure 4.4 shows the maximum part stress to be under 12,000 psi providing a safety factor of around 3. Furthermore, the maximum deflection (Figure 4.5) in the part was evaluated to be a fraction of a millimeter.

| Name | Type | Min | Max |
|---|---|---|---|
| Stress1 | VON: von Mises Stress | 0.339418 psi<br>Node: 4115 | 11569.4 psi<br>Node: 2642 |

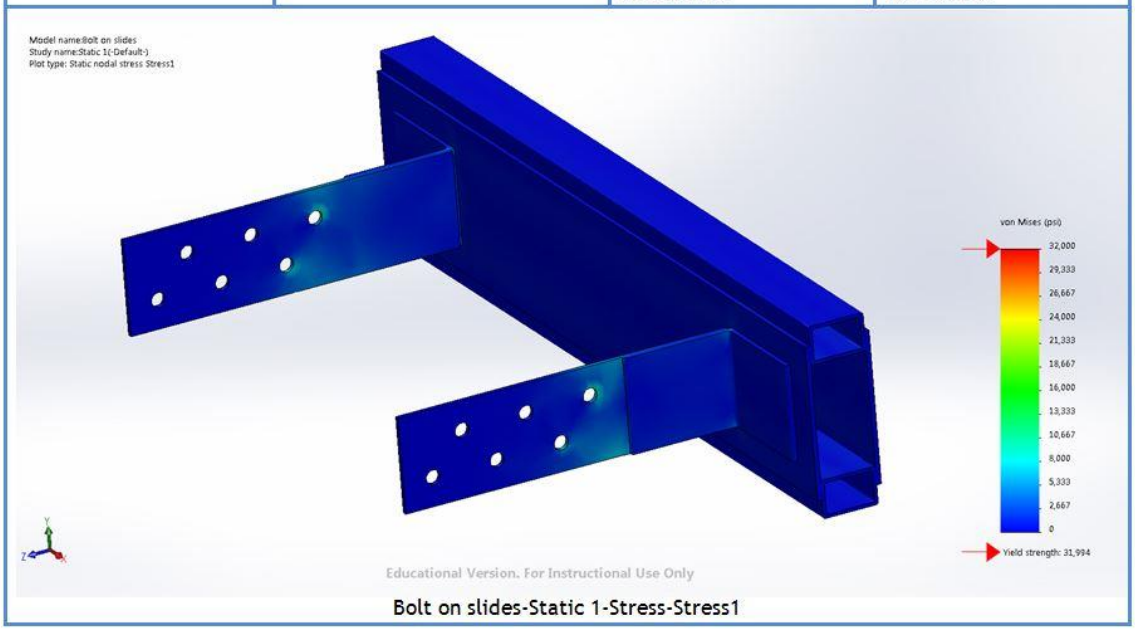



**Figure 4.4: Slide frame stress.**

| Name | Type | Min | Max |
|---|---|---|---|
| Displacement1 | URES: Resultant Displacement | 0 mm<br>Node: 1 | 0.169863 mm<br>Node: 16476 |

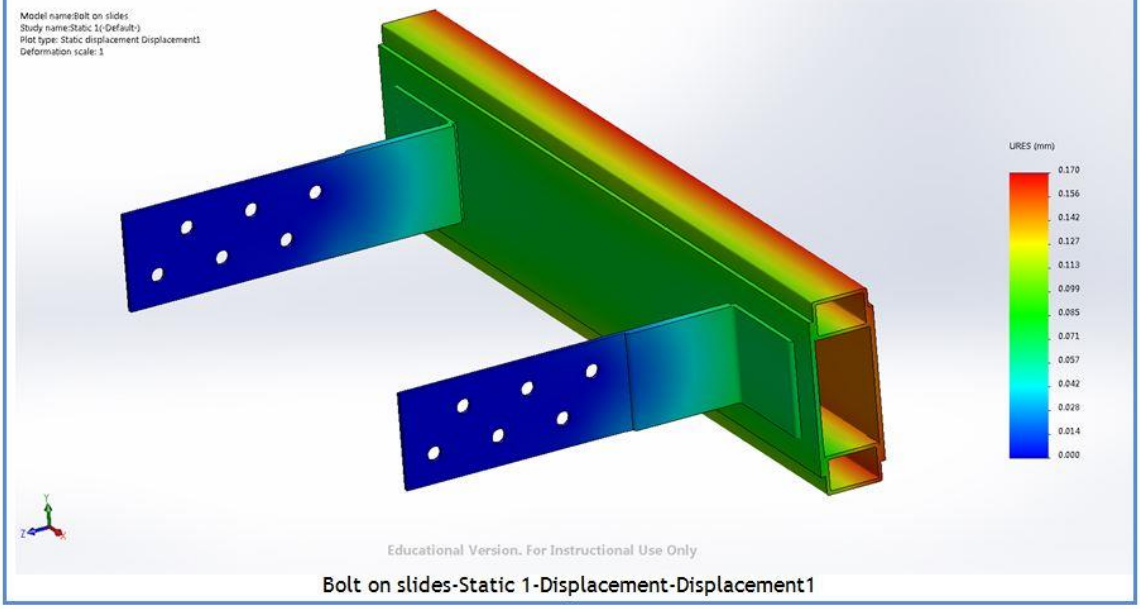



**Figure 4.5: Deflection of slide frame.**

The frame components were powder coated to provide a durable finish that would prevent weathering. The frame slides were fastened to the internal main frame with twelve ¾ inch grade 8 bolts (Figure 4.6).

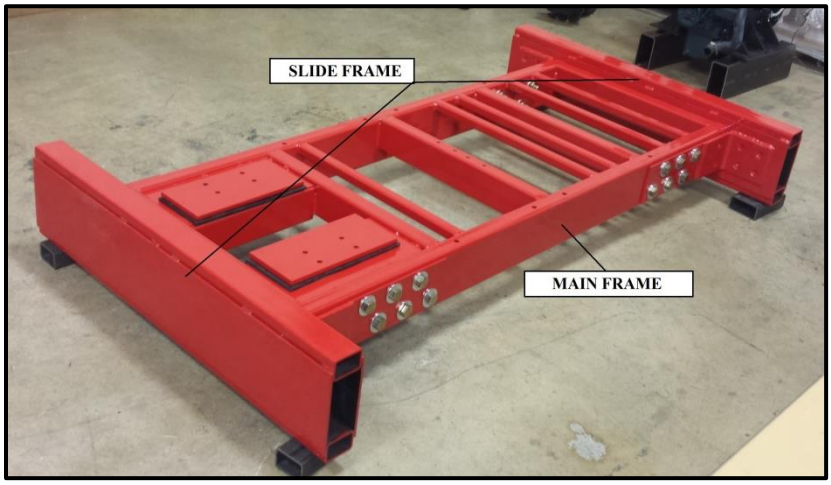



**Figure 4.6: Slide frame mounted to the main frame.**

## 4.3 Component Platform

A platform (Figure 4.7) was added for mounting electronics, as well as for securing a hydraulic tank. The platform was constructed of 1.25 inch square tubing with a 3/16 inch wall. This platform was secured to the main frame via eight 3/8 inch grade 8 bolts. The main advantage of the platform was to provide a mounting location for the hydraulic reservoir close to and above the pump.


**Figure 4.7: Component Platform with components mounted above.**

## 4.4 Collar axle

A collar axle was designed to slide into the slide frame, thus connecting the main frame to the uprights of the platform. This collar axel was designed to be fastened to both the slide frame, as well as the uprights, using ¾ in grade 8 bolts (Figure 4.8).

**Figure 4.8: Collar axle mounted into slide frame.**

## 4.5 Upright

Uprights were created to mate with each of the four collar axles while offering vertical adjustment in 6 inch increments. Four ¾ inch grade 8 bolts were used to lock the upright in place (Figure 4.9).

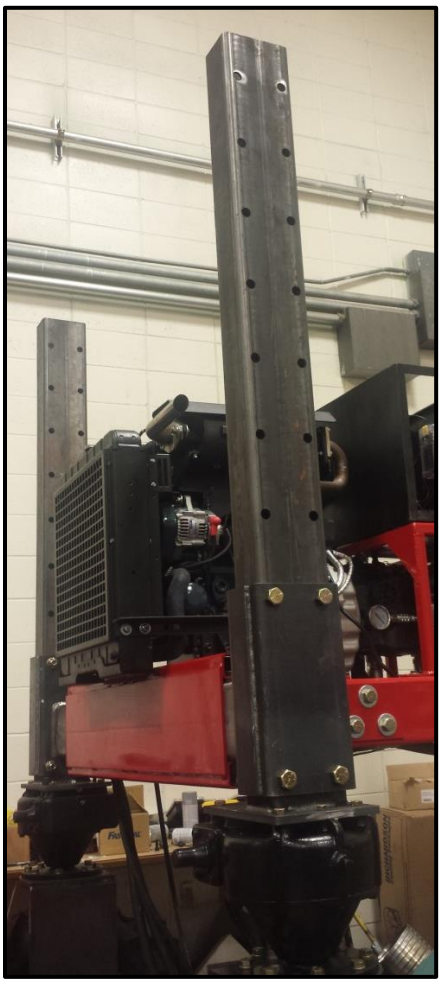


**Figure 4.9: Upright connected to collar axle.**

## 4.6 Wheel Mounts

On the bottom of each upright a ½ inch thick plate with laser cut holes was added to provide attachment to a steering gearbox. Connected to the gearbox was a wheel mount consisting of a laser cut ¾ inch plate weldment. Eight 9/16$^{th}$ inch diameter holes uniformly spaced around an eight inch concentric circle about a 6 inch diameter hole at the top of the wheel mount allowed the mount to be fastened to the steering gearbox. The 9/16$^{th}$ inch diameter holes have an expanding 45 degree taper on the bottom side of the wheel mount to permit lug fasteners to be used to aid in centering the mount. Near the

bottom of the wheel mount is a welded spacer ring with a 6.85 diameter inch hole surrounded by 4 concentric ½ inch holes to accommodate the attaching the hydraulic wheel motor. The spacer ring offsets the wheel from the wheel mount to center it in line with the gearbox. With 1000 pounds of force loaded onto the wheel mount in the SolidWorks simulation the maximum stress was 13654.5 psi resulting in a safety factor of 2.34 (Figure 4.10), while the deflection in the part was limited to less than 2 millimeters (Figure 4.11).

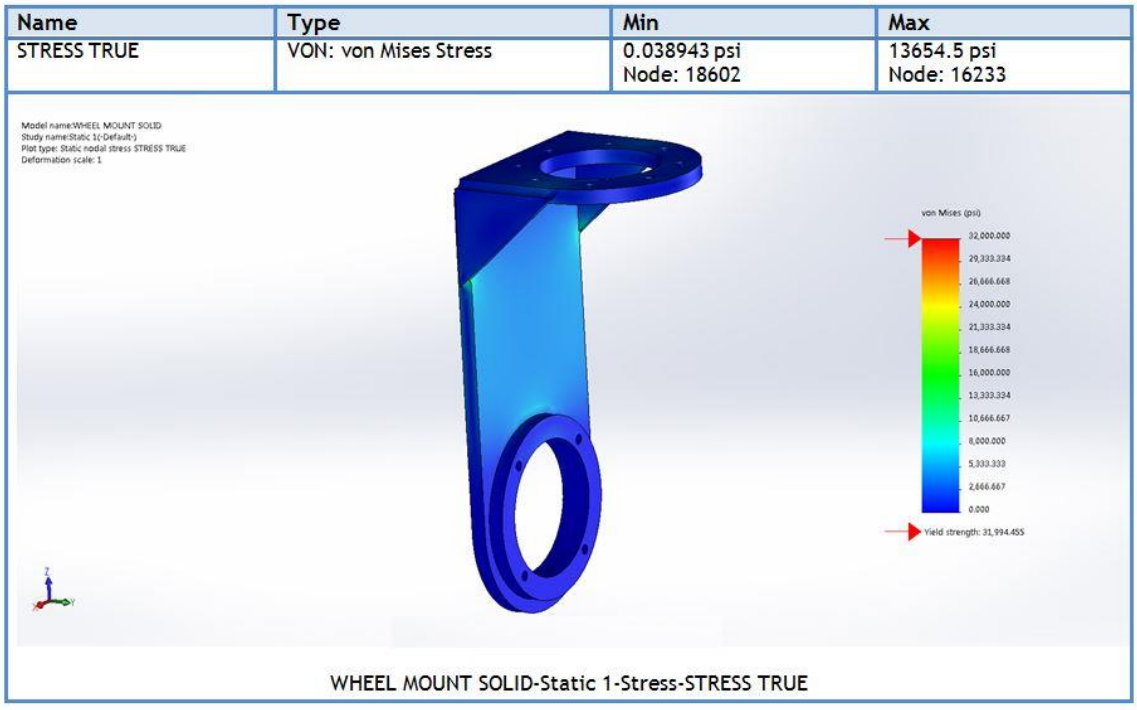



**Figure 4.10: Wheel mounts stress.**

| Name | Type | Min | Max |
|---|---|---|---|
| DISPLACEMENT TRUE | URES: Resultant Displacement | 0 mm<br>Node: 226 | 1.58593 mm<br>Node: 497 |

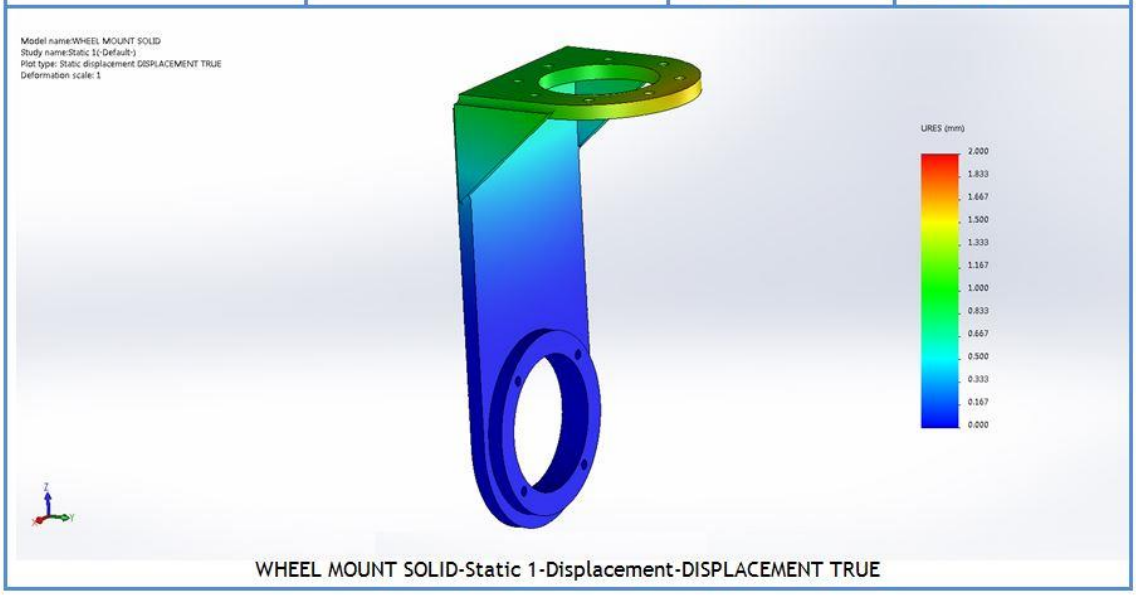



**Figure 4.11: Wheel mounts deflection in millimeters.**

## 4.7 Electrical and Electronics Component Enclosure

The electronics were encased in a metal box fabricated out of medium gage sheet metal with supports to mount components. This allowed electronic components to be located in the defined space of the electrical enclosure (Figure 4.12) for ease of access.

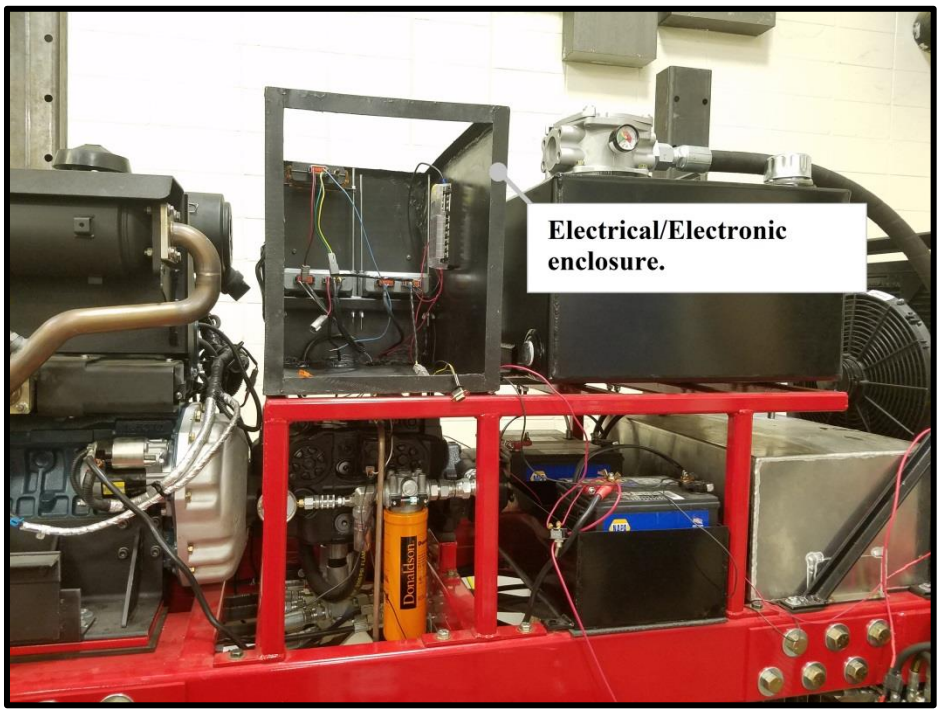



**Figure 4.12: Electrical/Electronic enclosure.**

The electronics enclosure also provided a mounting structure for the engine electric control box (Figure 4.13).

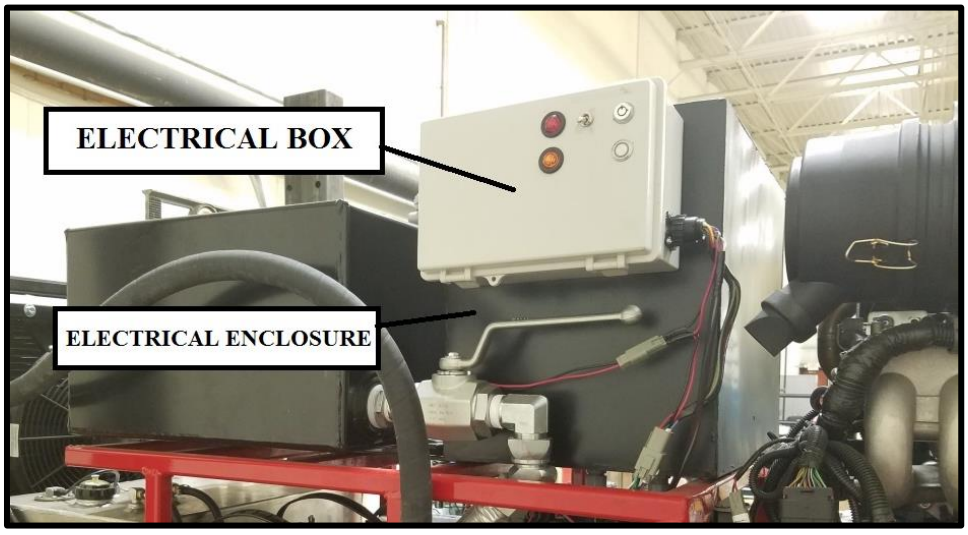



**Figure 4.13: Electrical control box mounted on the back of the enclosure.**

## 4.8 Final Frame

The final frame assembly (Figure 4.16) adequately supported the engine, electrical, electronic and hydraulic components of the Flex-Ro. The internal frame measured 2 foot in width and 6 foot in length. The wheels had adjustment capabilities to accommodate 30 and 36 inch row spacing. At the lowest height adjustment setting the bottom of the frame measured just less than 4 foot from ground level (Figure 4.14), and can be adjusted up to 8 foot (Figure 4.15).

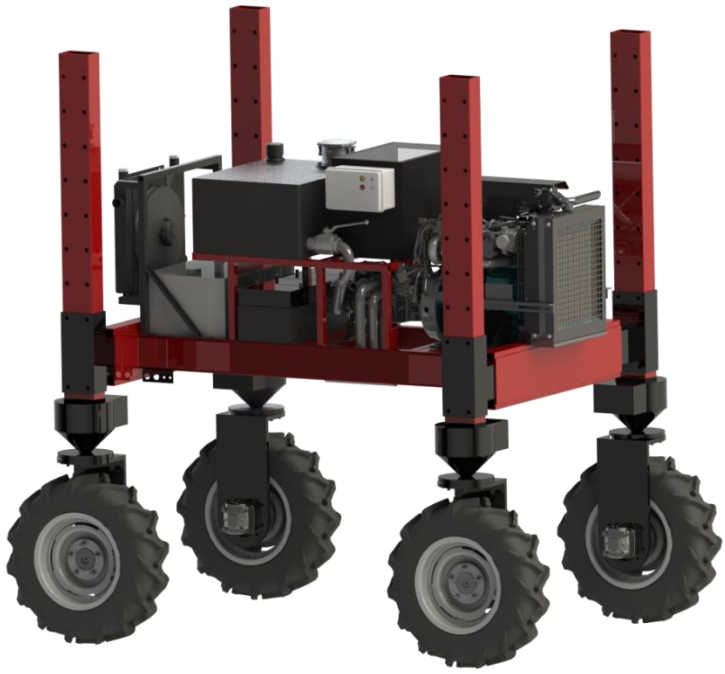


**Figure 4.14 Rendering of Flex-Ro in the shortest state, with the bottom of the frame being close to 4 ft. off the ground**

**Figure 4.15 Rendering of Flex-Ro in the tallest state, with the bottom of the frame being close to 8 ft. off the ground.**

**Figure 4.16: Flex-Ro's Frame with required components installed adjusted to the lowest height setting.**

# Chapter 5 Flex-Ro's Steering System and Embedded Steering Control Application Program Development

## 5.1 Steering Introduction

Flex-Ro required a steering mechanism in order to properly navigate in various steering configurations. While the size and weight allow many traditional steering methods to be considered, the vertical adjustment and the absence of an articulated frame requires Flex-Ro to implement an independent wheel steering system. Four wheel independent steering was chosen provide added steering modes for the machine. Furthermore, the use in Midwest row crops further constrains the footprint for which the steering mechanism must fit. If the steering mechanism was implemented at a height below the maximum crop height it must fit within the confines of the row spacing of the crops.

## 5.2 Steering Objectives

The specific objectives of this chapter were to:

(1) Develop a four wheel steering system to be implemented onto the Flex-Ro Robotic Platform

(2) Develop a CAN message control application for controlling the steering system

## 5.3 Methods and Materials

### *5.3.1 Steering Torque Calculations*

A preliminary calculation was performed based on a kingpin power steering design. This allowed for initial estimation of torque requirements for steering the robotic platform. It is important to note that in using this method to estimate torque requirements various

assumptions had to be made. First, the kingpin offset length was effectively zero, as the steering rotation on the robotic platform was designed to be directly in-line with the steered wheel. This allowed for a simple derivation of the equation below (Patil and Sonawane, 2016) leading to a required steering torque of 1782 in-lbs. per wheel.

$$T = W\mu\sqrt{\frac{B^2}{8}}$$

With

**T** = steering torque requirements in (in-lbs.).

**W** = Weight directly on steered wheel (lbs.).

**μ =** coefficient of friction.

**B** = Nominal width of the tire (in).

### *5.3.2 Gearbox*

Due to the large steering torque requirements, the need for a gearbox was apparent. While many different gearing options were available a worm gear drive system offered many advantages while providing the torque requirements. The first big advantage of this system is that it is a relatively simple design that is used in various industries. Second, the worm gear design avoids undesired feedback from outside factors to overcome the torque applied at the input, effectively developing a holding torque. A 50:1 gearbox that is implemented on center pivot drive wheels was chosen for steering each wheel of the Flex-Ro individually (Figure 5.1). The input drive for this gearbox was perpendicular to the output axis allowing for direct inline steering. Further, the gearbox was self-contained

allowing for simple implementation with bolt connections in line with the output drive axis.

**Figure 5.1: Worm gearbox used on the Flex-Ro robotic platform for steering of each wheel (Zimmatic, 2014).**

Using the 50:1 gear ratio of the gearbox the steering torque input required was adjusted for the new system. Assuming 90% gear efficiency, the adjustment led to an estimated torque requirement of about 40 in-lbs. or 3.3 ft.-lbs. The torque requirements were then tested with the configured frame and gearbox in alignment. These tests were performed with Flex-Ro, including all components, positioned on a smoothed concrete surface. A click torque wrench was set to the calculated torque values and used to actuate the inputs of the steering gearbox. The set value of the torque wrench was incrementally increased until the wheel could be turned without exceeding such value.

Table 5.1 shows the results of the torque test on each wheel, with a No representing that the wheel did not rotate prior to the torque wrench exceeding torque, and Yes showing that the wheel did rotate prior to reaching the torque wrench setting.

**Table 5.1: Torque values from test of the steering gearboxes.**

| Wheel | 1 ft-lb | 2 ft-lb | 3 ft-lb | 4 ft-lb | 5 ft-lb | 6 ft-lb | 7 ft-lb |
|---|---|---|---|---|---|---|---|
| Front left | No | No | No | No | No | YES | YES |
| Front right | No | No | No | No | No | No | YES |
| Rear left | No | No | No | No | No | No | YES |
| Rear right | No | No | No | No | No | YES | YES |

Given that all wheels rotated freely at a torque value of (7 ft.-lbs.) this was determined to be the required input torque for each steering system worm drive with a 0.75 coefficient of friction, rubber on concrete (Allen et al., 1997). To accommodate for other surface friction values the required input torque was divided by the estimated coefficient of concrete and rubber (0.75) to get an estimated torque requirement using a coefficient of friction equal to 1, and increased by 20 percent for good measure. The resulting torque of 11 ft.-lbs. was used for further development.

### *5.3.3 Electric Steering Motor sizing*

A 12V Direct Current (DC) motor was the preferred method of actuation for the steering gearboxes. This was mainly due to the platform electrical system operating on this voltage. Additionally, the DC motors have simple control architectures that can be implemented onto the Danfoss controllers in use by the Flex-Ro. The 11 ft-lb steering torque was used as the main constraint to the motor selected for driving the gearbox assembly. This value was taken as a continuous torque setting for the motor.

Additionally, the platform had to meet certain steering requirements. It was determined that the steering angle as defined in (International Organization for Standardization, 2008) would be 30 degrees in both directions for normal operating conditions (Figure 5.2). It is also important to note that this system can steer beyond the normal steering angles.

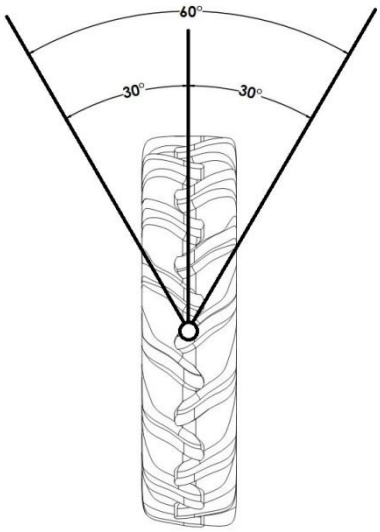



**Figure 5.2: Normal steering rotation for each wheel on Flex-Ro.**

The steering system should actuate from hard left to hard right (60 degrees total) in 2 seconds. This steering speed would allow for significant control abilities at low speeds. This also exceeds the ISO standard 10998 (International Organization for Standardization, 2008) requiring the steering speed to the largest steering angle, full left to full right or 60 degrees for Flex-Ro, to occur in 5 seconds for speeds less than 40 km/h. A 60 degree rotation in 2 seconds yields a rotational speed for the wheel of 5 rpm. Taking into account the gearbox ratio, the speed of the motor must be about 250 rpm. A DC electric motor (Model: D40-675A-12V GP81-007, Midwest Motion Products, Watertown MN) (Figure 5.3) was selected which can provide a speed of 284 rpm and continuous

torque value of 142 in-lbs. (11.83 ft.-lbs.) at 57 Amps current draw. Additionally, the motor can provide a maximum torque of 354 in-lbs. (29.5 ft.-lbs.) intermittently.


**Figure 5.3: 12V electric implemented on Flex-Ro for steering.**

### *5.3.4 Motor Drivers*

For steering motor control, a motor driver was needed to translate the Danfoss controller's commands into electrical signals. This motor driver had to provide a continuous current exceeding that of the motor (57 Amps), with the ability to be controlled via the Danfoss embedded controllers. The simplest control method was the use of a pulse width modulation (PWM) signal. The motor controller (Model: Victor SP, Vex Robotics, Greenville, Texas) (Figure 5.4) was selected with a continuous current rating of 60 Amps, a surge current rating of 100 Amps, and the ability to be controlled via a PWM signal. Each wheel steering motor was controlled by an individual motor driver.

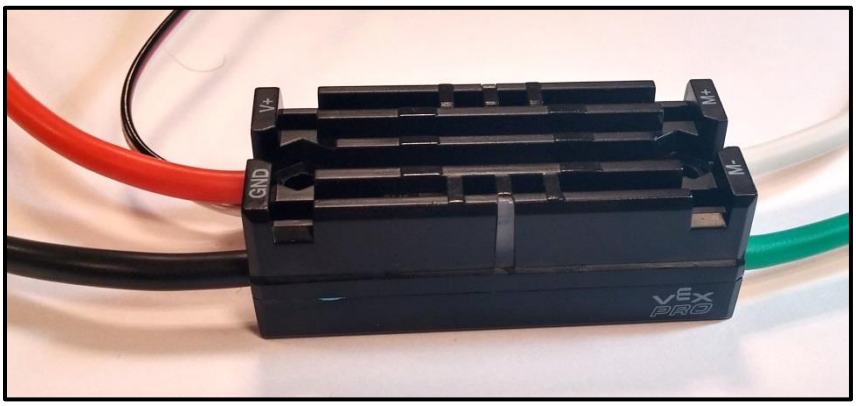



**Figure 5.4: Motor driver used to control the steering motors on Flex-Ro.**

The motor was connected to the gearbox using a shaft coupling (Figure 5.5). The output shaft of the motor was attached to the input shaft of the gearbox using two couplings fastened together. This configuration accommodated for the differences in the metric 19 mm keyed shaft of the motor and the 1 inch straight shaft of the gearbox. The gearbox coupling was modified to accommodate a bolt for fastening to the shaft, while the motor coupling utilizes a split collar that compresses the collar onto the keyed shaft.

**Figure 5.5: Shaft coupling connecting the output shaft of the steering motor to the input shaft of the steering gearbox.**

The coupling is housed in the steering motor mount (Figure 5.6). The steering motor mount was made from square tubing, with holes to mount to the motor, and the gearbox.

The top and bottom of this mount remain open, which allows the coupling to be properly secured to the gearbox and motor shafts.

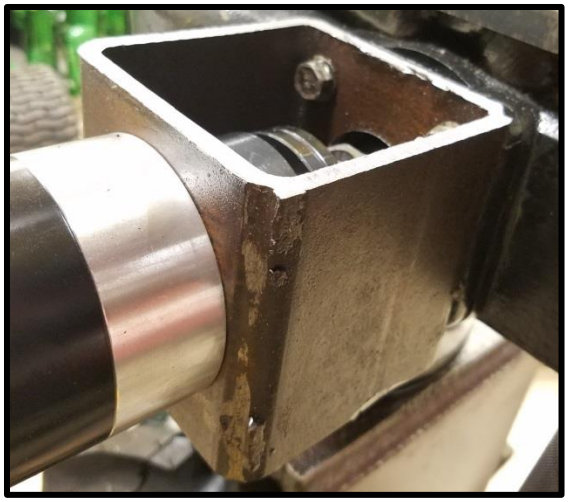


**Figure 5.6: Steering motor mount.**

### *5.3.5 Steering Application Program*

A proprietary CAN message was implemented for the steering control system (Figure 5.7). It was determined that the steering control would allow for the normal steering conditions and an additional 10 degrees in each direction of wheel rotation, resulting in 40 degrees left and right of center for steering. The message programmed to accommodate this value also provided the direction of the wheel. Additionally, it was established that the front wheels would always be steered in the same direction at any given time, while the same would hold true for the rear wheels. Therefore the message was created such that the byte in the D0 position of the steering CAN message controlled the direction of steering for the front wheels, with a 0 being in the positive direction (right) and a one being in the negative direction (left). The D3 byte position worked in the same way as the D0 byte position but controlled the direction of the rear wheels. To

gain resolution the steering control values are scaled such that 0 to 40 degrees was converted to 0 to 200 scale. This format was used for the D1, D2, D4, and D5 byte positions to control the left front, right front, left rear, and right rear wheels respectively.

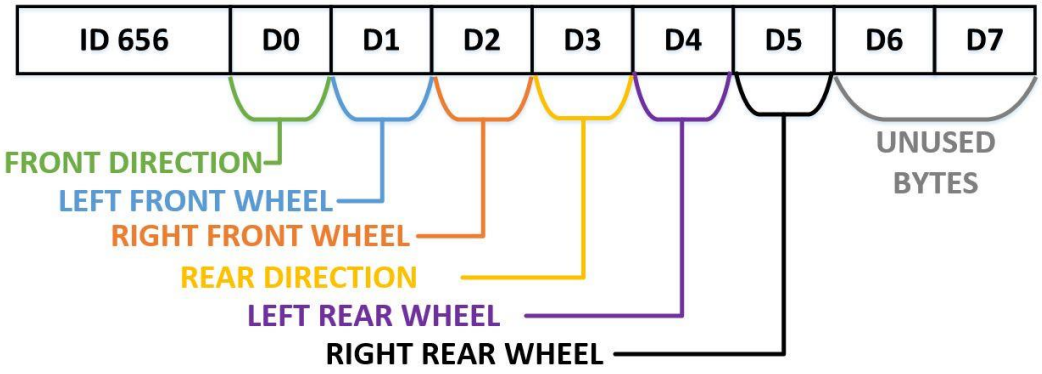



**Figure 5.7: Steering CAN message byte format.**

A Danfoss controller (Figure 5.8) was used to implement the steering control program with four outputs that were configured to generate PWM signals. One of the outputs was connected to each motor driver such that all four wheels were controlled independently.

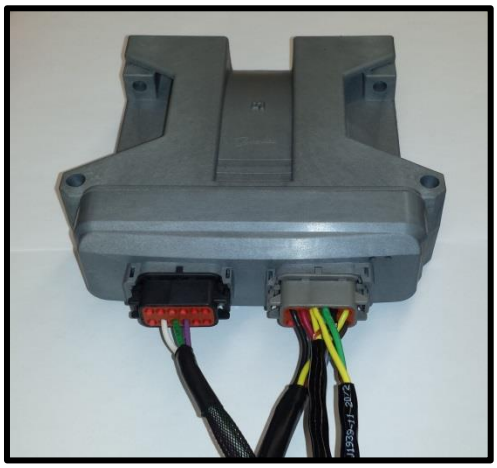


**Figure 5.8: Controller for steering.**

The CAN_RX block was used to receive the CAN message with the proper message ID for steering. This was routed into a separate STEERING CONTROL block, along with

the main input bus, for further use. The outputs from the block were connected to the main output bus to control the motor drivers (Figure 5.9).

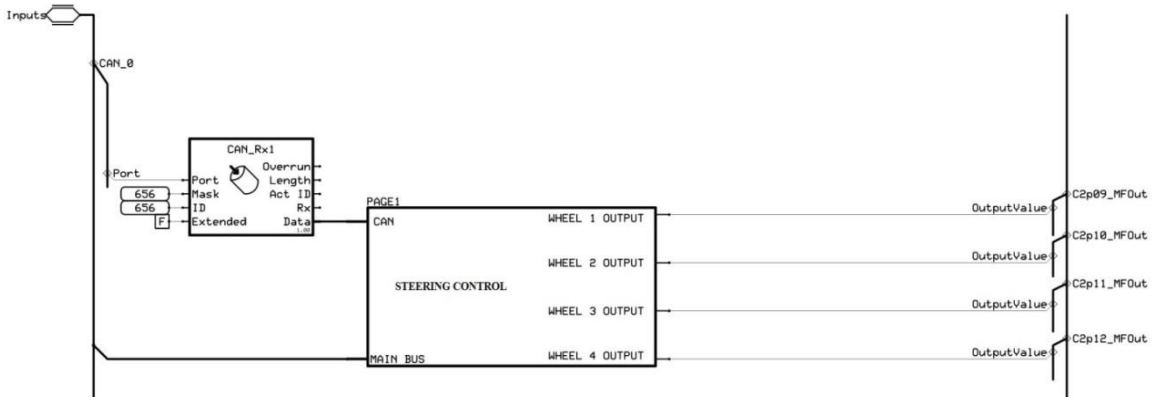



**Figure 5.9: Steering CAN receive and Steering control.**

The output values used to control the motor drivers connected to the steering motors were configured to meet the operational requirements of the motor drivers. The output values were configured to allow for a 0-100 percent duty cycle, adjustable in 0.1 percent increments, at a frequency of 100 Hz. The frequency value was assigned by setting the ReqFreq in the control settings to 100 (Figure 5.10).

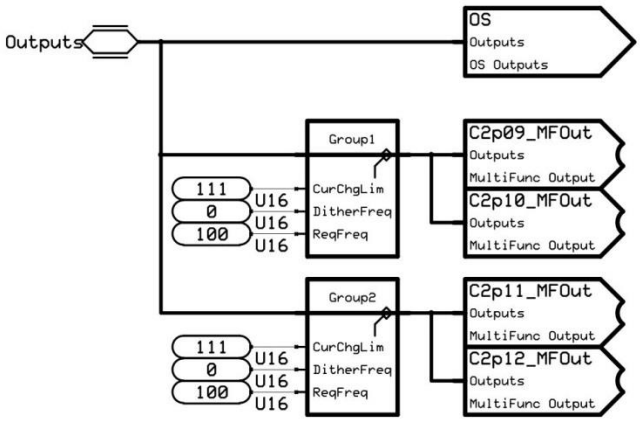



**Figure 5.10: Steering output configuration.**

Within the STEERING CONTROL block the analog signal feedback from the wheels

and the control values and directions were routed into a Steering PID control block. The outputs were then sent back to the main page (Figure 5.11).

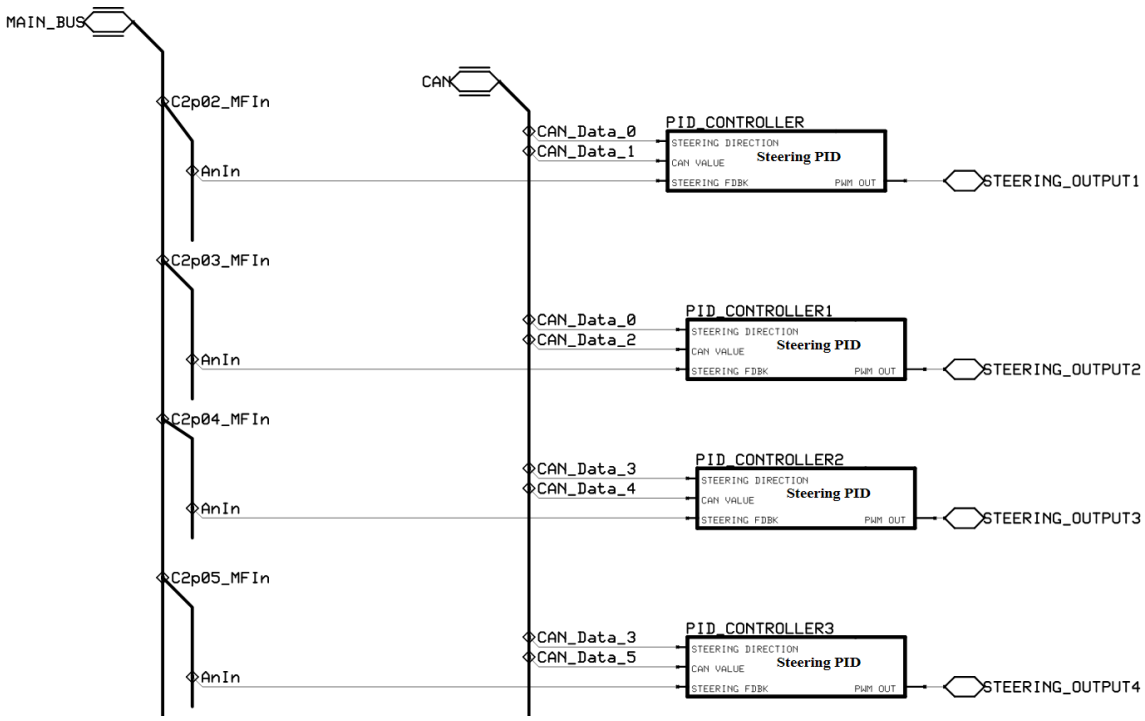



**Figure 5.11: CAN and Analog feedback to Steering PID controller.**

Within the Steering PID block there was some post-processing of the CAN and feedback information. The analog steering angle feedback was scaled from the analog value to values consistent (Figure 5.12) with the PI and PID nomenclature defined in the user manual (Danfoss, 2014) that allows for direction to be interpreted.


**Figure 5.12: Analog feedback scale to match PI input.**

The CAN bytes were first combined by retyping the value and signing it based on the direction. Once complete, the value was also scaled to be consistent with the PI controller inputs (Figure 5.13).

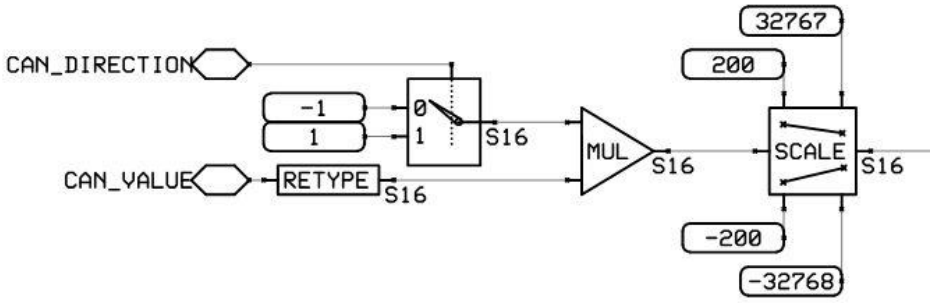



**Figure 5.13: CAN value configuration and scaling for PI controller.**

The PI control block (Figure 5.14) was utilized for controlling the steering system. This block was adjusted by setting the gains of the P and I, while adjusting various other components. The Stpt variable was the converted CAN control input data, while the Fdbk value was the converted measured analog feedback.

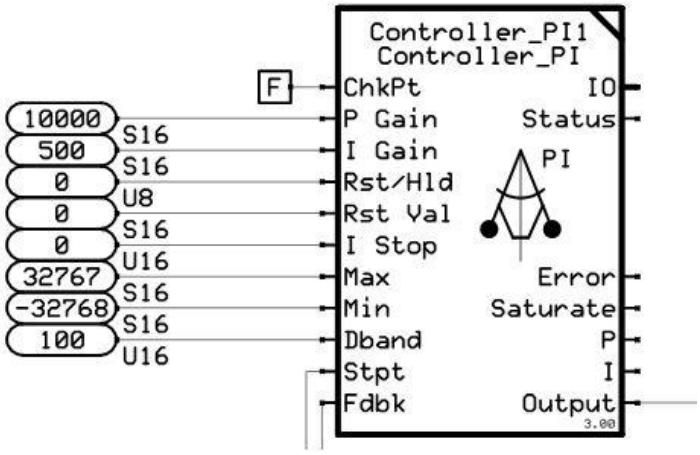



**Figure 5.14: PI control block.**

The outputs of the PI controllers were needed for controlling the steering motors. It first had to be scaled to a value with a minimum of 1 and a maximum of 10,000, and then retyped to an unsigned 16 bit integer (U16) number. It was then fed into a PWM driver block (Figure 5.15) which sent an output signal to control the motor.

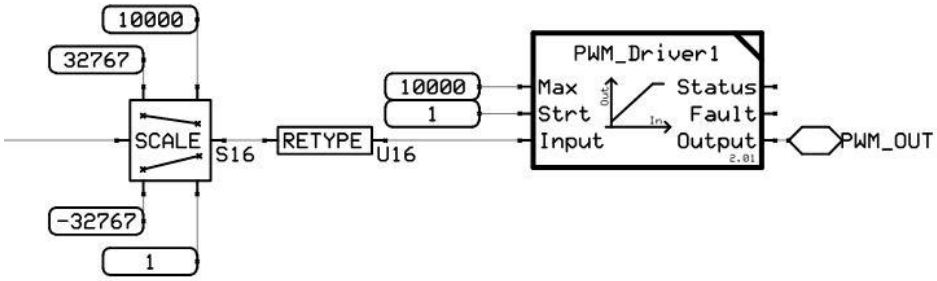



**Figure 5.15: PI output scaled and sent through PWM driver block.**

## 5.4 Steering Conclusions

The steering system including the gearboxes and electric steering motors were implemented on Flex-Ro. The design of the steering components allowed a significant

reduction in the torque required to steer. The application program was developed such that a CAN message sent to the steering control module controlled of the steering angle. However, the feedback system to send steering angle values to the controller is still under development and has not been completed. Once the feedback system is completed the steering system can be fully implemented, tuned, and evaluated. This closed loop configuration allows for significant control for steering, while also allowing machine steering state to be monitored.

# Chapter 6 Remote Control Application Program Development for Tele-Operated Machine Control of Flex-Ro

## 6.1 Remote Introduction

With a robotic platform there will be need for both autonomous operation and tele-operation. Specifically, operating in areas where the robotic platform must be manually controlled requires tele-operation. Instances where tele-operation is required are; loading and unloading of Flex-Ro to and from a transport vehicle, and moving it in and out of a storage shed.

## 6.2 Remote Objectives

The specific objectives of this chapter were to:

(1) Create a wireless user friendly remote control for operating the robotic platform using the existing control architecture on the machine.
(2) Develop and evaluate an embedded controller application that can generate CAN messages for operating the Flex-Ro in manual mode.

## 6.3 Remote Control Application Program

Based on the control architecture of the existing robotic platform it was decided that a Danfoss embedded controller should be used for creating the remote. The controller allowed for the configuration of multiple digital inputs to control various parts of the machine operation.

### *6.3.1 Engine Control*

The first process to operate via the remote controller is the engine start up and operation. A program block shown in Figure 6.1 titled Engine Control was created to convert the physical inputs into a CAN message to be sent over the bus.

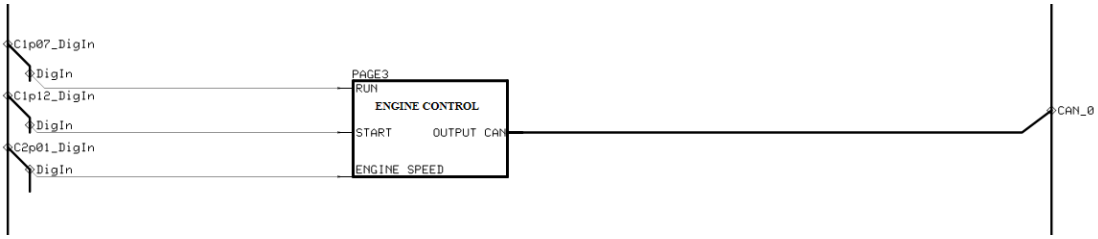



**Figure 6.1: Engine Control block.**

Three digital inputs are utilized for this operation and connected to physical switches built into the remote. The first two are for engine start up. A key switch is wired into C1P07 connector port of the controller for sending the run command to the engine. The physical input is read into the block as a Boolean value and a switch is used to convert that value to an unsigned 8 bit integer (U8) to be sent in the CAN message at the 0 byte position (Figure 6.2).

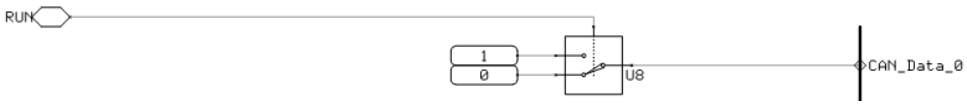



**Figure 6.2: Run command from digital input to CAN message.**

A momentary switch wired into the C1P12 connector port of the controller is used to toggle the start command of the engine. The same control method used for the run

Command is used to insert the start command byte into the 1 byte position of the engine control CAN message (Figure 6.3).

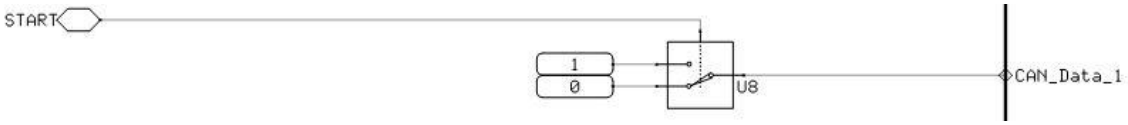



**Figure 6.3: Start command from digital input to CAN message**

The last input for the engine block is to control engine speed of the robotic platform. As previously discussed the engine speed can be fully controlled in 1 rpm increments from 850 to 3400 rpm. However, it was determined that during manual operation the speed of the machine should be limited to allow for slow maneuvers. Therefore the tele-operation control program was developed such that the remote would allow for two engine speed labeled as low and high. Another toggle switch was wired into the C2P01 connector to accomplish this. In the engine control message the value in the D4 byte position controls whether the TSC1 message is being transmitted while values in the D2 and D3 byte positions determine the speed of the engine. It was decided that the low engine speed would be idle (850 rpm) and as such could be accomplished by setting the value in the D4 byte position to zero. This also allowed the high engine speed, determined to be 1200 rpm, to be programmed as constant bytes values of 176 and 4 put into the D2 and D3 byte positions, respectively. This speed is only active when the value in the D4 byte position is set to 1 using the same logic as the start and run commands. The message is then sent with the proper identifier over the CAN bus (Figure 6.4).

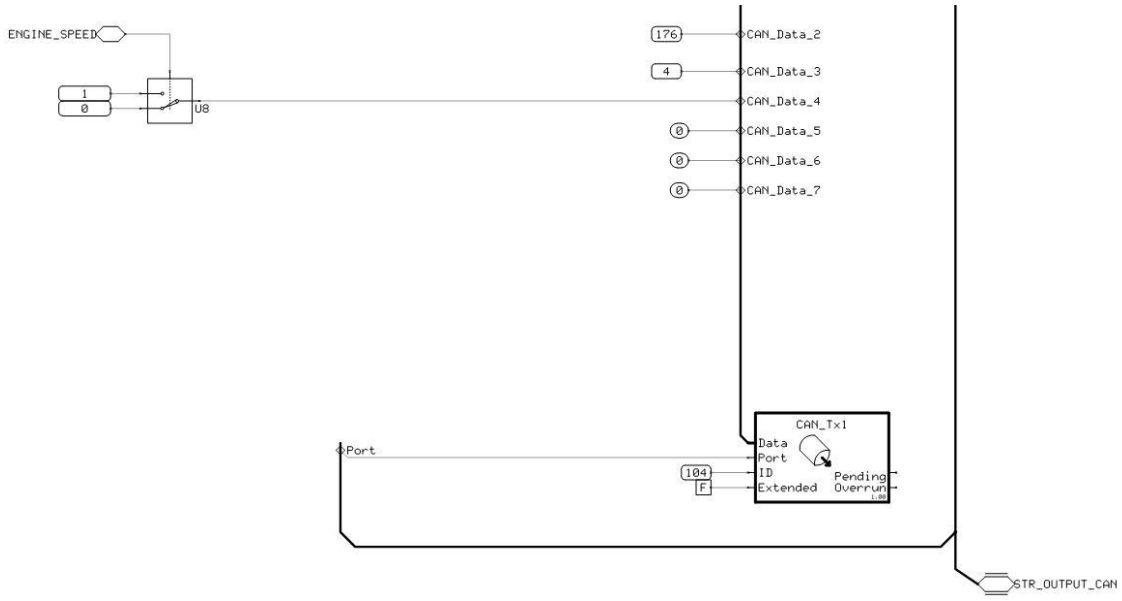



**Figure 6.4: Engine speed mode configuration for CAN TX message.**

### *6.3.2 Hydraulic Control*

The control for the hydraulic pump must be such that it allowed for forward and reverse drive motor control while also providing the full range of pump displacement. A joystick (JS1000, Danfoss IA) that communicates via CAN message was selected for this input (Figure 6.5).


**Figure 6.5: Danfoss JS1000 CAN enabled joystick.**

Danfoss provides a JS1000_CAN1 compliance block that was used to interface the joystick with the outputs being sent to the hydraulic control program block. The source address of the joystick used was factory set to 33 (Figure 6.6).

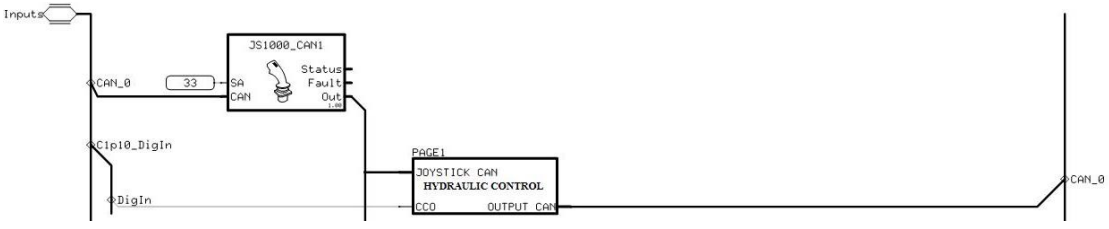



**Figure 6.6: Joystick and digital input for hydraulic control block.**

A physical switch was used to control the CCO portion of the message that allows for hydraulic brake release and enables the EDC on the pump (Figure 6.7). This converted the Boolean value to a U8 data structure, using the same method as the engine command, and output the byte in the byte 7 position of the message.


**Figure 6.7: Digital input for CCO control for CAN message.**

The Y axis position of the joystick (forward and backwards) is output from the joystick as a -10,000 to 10,000 value, which is the same range of values as the input for the pump control. This value was converted using a quotient and modulo to send the values over the D2 and D3 byte positions. A simple comparator was used to determine whether the signal was positive or negative and used in the D1 byte position to control the direction of the pump (Figure 6.8).

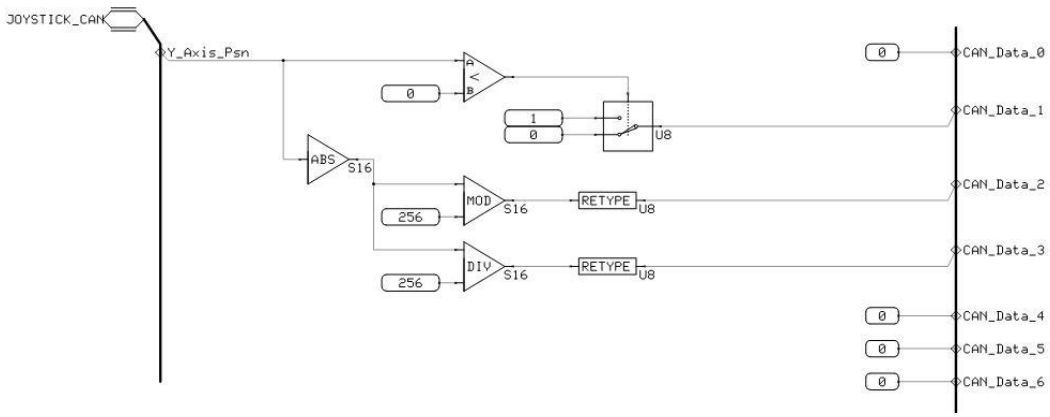



**Figure 6.8: Joystick input for controlling direction and speed in hydraulic CAN message.**

With all other bytes being set to zero again a CAN transmit block with the message identifier 333 was used to send the message onto the CAN bus (Figure 6.9).

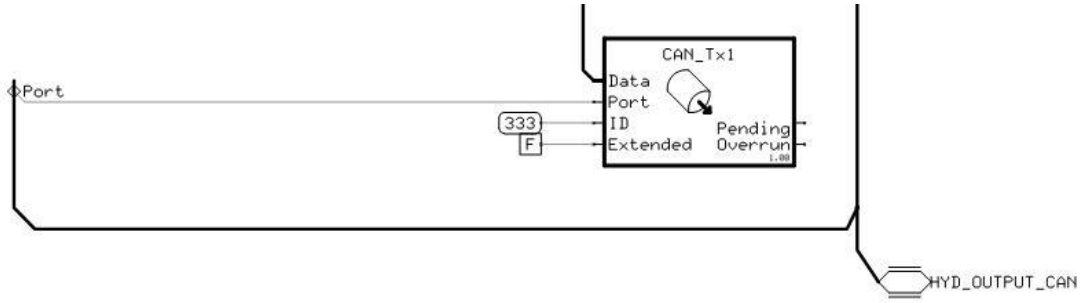



**Figure 6.9: CAN transmission block for hydraulic CAN message.**

### *6.3.3 Steering Control*

The last function of the remote control is to control the steering of the robotic platform. The same joystick is used to control steering with the outputs of the joystick compliance block also being inserted into the steering control block (Figure 6.10). Another toggle switch was used to control the two steering modes available on the remote.

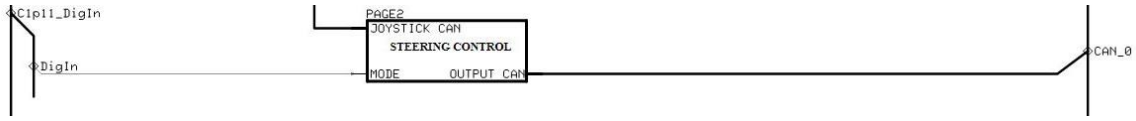



**Figure 6.10: Digital input and joystick to Steering Control Block.**

The X axis position of the joystick (left and right) was used for steering control. This again gives an output value of -10,000 to 10,000. A maximum steering angle of 40 degrees left and right was determined to be sufficient for manual operation of the robot. Therefore the value read from the joystick was scaled to a value of 0 to 80 in a U8 data format. The direction of rotation is also determined by the sign of the joystick input through the use of a comparator and switch. Furthermore, the mode is set for front wheel

only steering as a default, or four wheel steering when the steering mode switch is toggled. This is done by inserting the same steering angle value to the second two wheels, but in the opposite direction (Figure 6.11).

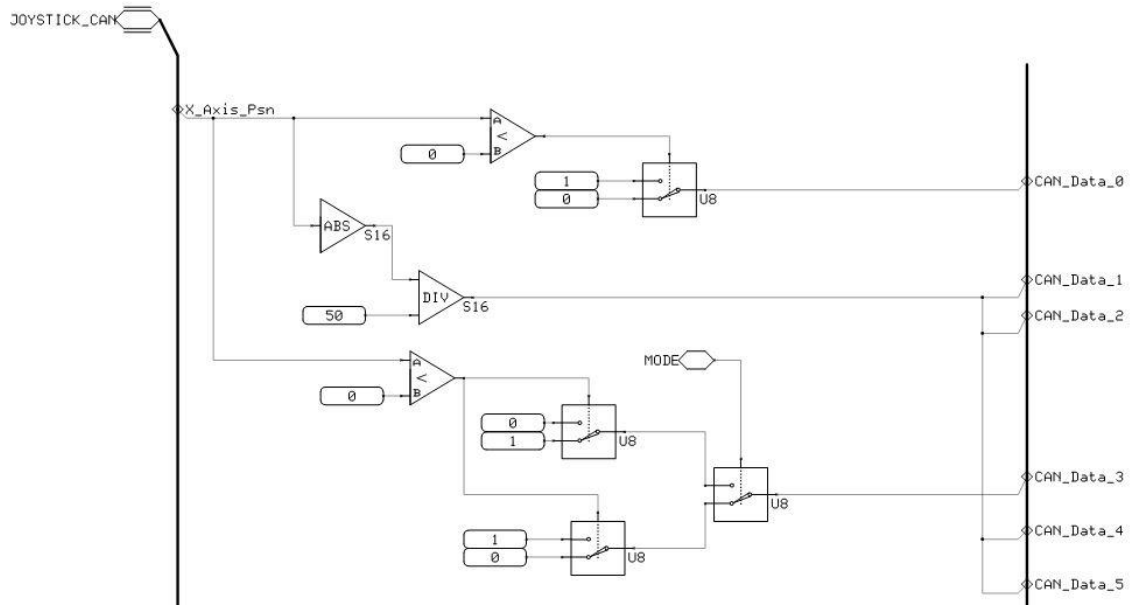



**Figure 6.11: Steering mode, direction, degree configuration for CAN message.**

Once the information is compiled it is again sent onto the bus with the message identifier 656 (Figure 6.12).

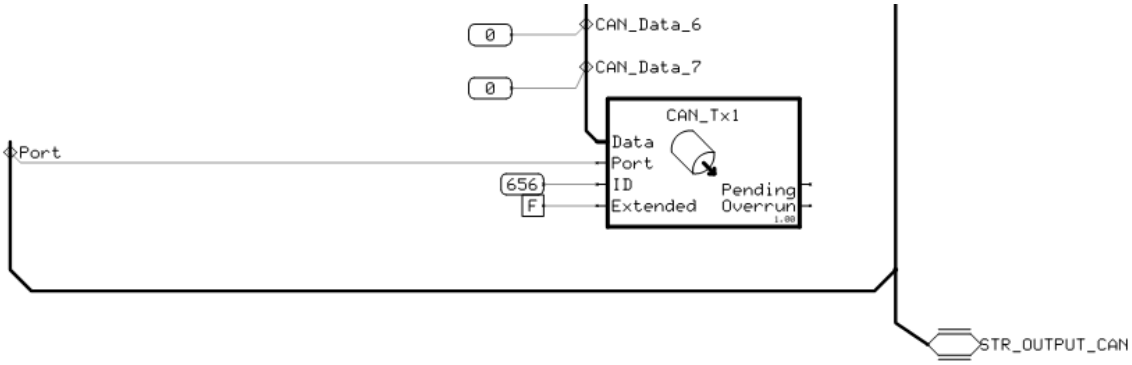



**Figure 6.12: Proper configuration of CAN TX block for steering CAN message.**

### *6.3.4 Wireless connection*

The above program worked when physically connecting the CAN bus to the machine. It was important however, to provide a wireless connection that allowed the operator to be a safe distance away from the machine during operation. The only way of achieving this was to create a wireless bridge for the CAN bus for all messages to be sent and received. A pair of Wireless CAN Modules (Model: WIC-2402, Magnetek, Menomonee Fall WI) (Figure 6.13) were implemented to accomplish this.


**Figure 6.13: Magnetek wireless CAN Module**

This wireless module provided advantageous features for a CAN control system. First, it allowed for two way communication of CAN messages, effectively creating a wireless bridge that acted as an extension of the CAN bus. This feature was imperative for tele-

operating the robotic platform through the use of CAN based control messages. This was done using two communication modules, one physically connected to the CAN bus on the Flex-Ro, and the other connected to the remote. This sent all messages across both CAN busses as if they were one physically connected bus. Further, this system implemented an emergency failsafe for situations where a connection is lost between the paired modules. It employed a digital output that was active only when the modules were connected, which was used as a digital input to stop some of the robotic platforms functionality in the event of communication loss. The range of these modules was about 1500 feet beyond which the communication will be lost.

# Chapter 7 Conclusions and Future Work

Overall, the design and control of Flex-Ro's subassemblies was successful. Flex-Ro's control architecture implemented a CAN bus for various distributed control and data acquisition. Furthermore, each subassembly that required control were performed using a control architecture consisting of a CAN control message being sent to a designated controller, that in turn governed various outputs for operation. The engine and its components were successfully installed and provided satisfactory control for the robotic platform. Additionally, the hydrostatic transmission operated effectively given the complexity of the system. All components within the hydrostatic transmission were properly sized and configured for use on Flex-Ro. The frame was able to support the machine, and be configured for various widths and heights. The steering was fully designed and programmed but still requires a feedback system to be installed and tested. The remote has been fully developed but still needs to be assembled and tested in conjunction with the machine. The wireless module was tested and works appropriately to transfer messages between the robotic platform and a computer, thus will work the same with the remote. The implementation of a full CAN bus that allows for control of all drive components permits for the autonomous navigation software that will be developed to seamlessly integrate into Flex-Ro.

Flex-Ro will serve as a medium for further research and development. This will eventually lead to a fully autonomous navigation protocol to propel the machine, while performing various agricultural operations.

Before further research can be performed the robotic platform needs to be completed for full operation. All the procedures have been prepared for an easy transition into teleoperation then full autonomous operation. The steering system needs to be implemented with a feedback mechanism and have the PI controller tuned for optimal operation. The remote needs to be assembled and bench tested to properly verify control messages prior to wireless implementation. Additionally, safety features need to be the next focus of the machine. A safety protocol has to be implemented as discussed; in conjunction with the wireless CAN module, to provide physical emergency switches on the robotic platform to safely stop the machine.

The next step in development of Flex-Ro once fully functional will be the implementation of high level software for navigation. This will be significantly simplified through the CAN message control architecture that has been developed for this thesis. The navigation software will need to integrate with a GPS unit and process position and machine control parameters, but will simply need to publish the correct control messages onto the bus to control the machine.

Last, once the navigation is implemented the platform will be used to test various machine vision technologies in field conditions. Machine vision will allow for improvements in safety. It will also allow for the development of technologies that will further the domains of crop scouting, phenotyping, and various other low draft applications.

**REFERENCES**

Allen, R.W., Rosenthal, T.J., Chrstos, J.P., 1997. A Vehicle Dynamics Tire Model for Both Pavement and Off-Road Conditions. SAE International. doi:10.4271/970559

Bakker, T., Asselt van, K., Bontsema, J., Müller, J., Straten van, G., 2010. Systematic design of an autonomous platform for robotic weeding. J. Terramechanics 47, 63–73. doi:10.1016/j.jterra.2009.06.002

Bak, T., Jakobsen, H., 2004. Agricultural Robotic Platform with Four Wheel Steering for Weed Detection. Biosyst. Eng. 87, 125–136. doi:10.1016/j.biosystemseng.2003.10.009

Bangert, W., Kielhorn, A., Rahe, F., Albert, A., Biber, P., Grzonka, S., Haug, S., Michaels, A., Mentrup, D., Hänsel, M., others, 2013. Field-Robot-Based Agriculture:"RemoteFarming. 1" and "BoniRob-Apps," in: 71th Conference LAND. TECHNIK-AgEng 2013. pp. 439–446.

Bawden, O., Ball, D., Kulk, J., Perez, T., Russell, R., 2014. A lightweight, modular robotic vehicle for the sustainable intensification of agriculture, in: School of Electrical Engineering & Computer Science; Science & Engineering Faculty. Presented at the Australian Conference on Robotics and Automation (ACRA 2014), Australian Robotics & Automation Association ARAA, University of Melbourne, Melbourne, VIC.

Bergerman, M., Singh, S., Hamner, B., 2012. Results with autonomous vehicles operating in specialty crops, in: 2012 IEEE International Conference on Robotics and Automation (ICRA). Presented at the 2012 IEEE International Conference on

Robotics and Automation (ICRA), pp. 1829–1835. doi:10.1109/ICRA.2012.6225150

Bulanon, D.M., Kataoka, T., Okamoto, H., Hata, S., 2004. Development of a real-time machine vision system for the apple harvesting robot, in: SICE 2004 Annual Conference. Presented at the SICE 2004 Annual Conference, pp. 595–598 vol. 1.

Chen, B., Tojo, S., Watanabe, K., 2003. Machine Vision for a Micro Weeding Robot in a Paddy Field. Biosyst. Eng. 85, 393–404. doi:10.1016/S1537-5110(03)00078-3

Cherry, 2016. GS1005 – GS1007 Sensors Datasheet.

Cundiff, J.S., 2001. Fluid power circuits and controls: fundamentals and applications. CRC Press.

Danfoss, 2015. Technical Information H1 Axial Piston Tandem Pumps Size 045/053.

Danfoss, 2014. PLUS+1 Function Block Library—Control Function Blocks.

Danfoss, 2012. J1939 Function Block Library User Manual.

Danfoss, 2008. PLUS+1$^{TM}$ MC024-010 andMC024-012Controllers.

Danfoss, 2005. Basic Function Blocks Library User's Manual.

Dieter Kutzbach, H., 2000. Trends in Power and Machinery. J. Agric. Eng. Res. 76, 237–247. doi:10.1006/jaer.2000.0574

EControls Inc., 2008. ECOM Software Driver Installation.

G. Muscato, D. Caltabiano, S. Guccione, D. Longo, M. Coltelli, A. Cristaldi, E. Pecora, V. Sacco, P. Sim, G.S. Virk, P. Briole, A. Semerano, T. White, 2003. ROBOVOLC: a robot for volcano exploration result of first test campaign. Ind. Robot Int. J. 30, 231–242. doi:10.1108/01439910310473942

Godoy, E.P., Tangerino, G.T., Tabile, R.A., Inamasu, R.Y., Porto, A.J.V., 2012a. Networked Control System for the Guidance of a Four-wheel Steering Agricultural Robotic Platform. J Control Sci Eng 2012, 4:4–4:4. doi:10.1155/2012/368503

Goering, C.E., 1992. Engine and tractor power. American Society of Agricultural Engineers (ASAE).

Gonzalez-de-Soto, M., Emmi, L., Benavides, C., Garcia, I., Gonzalez-de-Santos, P., 2016. Reducing air pollution with hybrid-powered robotic tractors for precision agriculture. Biosyst. Eng. 143, 79–94. doi:10.1016/j.biosystemseng.2016.01.008

Hansen, K.D., Garcia-Ruiz, F., Kazmi, W., Bisgaard, M., la Cour-Harbo, A., Rasmussen, J., Andersen, H.J., 2013. 8th IFAC Symposium on Intelligent Autonomous VehiclesAn Autonomous Robotic System for Mapping Weeds in Fields. IFAC Proc. Vol. 46, 217–224. doi:10.3182/20130626-3-AU-2035.00055

International Organization for Standardization, 2008. ISO 10998:2008 - Agricultural tractors -- Requirements for steering.

Jinlin, X., Liming, X., 2010. Autonomous Agricultural Robot and its Row Guidance, in: 2010 International Conference on Measuring Technology and Mechatronics Automation (ICMTMA). Presented at the 2010 International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), pp. 725–729. doi:10.1109/ICMTMA.2010.251

Kubota, 2014. Kubota Engines Application Manual (WG1605-E3, WG2503-E3 Model) Third edition.

Kubota, 2012. WORKSHOP MANUAL GASOLINE,LPG, NATURAL GAS ENGINE WG1605-E3.

Marx, S.E., Luck, J.D., Hoy, R.M., Pitla, S.K., Blankenship, E.E., Darr, M.J., 2015. Validation of machine CAN bus J1939 fuel rate accuracy using Nebraska Tractor Test Laboratory fuel rate data. Comput. Electron. Agric. 118, 179–185. doi:10.1016/j.compag.2015.08.032

Oksanen, T., 2015. Accuracy and Performance Experiences of Four Wheel Steered Autonomous Agricultural Tractor in Sowing Operation, in: Mejias, L., Corke, P., Roberts, J. (Eds.), Field and Service Robotics, Springer Tracts in Advanced Robotics. Springer International Publishing, pp. 425–438.

Patil, D.S., Sonawane, P., 2016. Design Analysis of Mounting Bracket for Steering Gearbox.

Pitla, S.K., Lin, N., Shearer, S.A., Luck, J.D., 2014. Use of Controller Area Network (can) Data to Determine Field Efficiencies of Agricultural Machinery. Appl. Eng. Agric. 30, 829–839.

Pitla, S.K., Luck, J.D., Werner, J., Lin, N., Shearer, S.A., 2016. In-field fuel use and load states of agricultural field machinery. Comput. Electron. Agric. 121, 290–300. doi:10.1016/j.compag.2015.12.023

SAE, 1994. 71,(R) Vehicle Application Layer-J1939-71. USA: SAE J1939 Committee Draft, 1994: 5-98.

Slaughter, D.C., Giles, D.K., Downey, D., 2008. Autonomous robotic weed control systems: A review. Comput. Electron. Agric., Emerging Technologies For Real-

time and Integrated Agriculture Decisions 61, 63–78. doi:10.1016/j.compag.2007.05.008

Stone, M.L., Benneweis, R.K., Van Bergeijk, J., 2008. Evolution of electronics for mobile agricultural equipment. Trans. ASABE 51, 385–390.

Tabile, R.A., Godoy, E.P., Pereira, R.R.D., Tangerino, G.T., Porto, A.J.V., Inamasu, R.Y., 2011. Design and development of the architecture of an agricultural mobile robot. Eng. Agríc. 31, 130–142. doi:10.1590/S0100-69162011000100013

Tanigaki, K., Fujiura, T., Akase, A., Imagawa, J., 2008. Cherry-harvesting robot. Comput. Electron. Agric., Special issue on bio-robotics 63, 65–72. doi:10.1016/j.compag.2008.01.018

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M.N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T.M., Kolski, S., Kelly, A., Likhachev, M., McNaughton, M., Miller, N., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Salesky, B., Seo, Y.-W., Singh, S., Snider, J., Stentz, A., Whittaker, W. "Red," Wolkowicki, Z., Ziglar, J., Bae, H., Brown, T., Demitrish, D., Litkouhi, B., Nickolaou, J., Sadekar, V., Zhang, W., Struble, J., Taylor, M., Darms, M., Ferguson, D., 2008. Autonomous driving in urban environments: Boss and the Urban Challenge. J. Field Robot. 25, 425–466. doi:10.1002/rob.20255

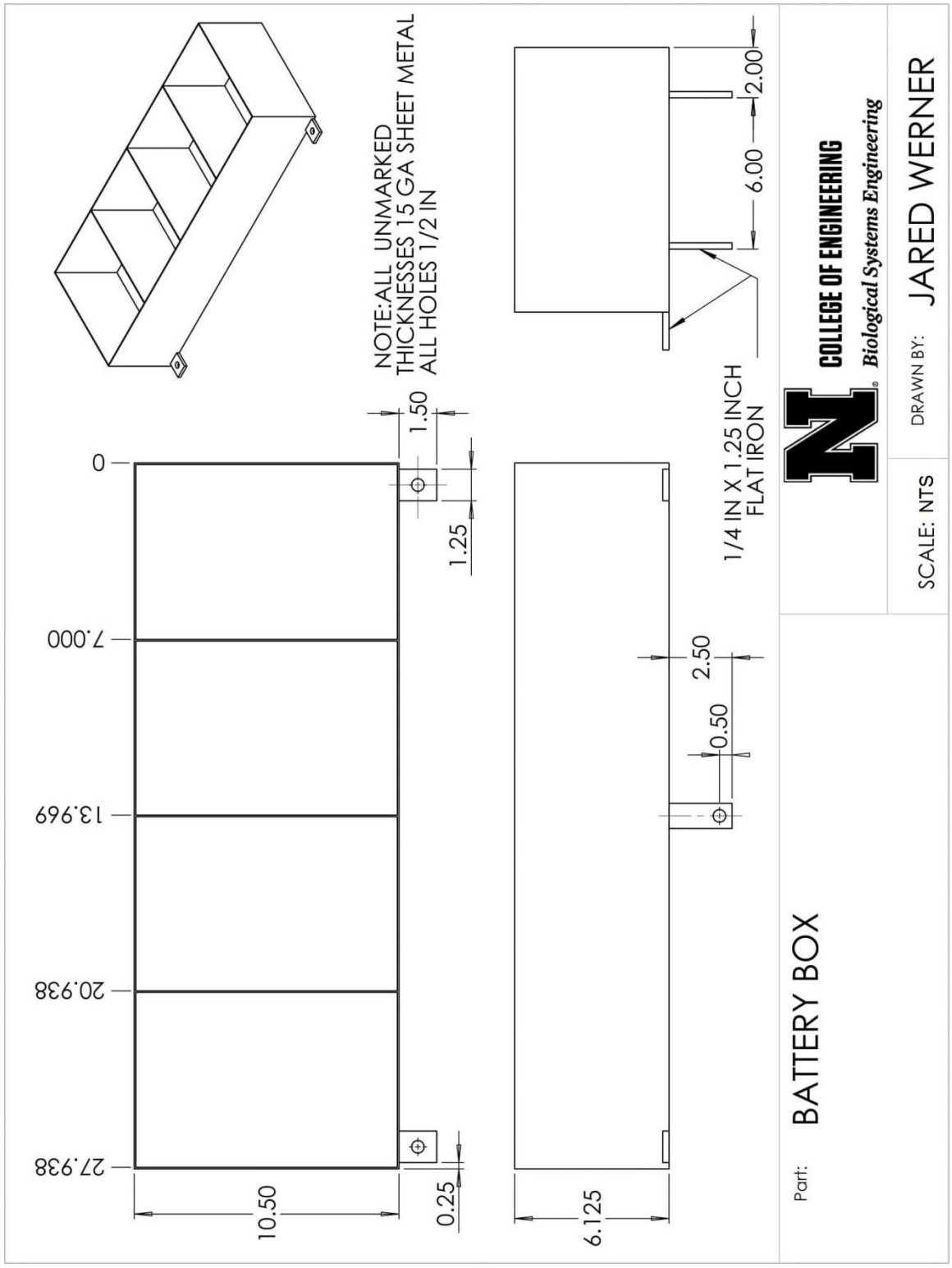
Voss, W., 2008. A comprehensible guide to J1939. Copperhill Technologies Corporation.

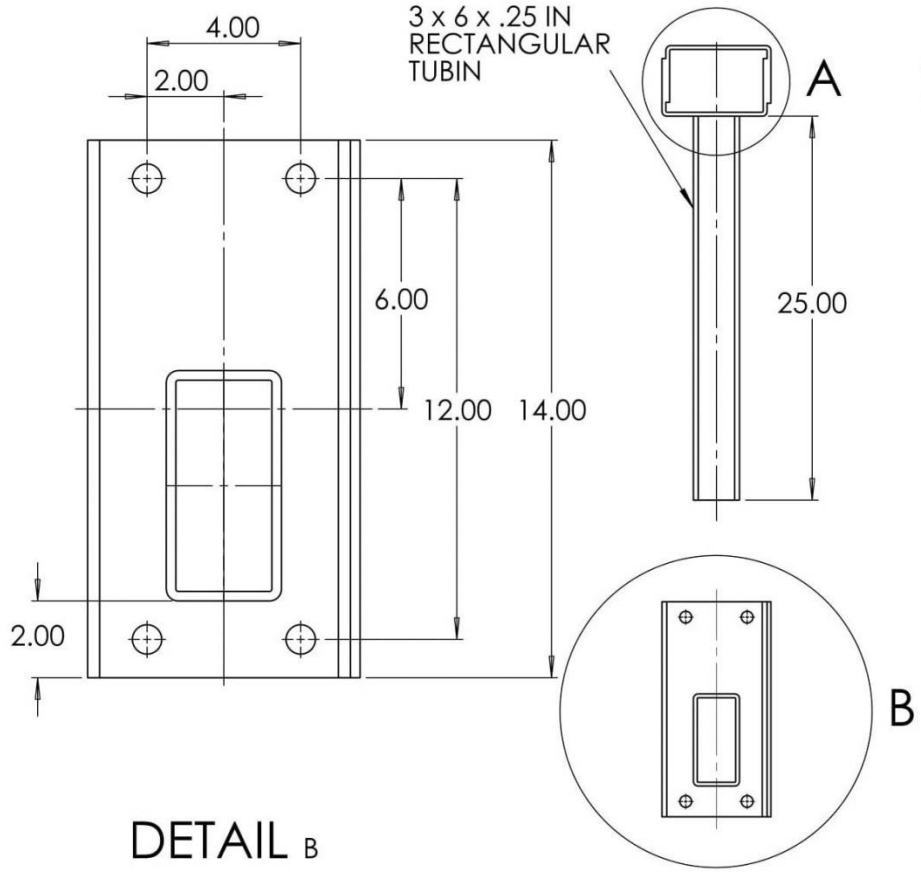
Voss, W., 2005. A comprehensible guide to controller area network. Copperhill Media.

Zhang, Y., Ye, Y., Wang, Z., Taylor, M.E., Hollinger, G.A., Zhang, Q., 2016. Intelligent In-Orchard Bin-Managing System for Tree Fruit Production.

Zimmatic, 2014. LINDSAY DRIVELINE BROCHURE.

# APPENDICES

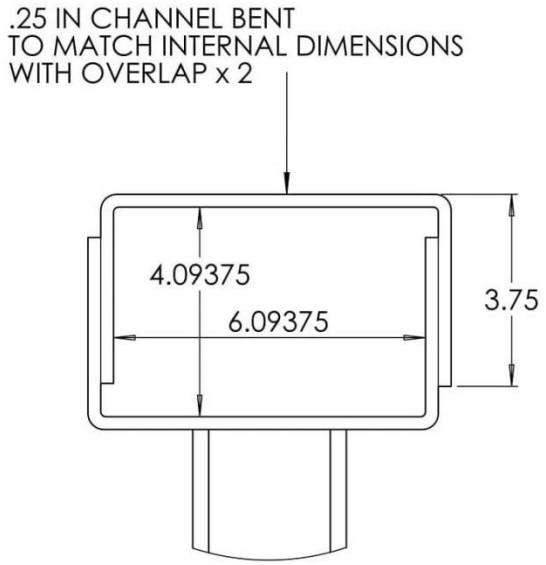

NOTE:ALL UNMARKED
THICKNESSES 15 GA SHEET METAL
ALL HOLES 1/2 IN
1/4 IN X 1.25 INCH
FLAT IRON
1.50
1.25
2.00
6.00
2.50
0.50
0
7.000
13.969
20.938
27.938
10.50
0.25
6.125
COLLEGE OF ENGINEERING
Biological Systems Engineering
DRAWN BY: JARED WERNER
SCALE: NTS
Part: BATTERY BOX




DETAIL B

SCALE 1 : 4




DETAIL A

SCALE 1 : 3

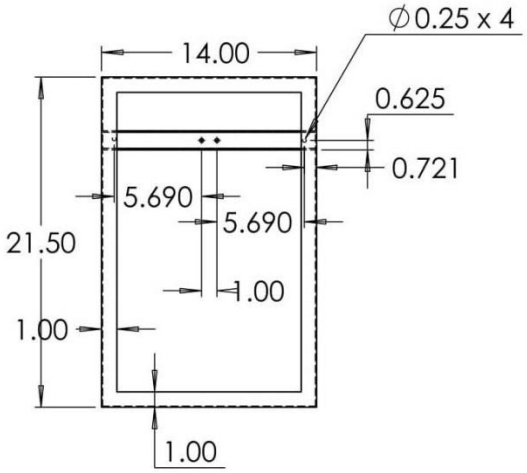
Part: SLIDE COLLAR

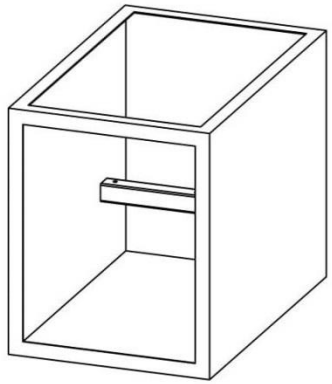
COLLEGE OF ENGINEERING

*Biological Systems Engineering*

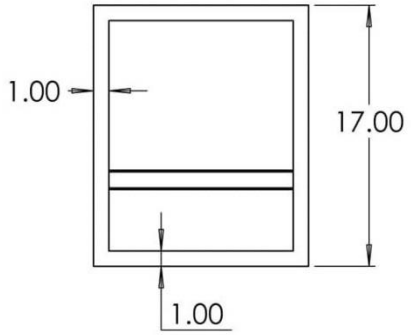
SCALE: NTS

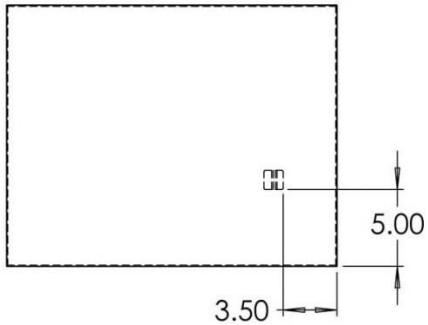
DRAWN BY: JARED WERNER



NOTE:
ALL 15 GA SHEET METAL
SUPPORT BAR 1.25 IN x 3/16 IN
SQUARE TUBING



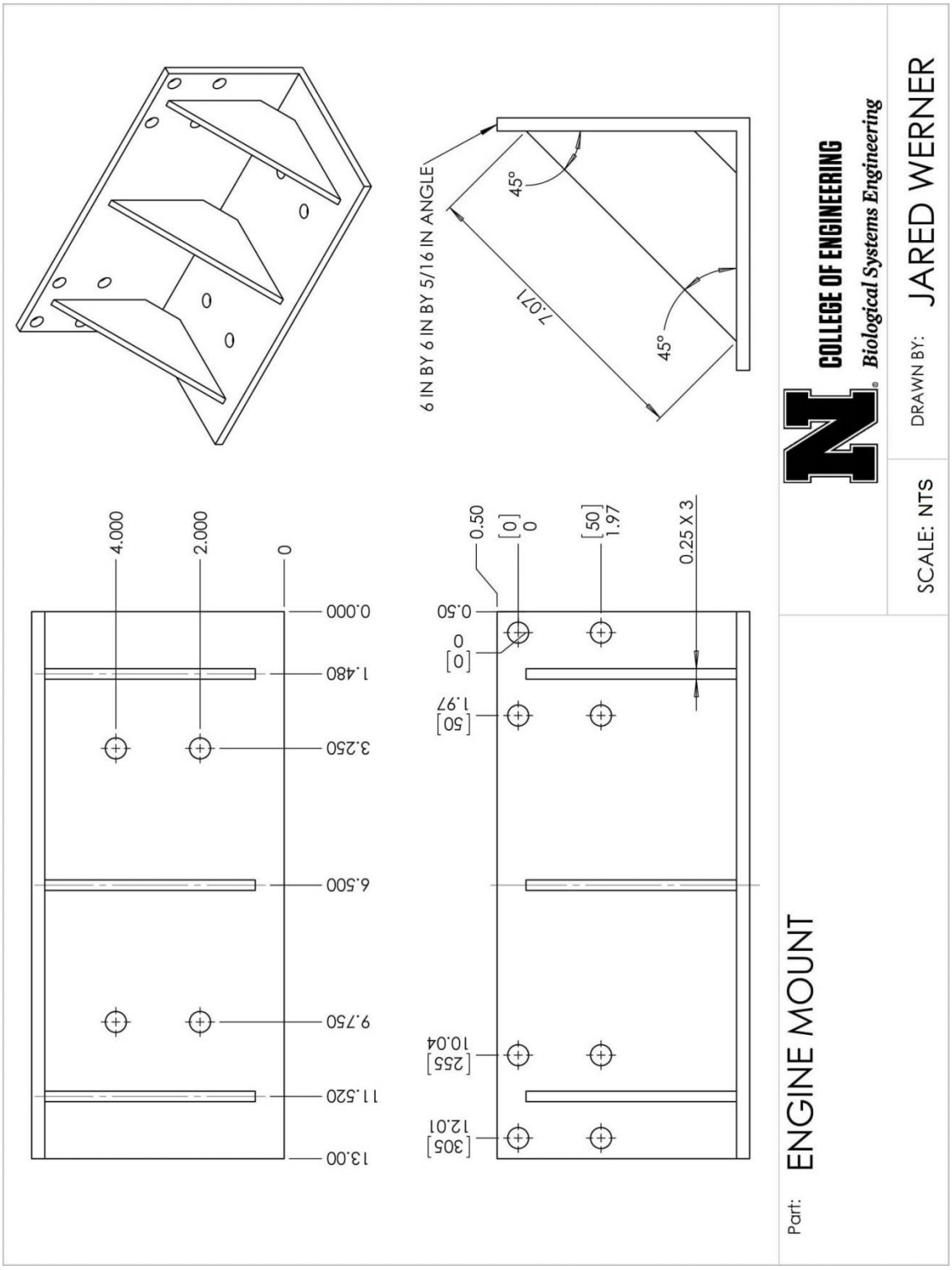



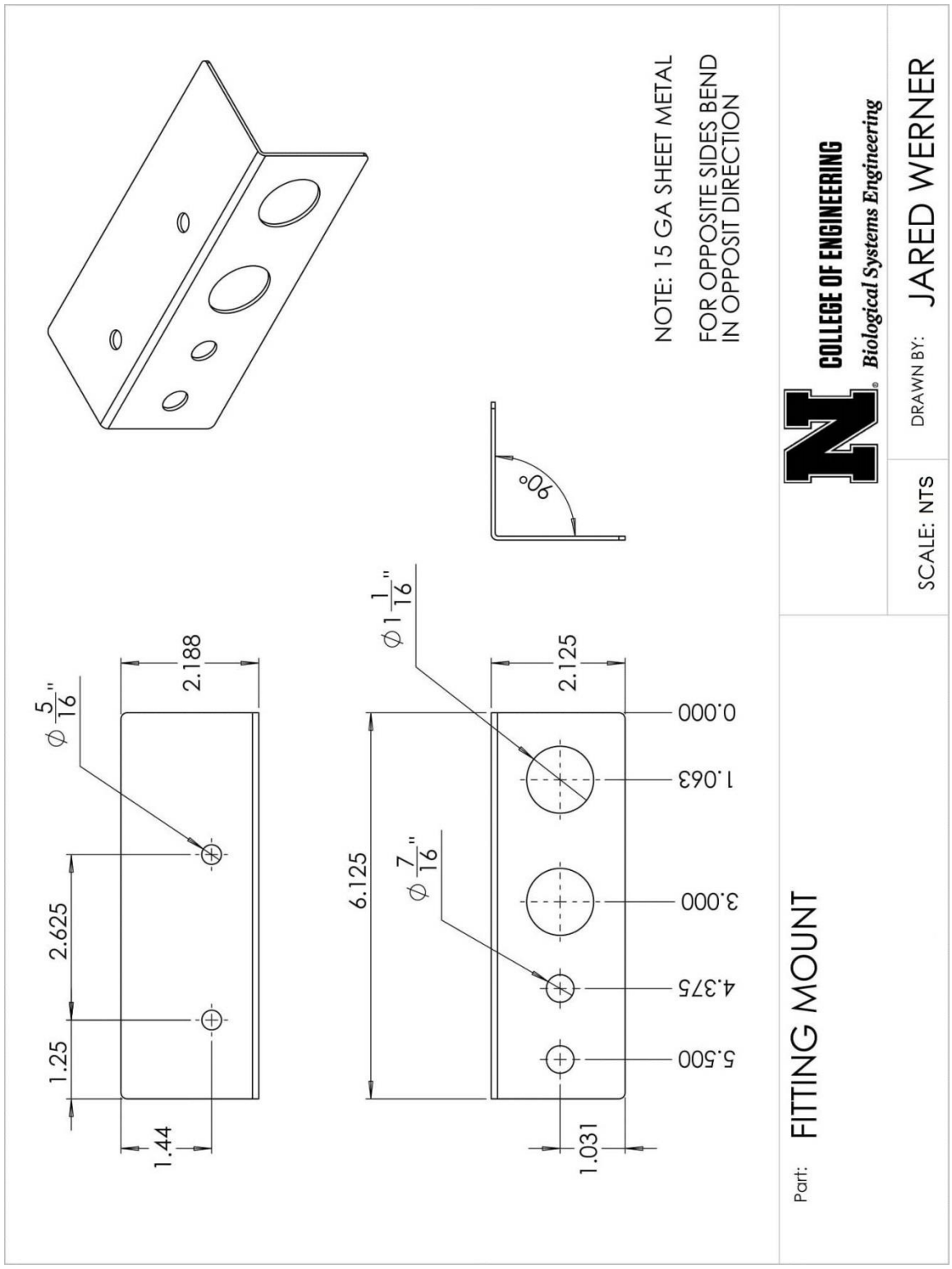

Part: ELECTRICAL HOUSING

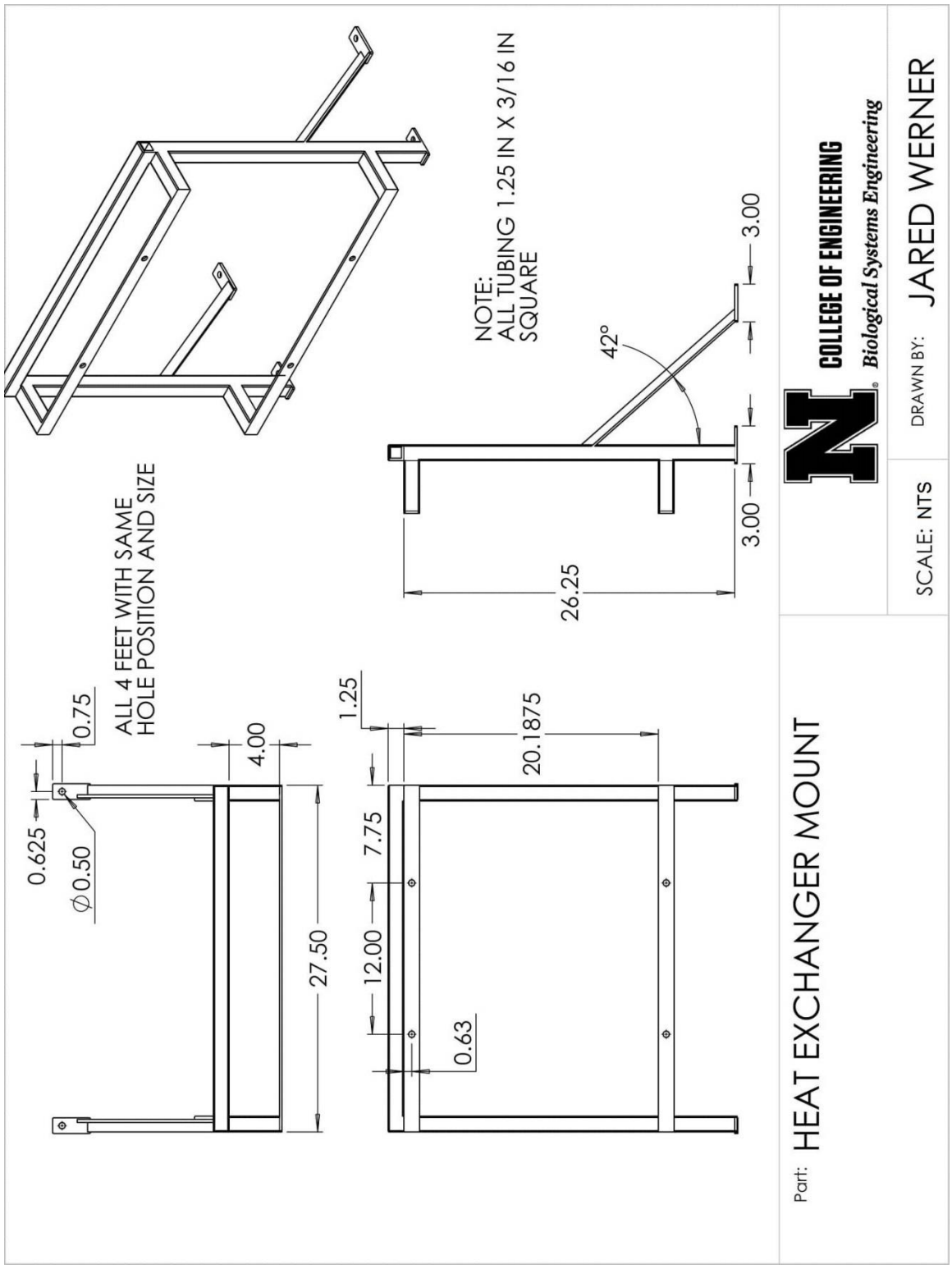
COLLEGE OF ENGINEERING
*Biological Systems Engineering*

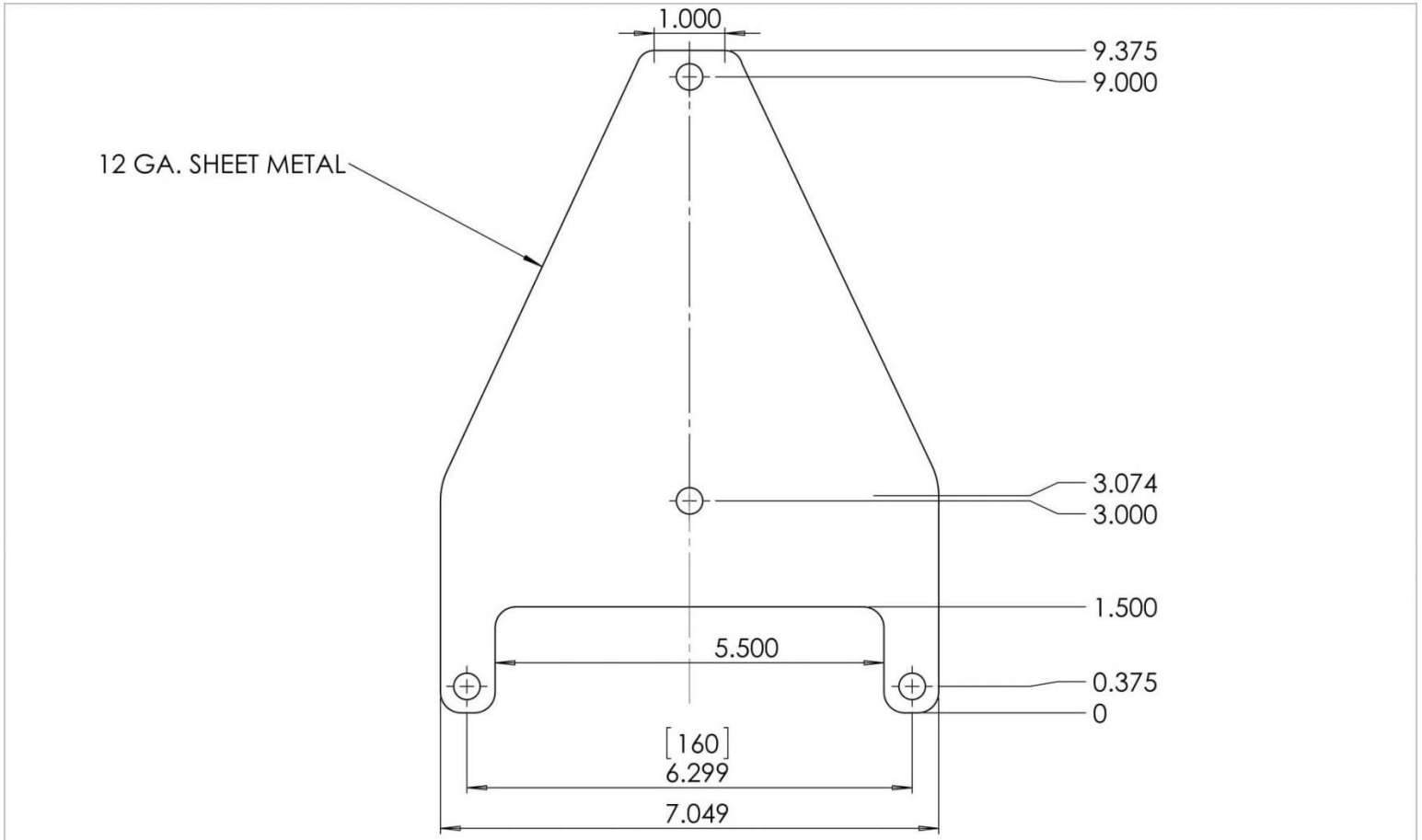
SCALE: NTS

DRAWN BY: JARED WERNER


6 IN BY 6 IN BY 5/16 IN ANGLE
45°
7.071
45°
0.50
[0]
0
[50]
1.97
0.25 X 3
0.50
[0]
0
[50]
1.97
[255]
10.04
[305]
12.01
4.000
2.000
0
0.000
1.480
3.250
6.500
9.750
11.520
13.00
COLLEGE OF ENGINEERING
Biological Systems Engineering
DRAWN BY: JARED WERNER
SCALE: NTS
Part: ENGINE MOUNT


NOTE: 15 GA SHEET METAL
FOR OPPOSITE SIDES BEND IN OPPOSIT DIRECTION
90°
Ø 5/16"
2.188
2.625
1.25
1.44
Ø 1 1/16"
2.125
6.125
Ø 7/16"
0.000
1.063
3.000
4.375
5.500
1.031
COLLEGE OF ENGINEERING
Biological Systems Engineering
Part: FITTING MOUNT
SCALE: NTS
DRAWN BY: JARED WERNER


NOTE:
ALL TUBING 1.25 IN X 3/16 IN
SQUARE
42°
3.00
3.00
26.25
ALL 4 FEET WITH SAME
HOLE POSITION AND SIZE
0.75
0.625
ϕ0.50
4.00
27.50
1.25
20.1875
7.75
12.00
0.63
COLLEGE OF ENGINEERING
Biological Systems Engineering
DRAWN BY: JARED WERNER
SCALE: NTS
Part: HEAT EXCHANGER MOUNT

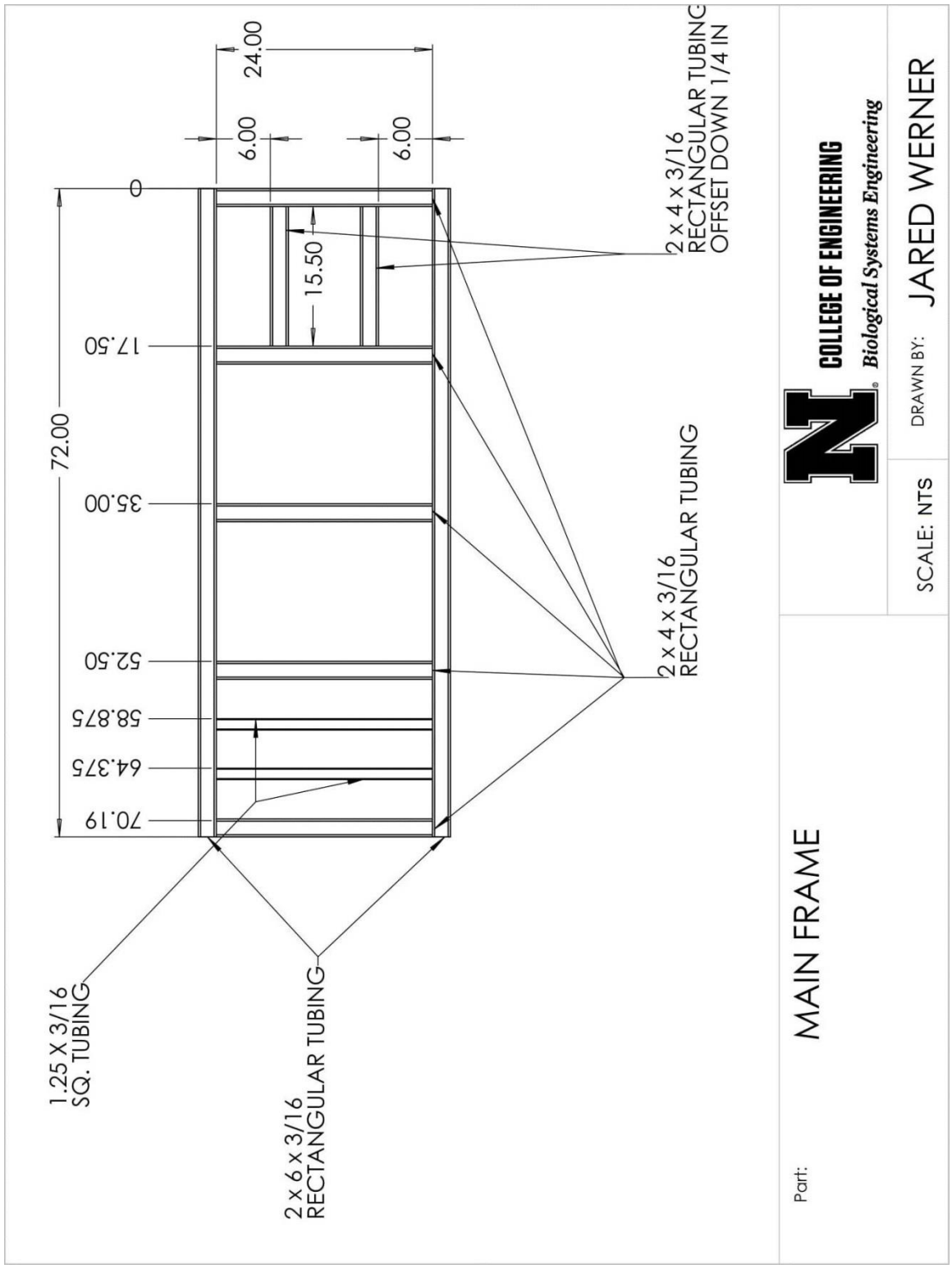


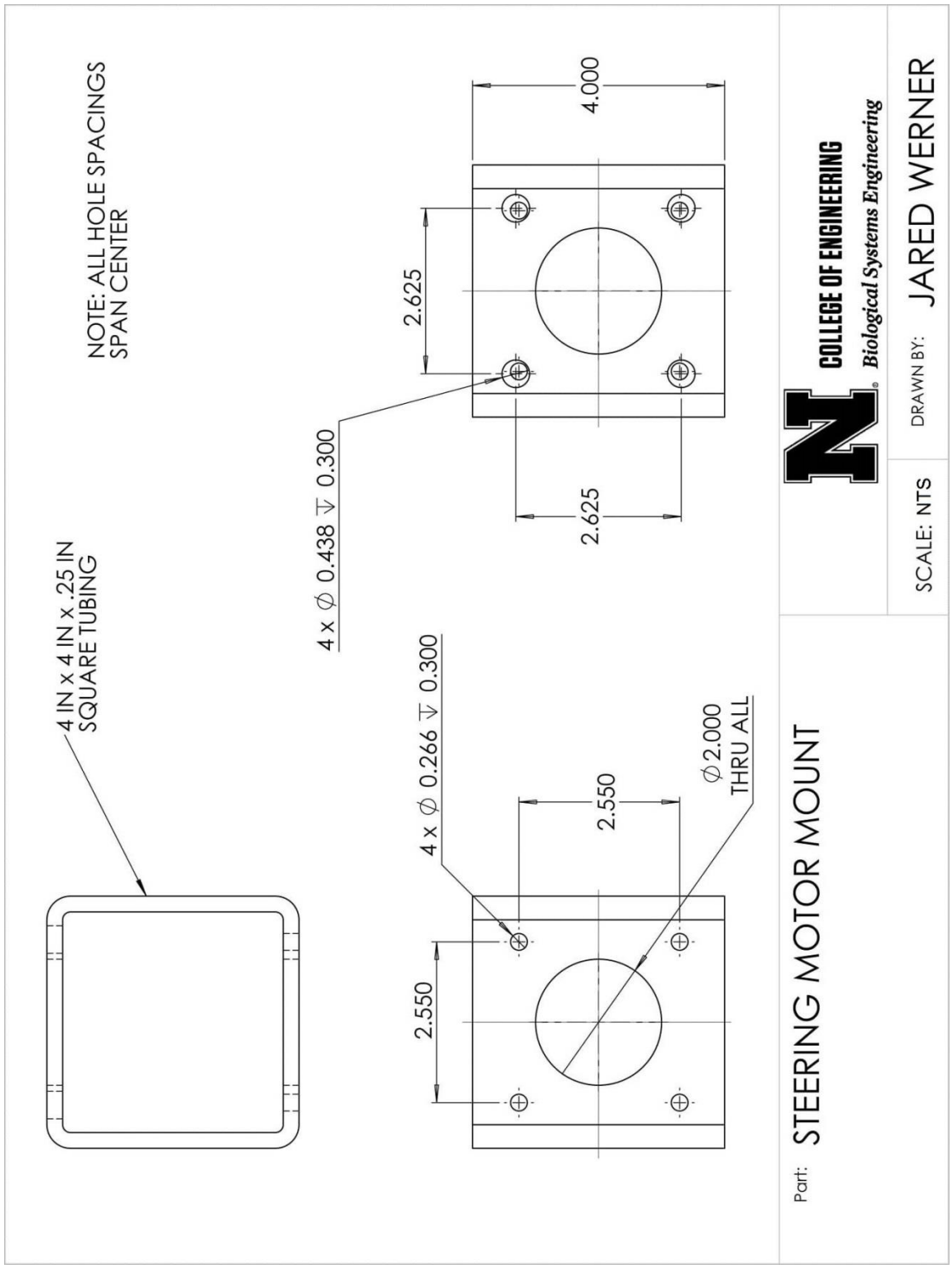

Part: AIR INTAKE MOUNT

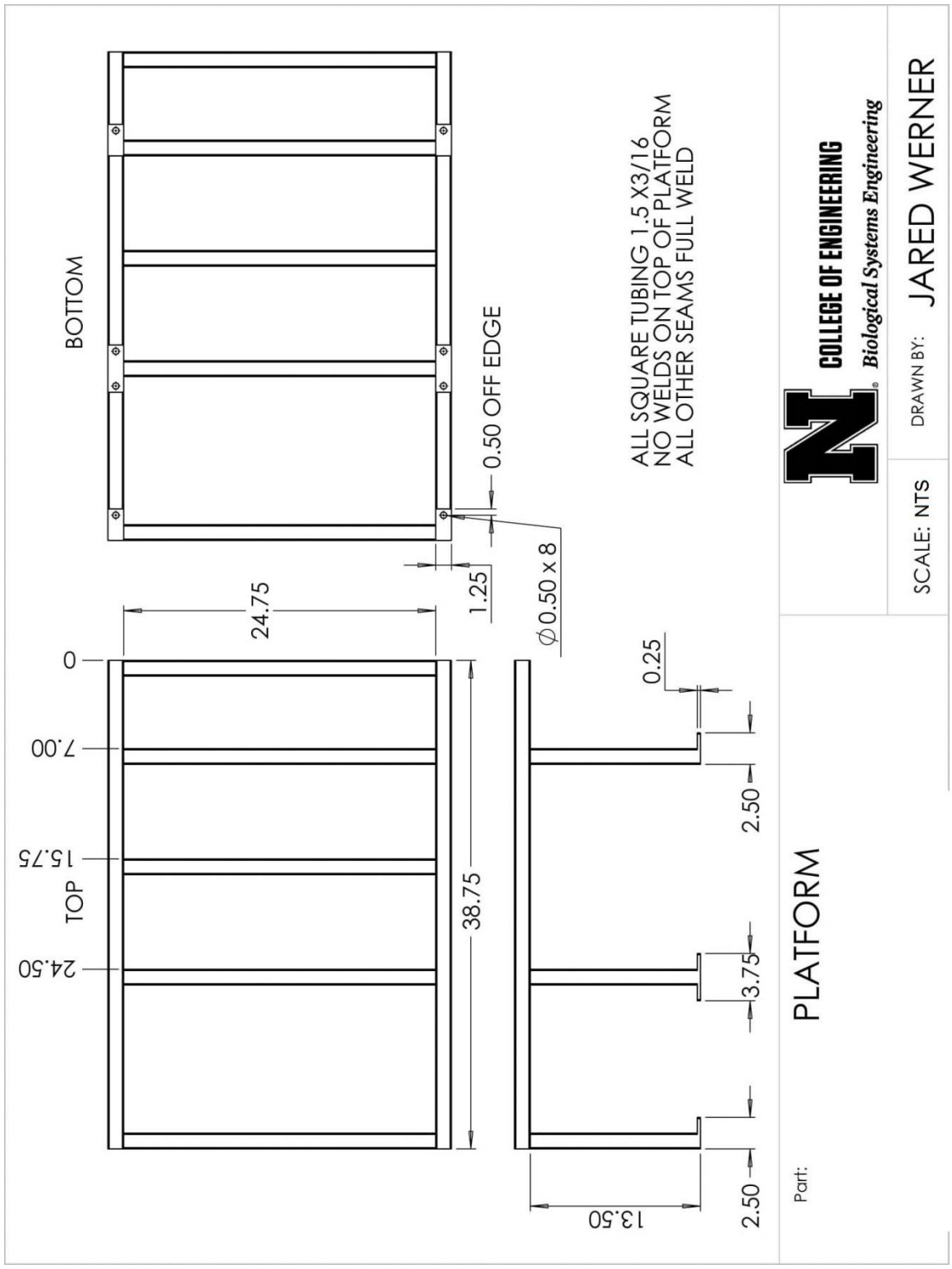
COLLEGE OF ENGINEERING
*Biological Systems Engineering*

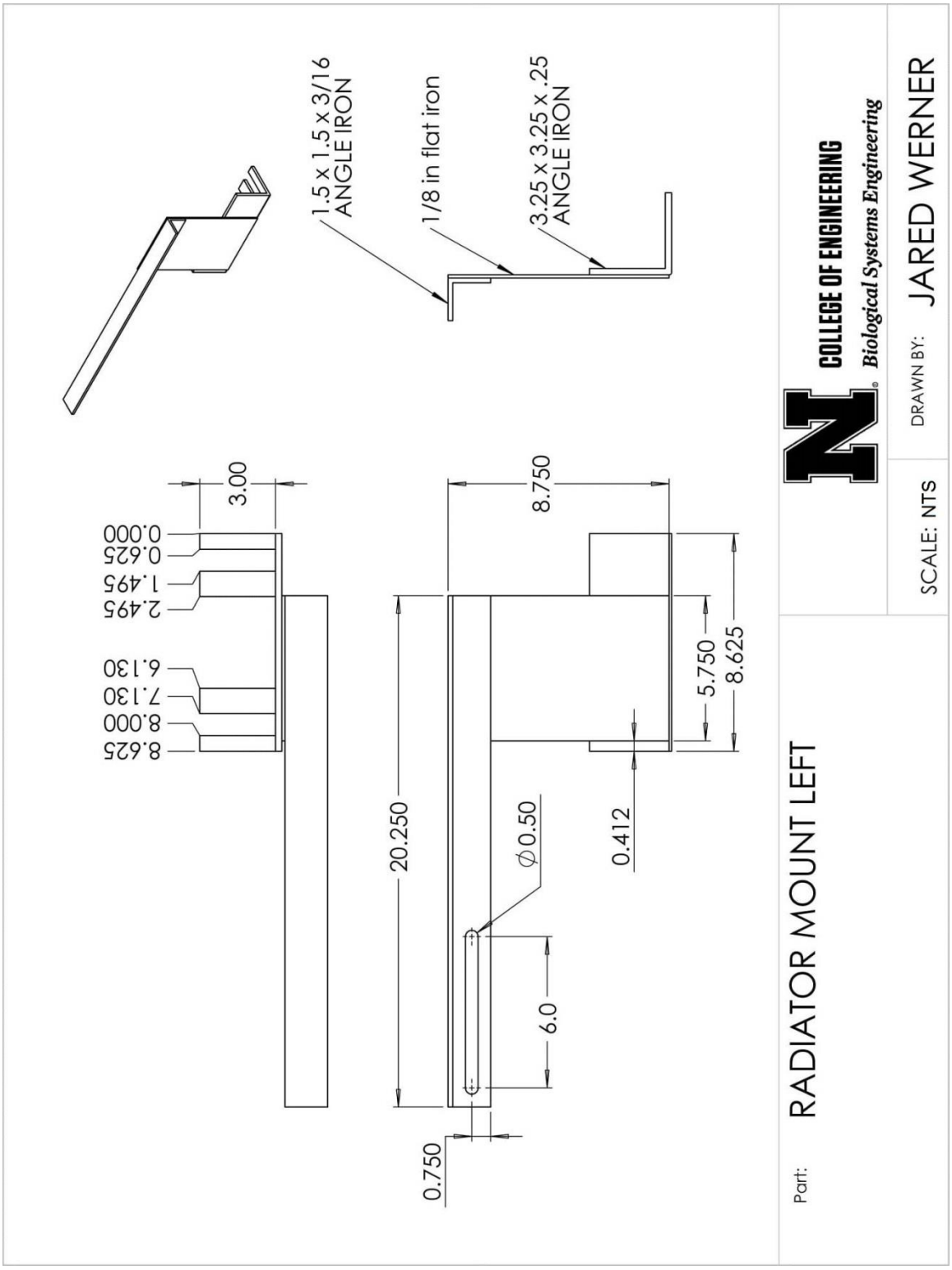
SCALE: NTS

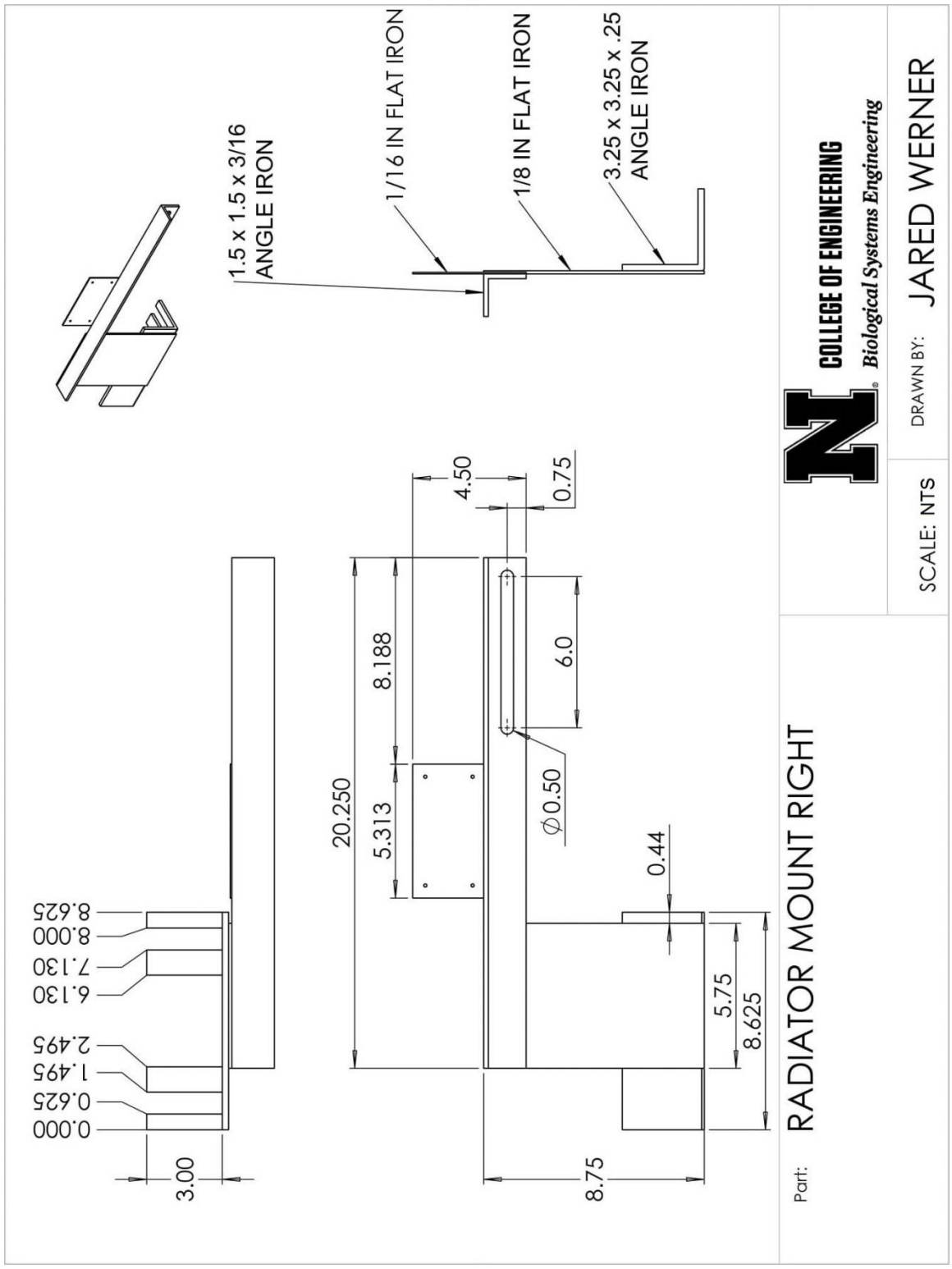
DRAWN BY: JARED WERNER

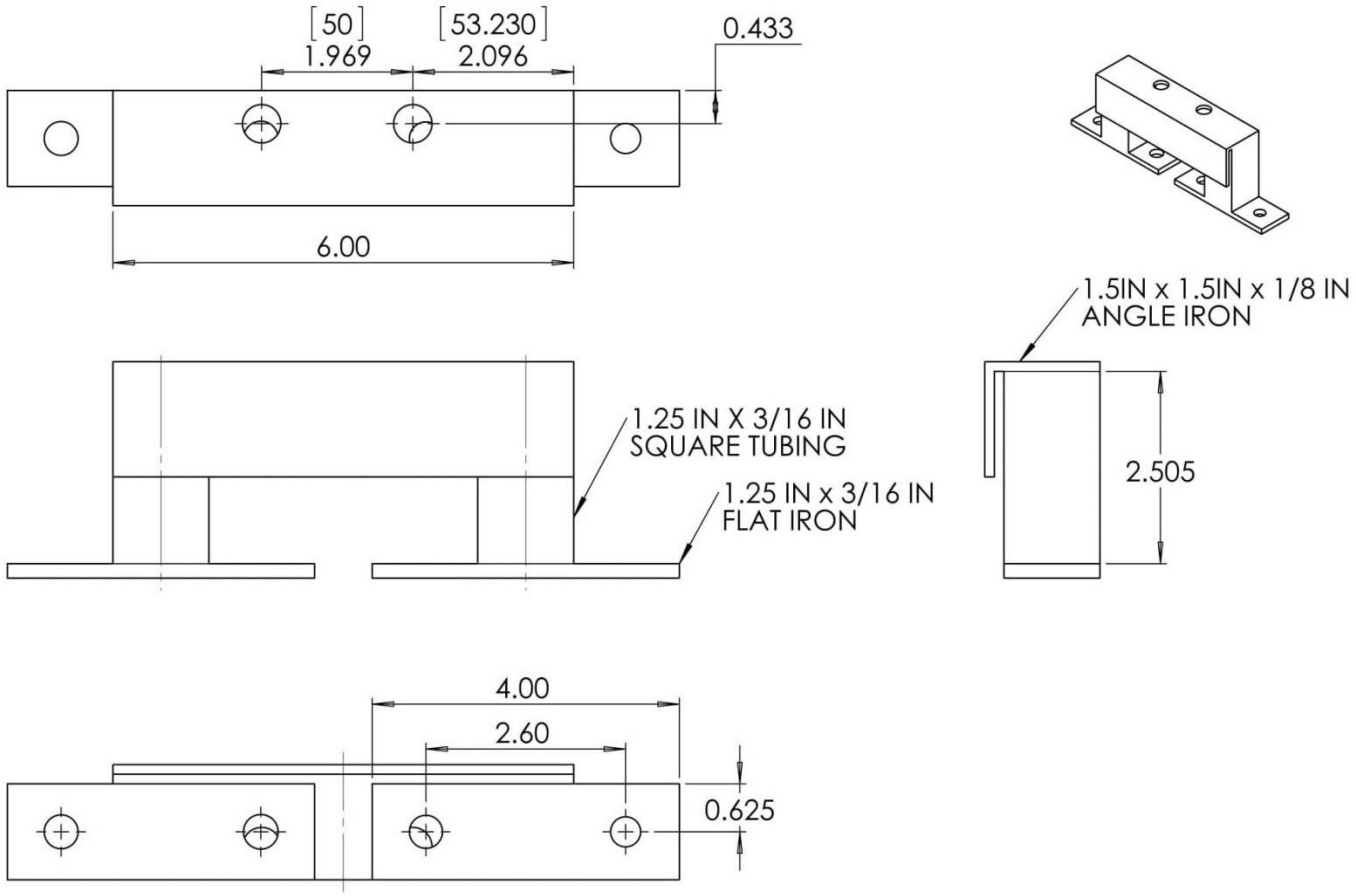

24.00
6.00
6.00
0
15.50
17.50
35.00
52.50
58.875
64.375
70.19
72.00
2 x 4 x 3/16
RECTANGULAR TUBING
OFFSET DOWN 1/4 IN
2 x 4 x 3/16
RECTANGULAR TUBING
1.25 X 3/16
SQ. TUBING
2 x 6 x 3/16
RECTANGULAR TUBING
COLLEGE OF ENGINEERING
Biological Systems Engineering
Part:
MAIN FRAME
SCALE: NTS
DRAWN BY:
JARED WERNER


NOTE: ALL HOLE SPACINGS
SPAN CENTER
4 IN x 4 IN x .25 IN
SQUARE TUBING
4 x ∅ 0.438 ∓ 0.300
2.625
4.000
2.625
4 x ∅ 0.266 ∓ 0.300
2.550
∅2.000
THRU ALL
2.550
Part: STEERING MOTOR MOUNT
COLLEGE OF ENGINEERING
Biological Systems Engineering
SCALE: NTS
DRAWN BY: JARED WERNER


BOTTOM
TOP
0
7.00
15.75
24.50
24.75
38.75
1.25
0.50 OFF EDGE
Ø0.50 x 8
ALL SQUARE TUBING 1.5 X3/16
NO WELDS ON TOP OF PLATFORM
ALL OTHER SEAMS FULL WELD
0.25
2.50
3.75
2.50
13.50
COLLEGE OF ENGINEERING
Biological Systems Engineering
DRAWN BY: JARED WERNER
SCALE: NTS
Part: PLATFORM


1.5 x 1.5 x 3/16
ANGLE IRON
1/8 in flat iron
3.25 x 3.25 x .25
ANGLE IRON
3.00
0.000
0.625
1.495
2.495
6.130
7.130
8.000
8.625
8.750
20.250
Ø0.50
6.0
0.750
0.412
5.750
8.625
COLLEGE OF ENGINEERING
Biological Systems Engineering
Part: RADIATOR MOUNT LEFT
SCALE: NTS
DRAWN BY: JARED WERNER


1.5 x 1.5 x 3/16
ANGLE IRON
1/16 IN FLAT IRON
1/8 IN FLAT IRON
3.25 x 3.25 x .25
ANGLE IRON
COLLEGE OF ENGINEERING
Biological Systems Engineering
DRAWN BY: JARED WERNER
SCALE: NTS
Part: RADIATOR MOUNT RIGHT
0.000
0.625
1.495
2.495
6.130
7.130
8.000
8.625
3.00
20.250
8.188
5.313
4.50
0.75
6.0
ϕ0.50
0.44
5.75
8.625
8.75

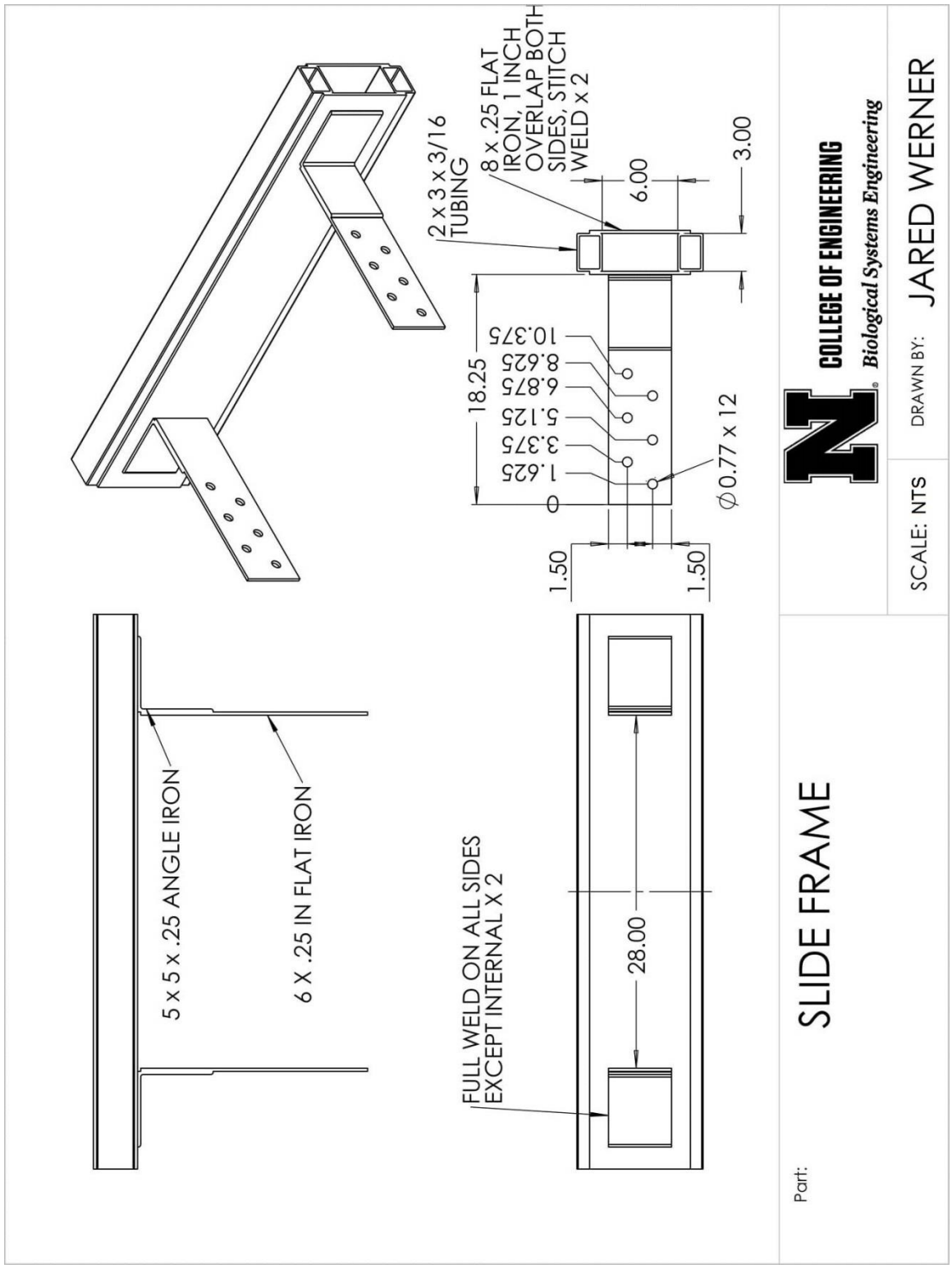


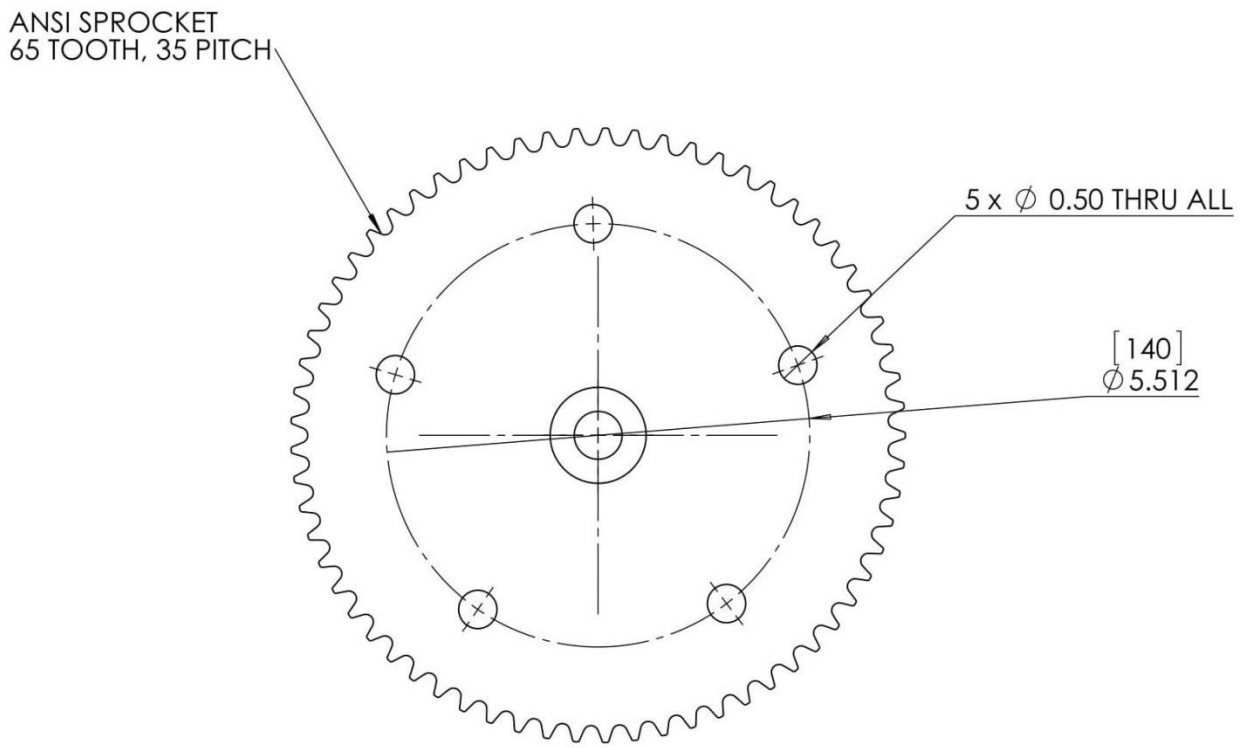

Part: REAR PUMP MOUNT

COLLEGE OF ENGINEERING
*Biological Systems Engineering*

SCALE: NTS

DRAWN BY: JARED WERNER


5 x 5 x .25 ANGLE IRON
6 X .25 IN FLAT IRON
FULL WELD ON ALL SIDES EXCEPT INTERNAL X 2
28.00
2 x 3 x 3/16 TUBING
8 x .25 FLAT IRON, 1 INCH OVERLAP BOTH SIDES, STITCH WELD x 2
6.00
3.00
18.25
10.375
8.625
6.875
5.125
3.375
1.625
0
ϕ0.77 x 12
1.50
1.50
COLLEGE OF ENGINEERING
Biological Systems Engineering
Part:
SLIDE FRAME
SCALE: NTS
DRAWN BY:
JARED WERNER

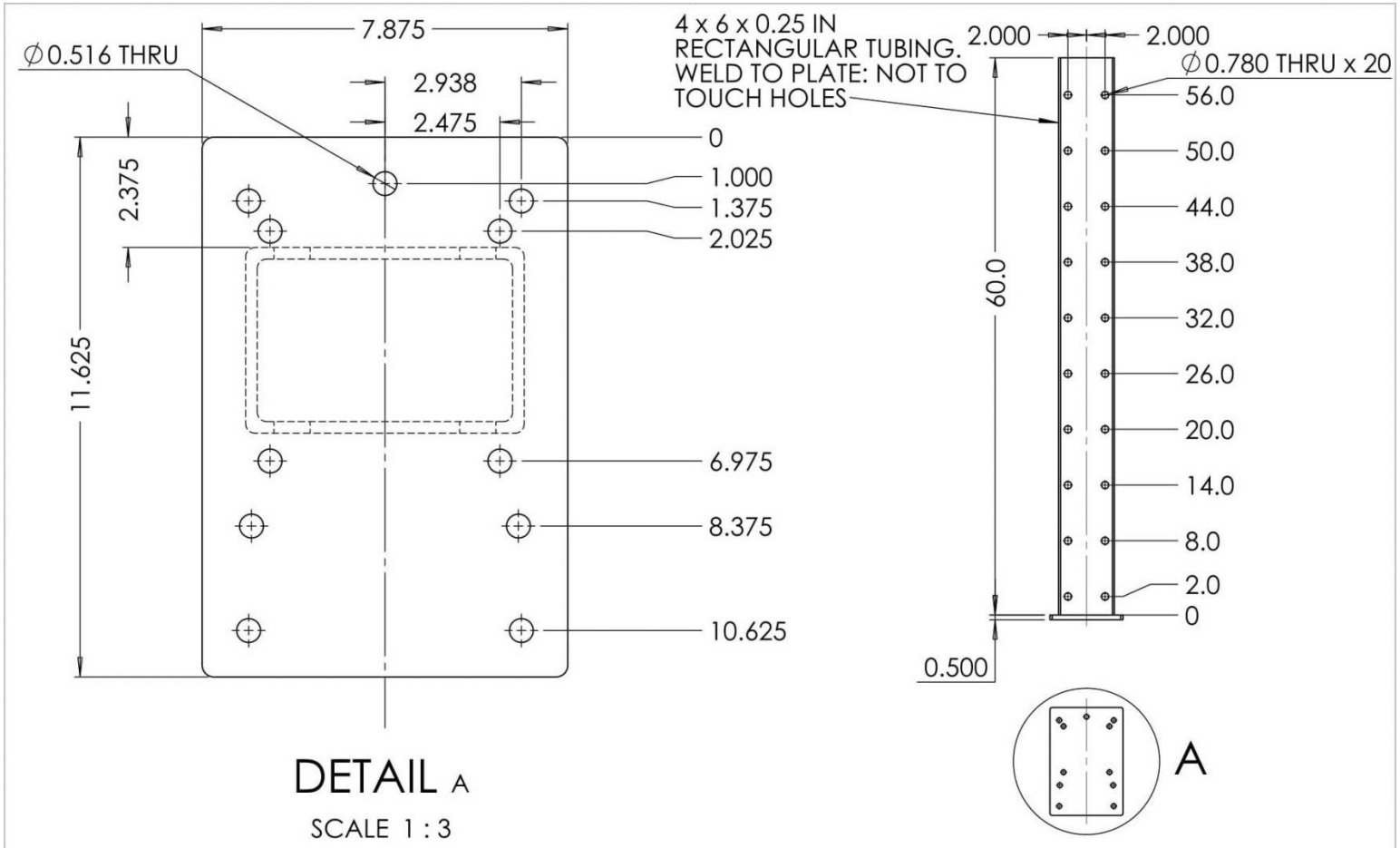



Part: DATA COLLECTION SPROCKET

COLLEGE OF ENGINEERING
*Biological Systems Engineering*

SCALE: NTS

DRAWN BY: JARED WERNER




DETAIL A

SCALE 1 : 3

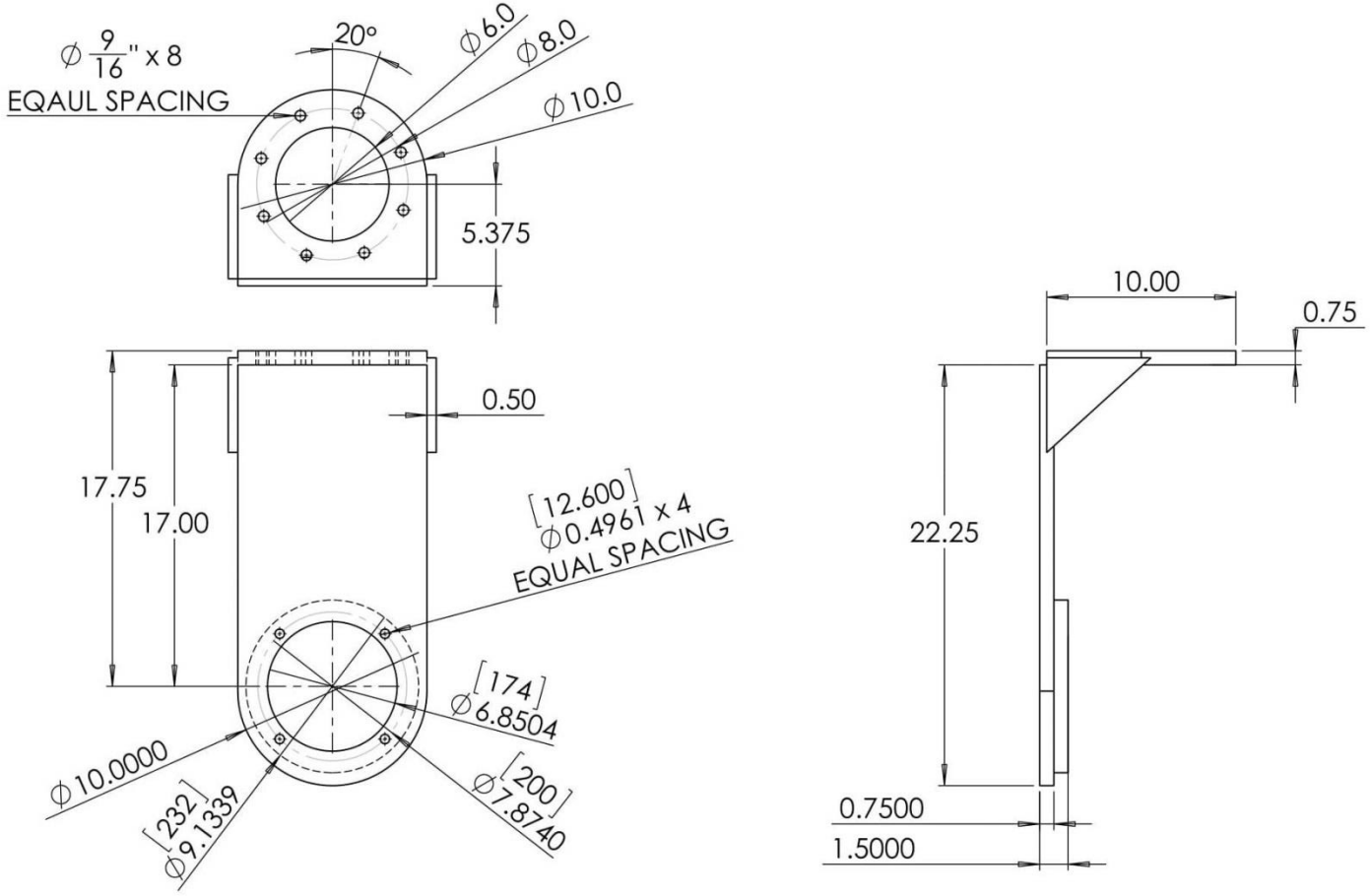
Part: UPRIGHT

COLLEGE OF ENGINEERING

*Biological Systems Engineering*

SCALE: NTS

DRAWN BY: JARED WERNER




Part: WHEEL MOUNT

COLLEGE OF ENGINEERING
*Biological Systems Engineering*

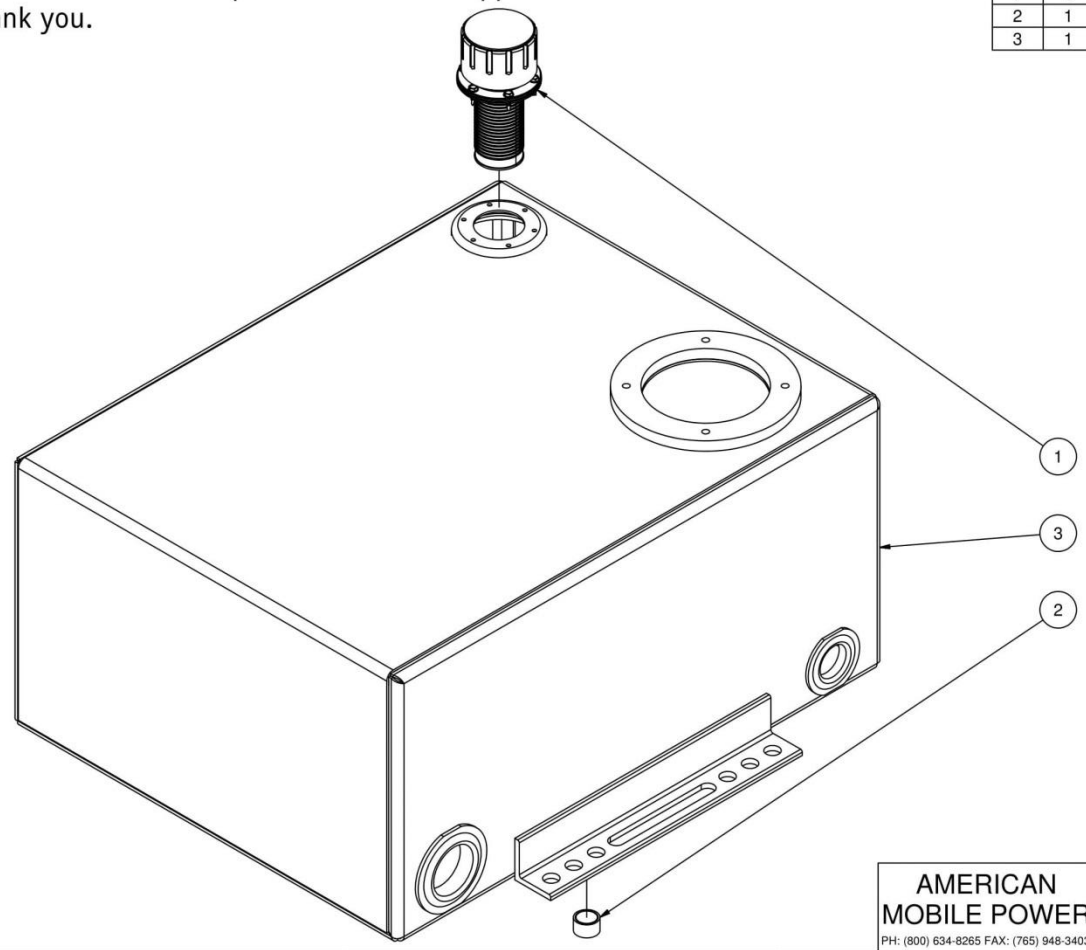
SCALE: NTS

DRAWN BY: JARED WERNER

Please review design and remit with approval or changes.
Your order will not be processed without approval.
Thank you.

| Parts List | | | |
|---|---|---|---|
| ITEM | QTY | PART NUMBER | DESCRIPTION |
| 1 | 1 | 57XL-40N | 40 MICRON FILL W/NYLON STRAINER |
| 2 | 1 | APP0750-M | 3/4" NPT MAG PLUG |
| 3 | 1 | N_SS3297UNE | NAM WELDMENT ~91.5 lbs |

NOTES:
-POWDER COAT BLACK
-APPLY RUST INHIBITOR INSIDE
-PLUG ALL PORTS FOR SHIPMENT




**SUBMITTED FOR CUSTOMER APPROVAL**

☑ APPROVED AS SHOWN

☐ APPROVED WITH NOTED CHANGES

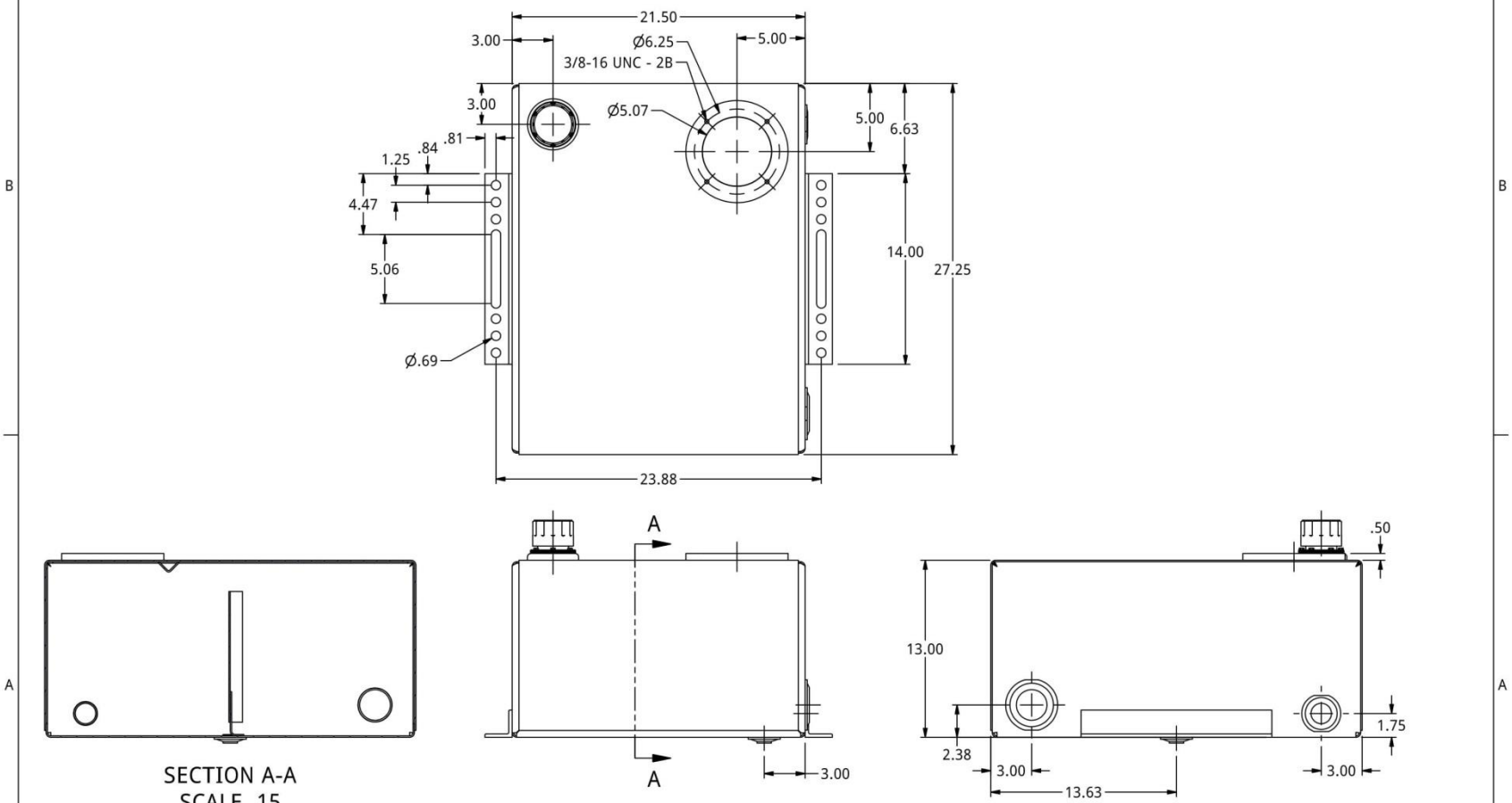
☐ RESUBMIT WITH NOTED CHANGES

APPROVED BY: [redacted] Jared Werner 12/21/2015

| | | | | |
|---|---|---|---|---|
| AMERICAN MOBILE POWER<br>PH: (800) 634-8265 FAX: (765) 948-3403 | DRAWING BY: BPKNIGHT<br>COMPLETED: 12/17/2015 | DESIGN FOR: SUNSOURCE MOBILE ENGINEERING CENTER | FILE NAME: SS3297UNE<br>SHEET: 1 of 8 | SHEET SIZE: B |

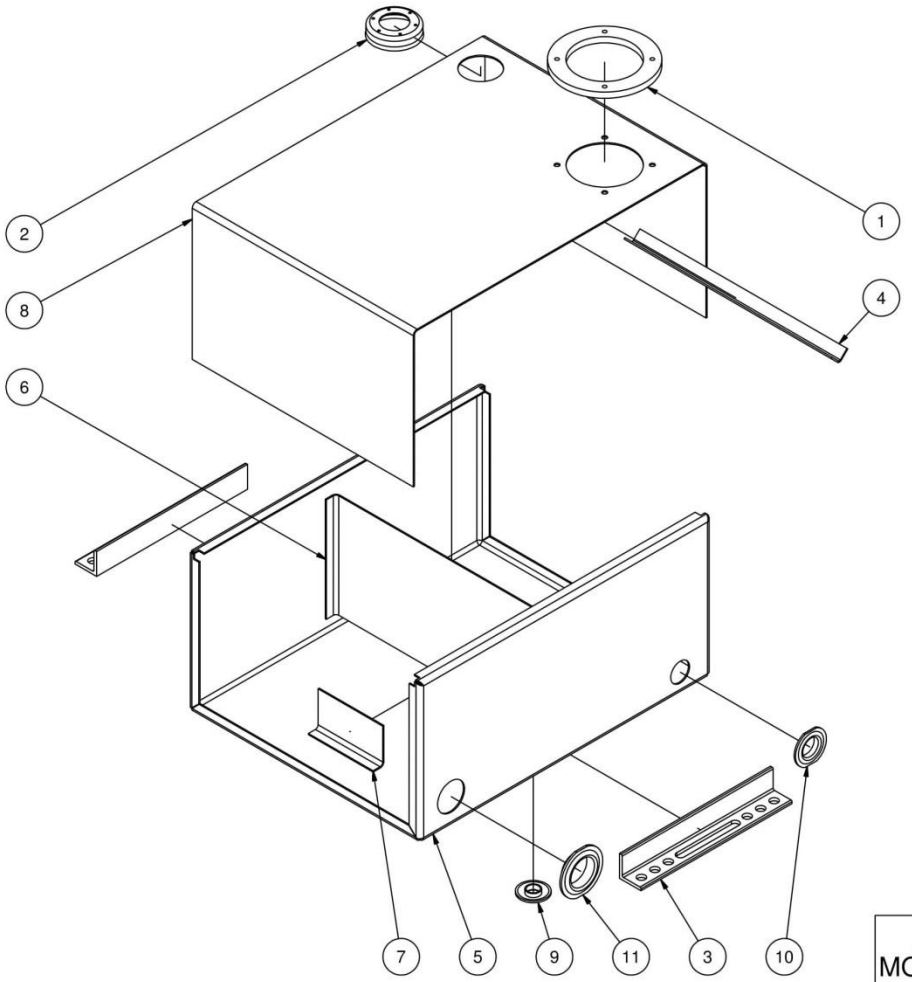
ATTENTION: ENGINEERING

OUR ENGINEERING DEPARTMENT HAS DESIGNED THIS TANK SYSTEM TO MEET YOUR SPECIFICATIONS/REQUIREMENTS. AMERICAN MOBILE POWER ASSURES ITS PRODUCT(S) TO BE OF SUPERIOR QUALITY WITHIN ITS DESIGN PARAMETERS.
THE COMPANY CANNOT DETERMINE SUITABILTY OF THIS PRODUCT, AS DESIGNED, FOR A PARTICULAR APPLICATION WITHOUT COMPLETE ENGINEERING AND STRESS SPECIFICATIONS FROM THE CUSTOMER.
PURCHASER ASSUMES RESPONSIBILITY FOR FINAL SUITABILITY, INSTALLATION, AND MOUNTING IN ORDER TO INSURE FULL LIFE OF THIS PRODUCT IN ITS SPECIFIC AND INTENDED USE.



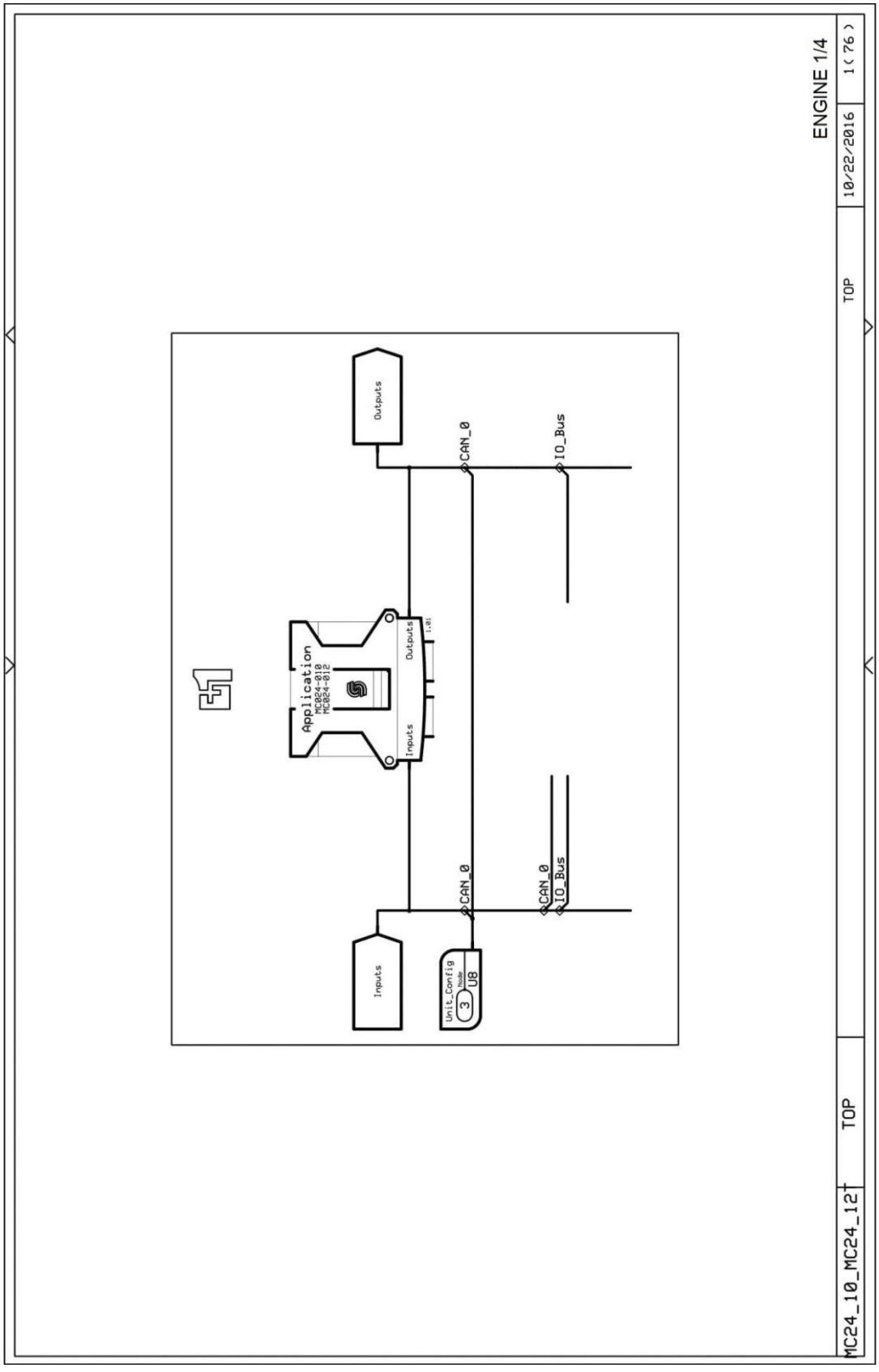

SECTION A-A
SCALE .15

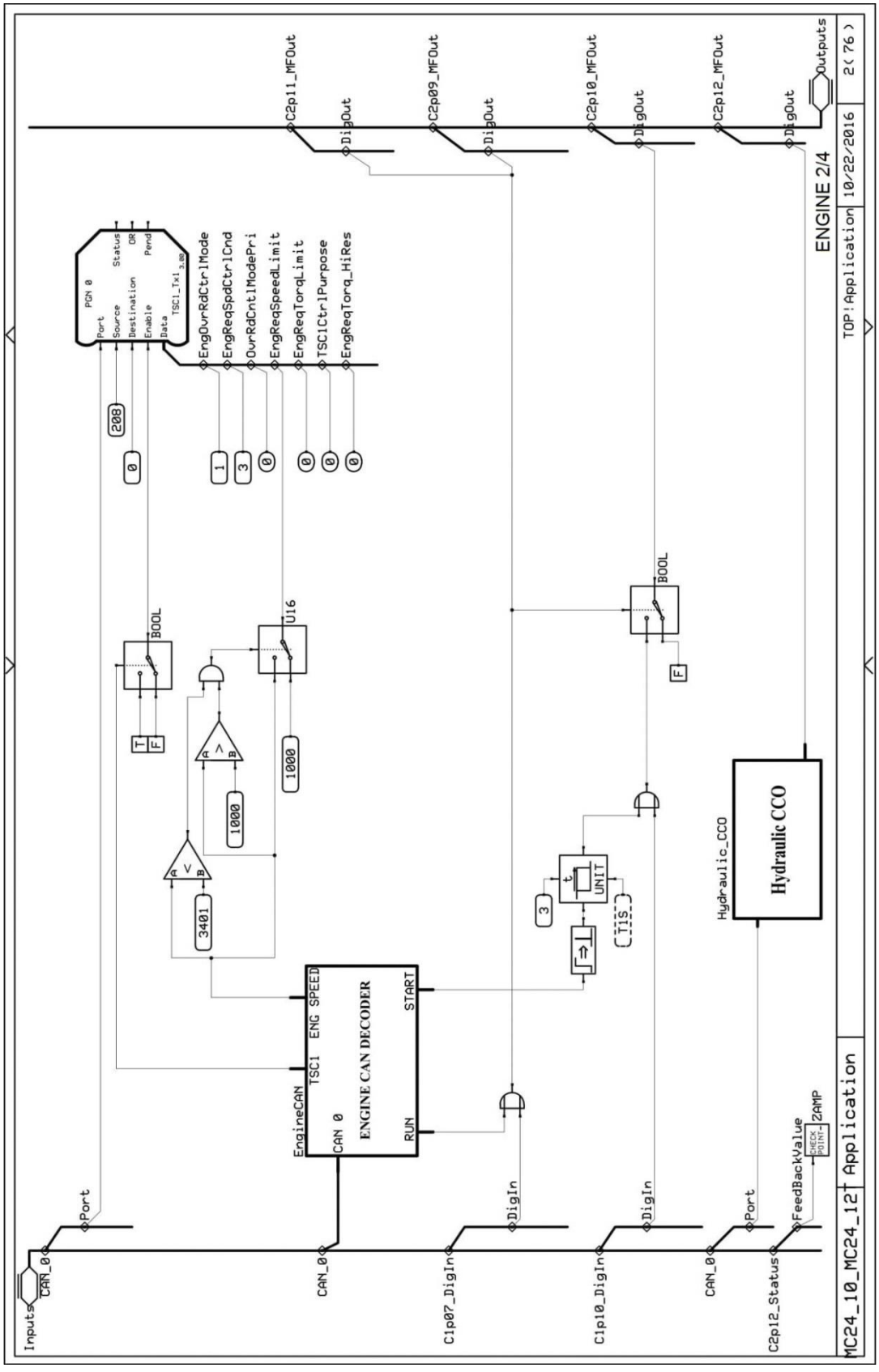
| | | | | |
|---|---|---|---|---|
| AMERICAN MOBILE POWER<br>PH: (800) 634-8265 FAX: (765) 948-3403 | DRAWING BY: BPKNIGHT<br>COMPLETED: 12/17/2015 | DESIGN FOR: SUNSOURCE MOBILE ENGINEERING CENTER | FILE NAME: SS3297UNE<br>SHEET: 2 of 8 | SHEET SIZE: B |

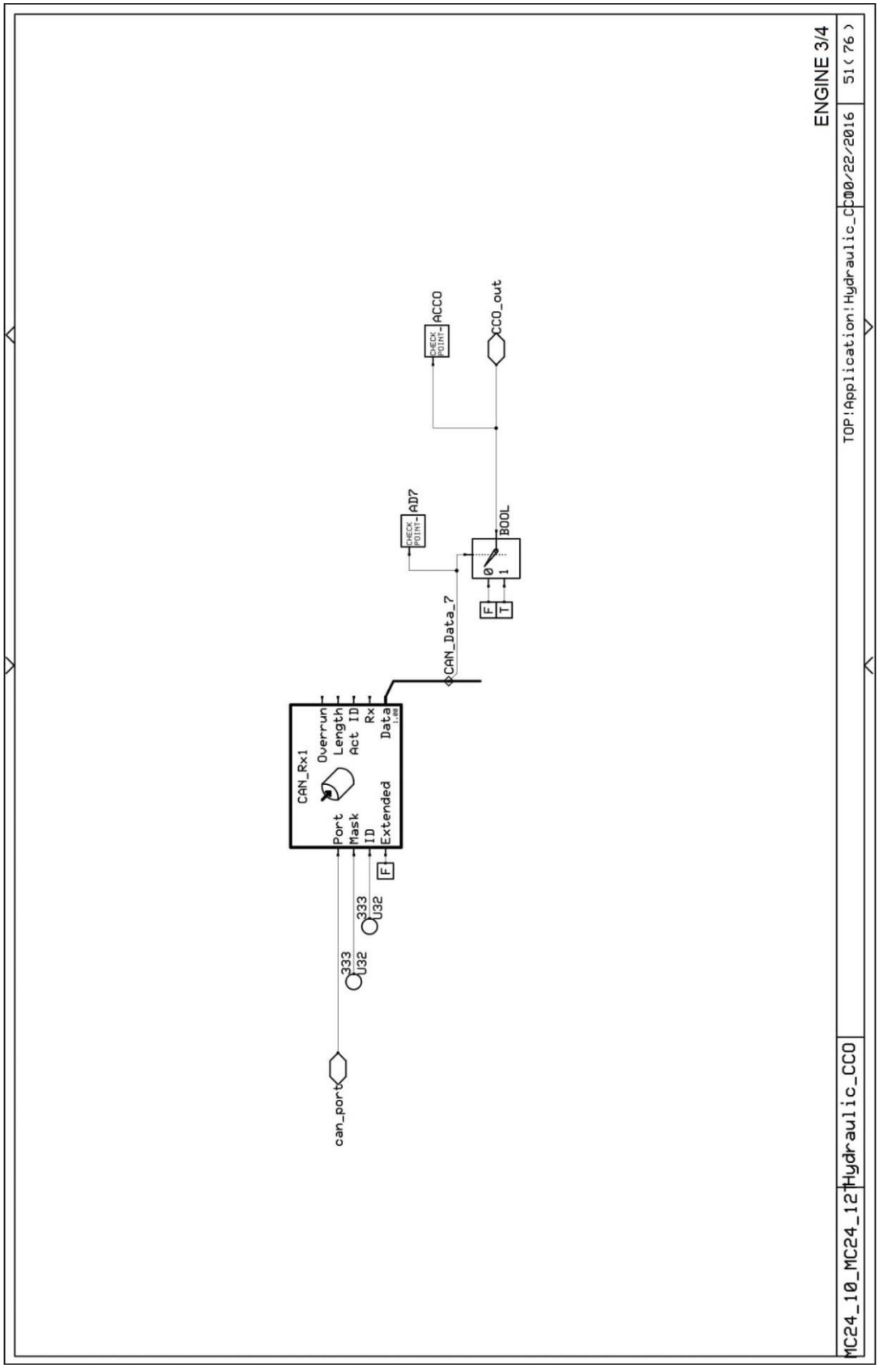


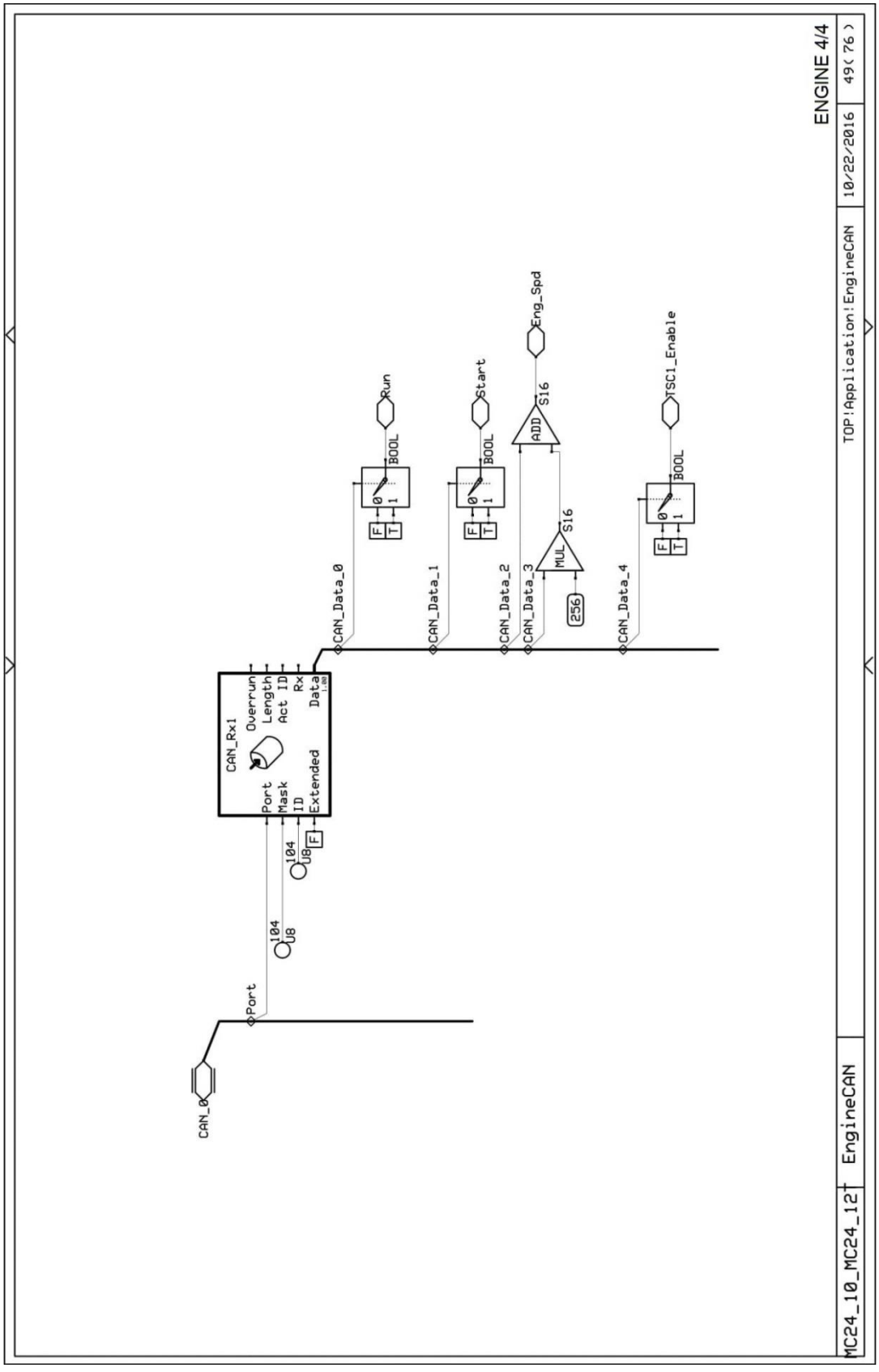

| Parts List | | | | |
|---|---|---|---|---|
| ITEM | QTY | PART NUMBER | DESCRIPTION | Fab Qty |
| 1 | 1 | AFF-3-1250 | .50 x 7.5 O.D. HRPO Filter Flange | |
| 2 | 1 | AFS-301 | 13 GA HRPO Fill Ass'y Weld Flange | |
| 3 | 2 | ANGLE14MFG | 2 X 2 X 1/4" MFG ANGLE | |
| 4 | 1 | ASTF12-3192 | 12 GA HRPO x 19.2 STIFFENER | |
| 5 | 1 | SS3297UNEB | 12 GA HRPO BOTTOM SECTION | |
| 6 | 1 | SS3297UNEF | 12 GA HRPO BAFFLE | |
| 7 | 1 | SS3297UNEL | 12 GA HRPO BOTTOM STIFFENER | |
| 8 | 1 | SS3297UNET | 12 GA HRPO TOP SECTION | |
| 9 | 1 | SWF0750 | 3/4" NPTF STEEL WELD FLANGE | |
| 10 | 1 | SWF1250 | 1 1/4" NPSC STEEL WELD FLANGE | |
| 11 | 1 | SWF2000 | 2" NPSC STEEL WELD FLANGE | |

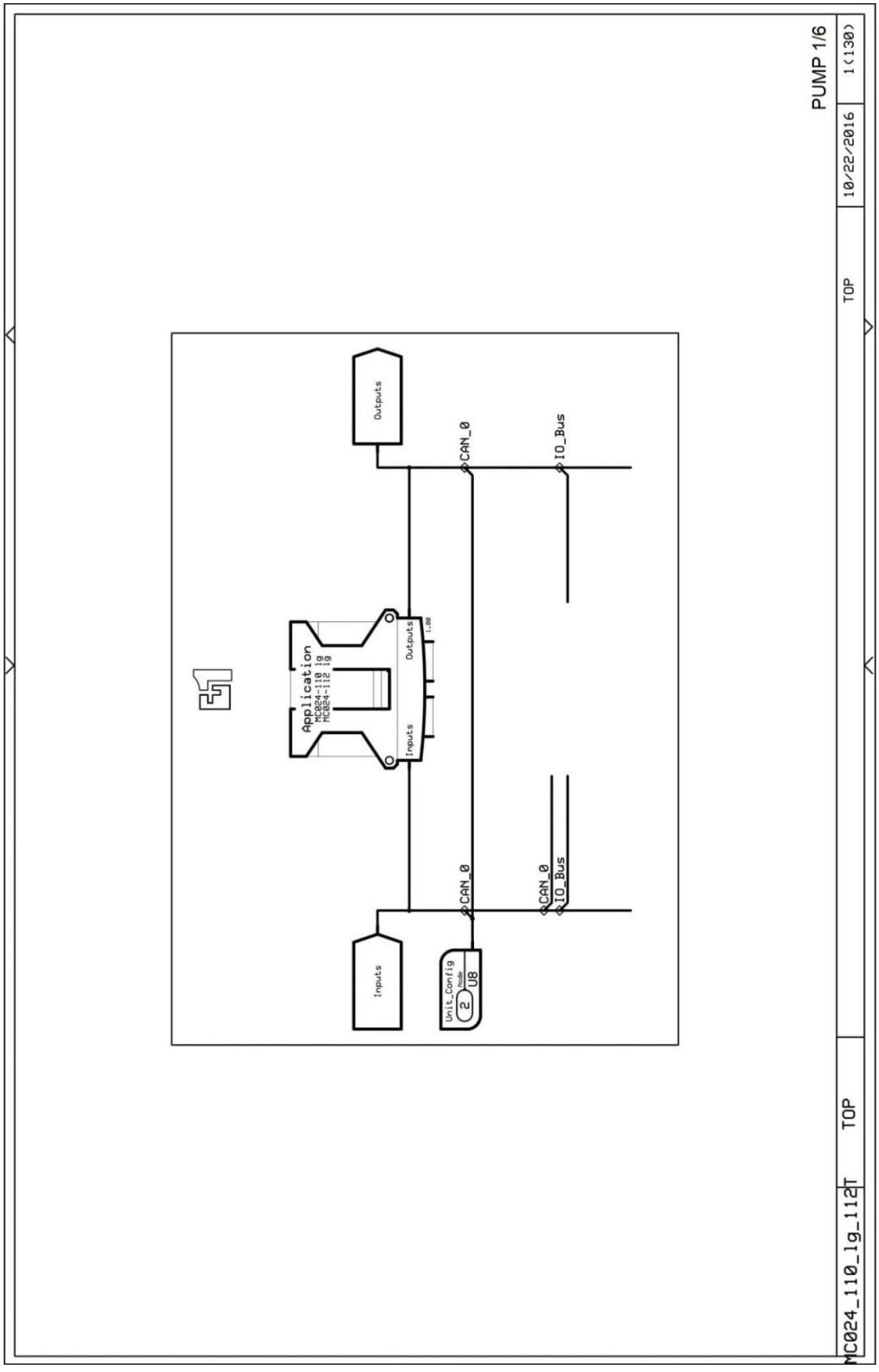
| | | | | |
|---|---|---|---|---|
| AMERICAN MOBILE POWER<br>PH: (800) 634-8265 FAX: (765) 948-3403 | DRAWING BY: BPKNIGHT<br>COMPLETED: 12/17/2015 | DESIGN FOR: SUNSOURCE MOBILE ENGINEERING CENTER | FILE NAME: SS3297UNE | |
| | | | SHEET: 3 of 8 | SHEET SIZE: B |

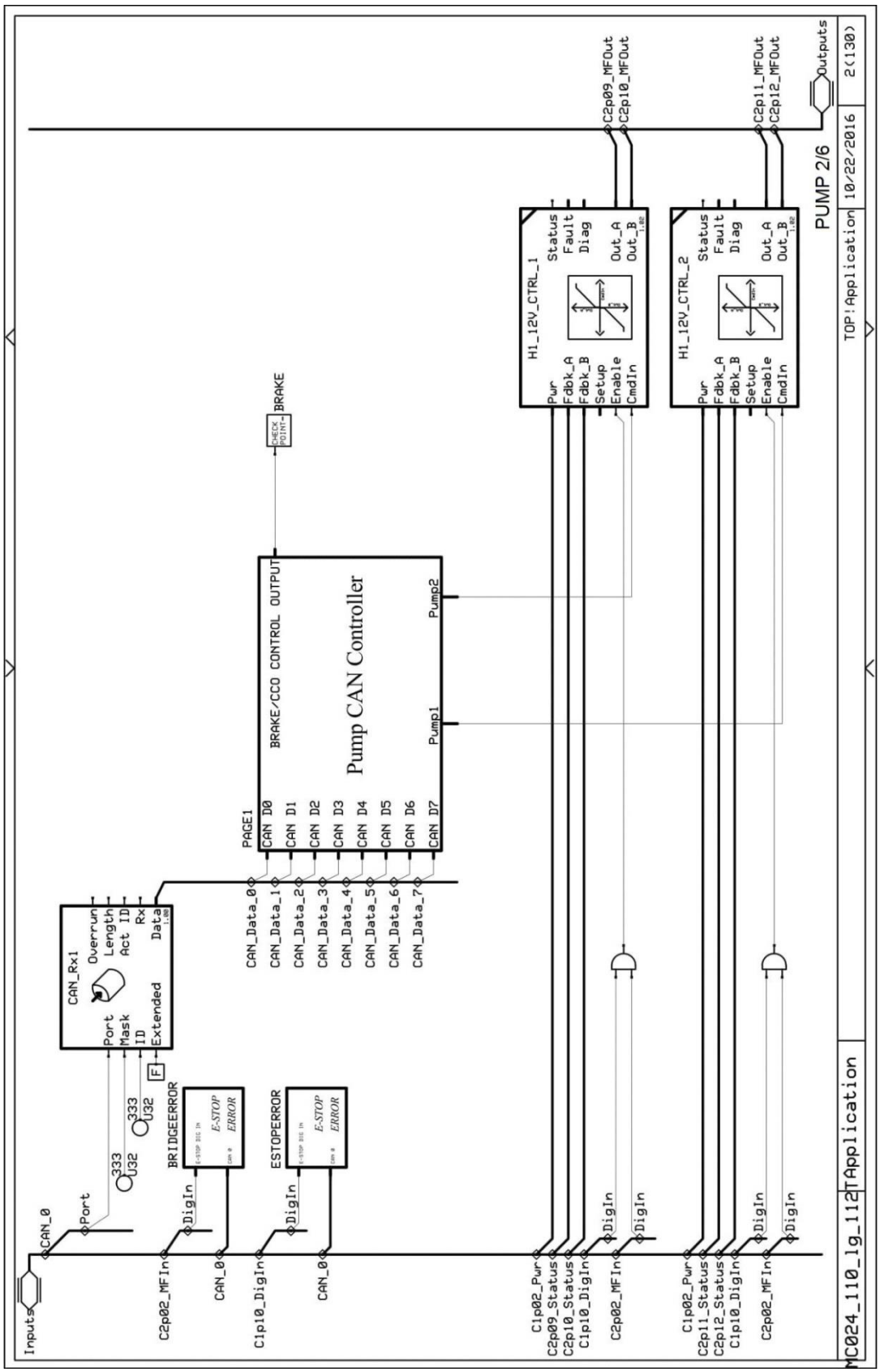

ENGINE 1/4
1 ( 76 )
10/22/2016
TOP
MC24_10_MC24_12
TOP
Application
MC824-810
MC824-812
Outputs
Inputs
Outputs
Inputs
CAN_0
IO_Bus
CAN_0
IO_Bus
CAN_0
IO_Bus
Unit_Config
Mode
3
U8

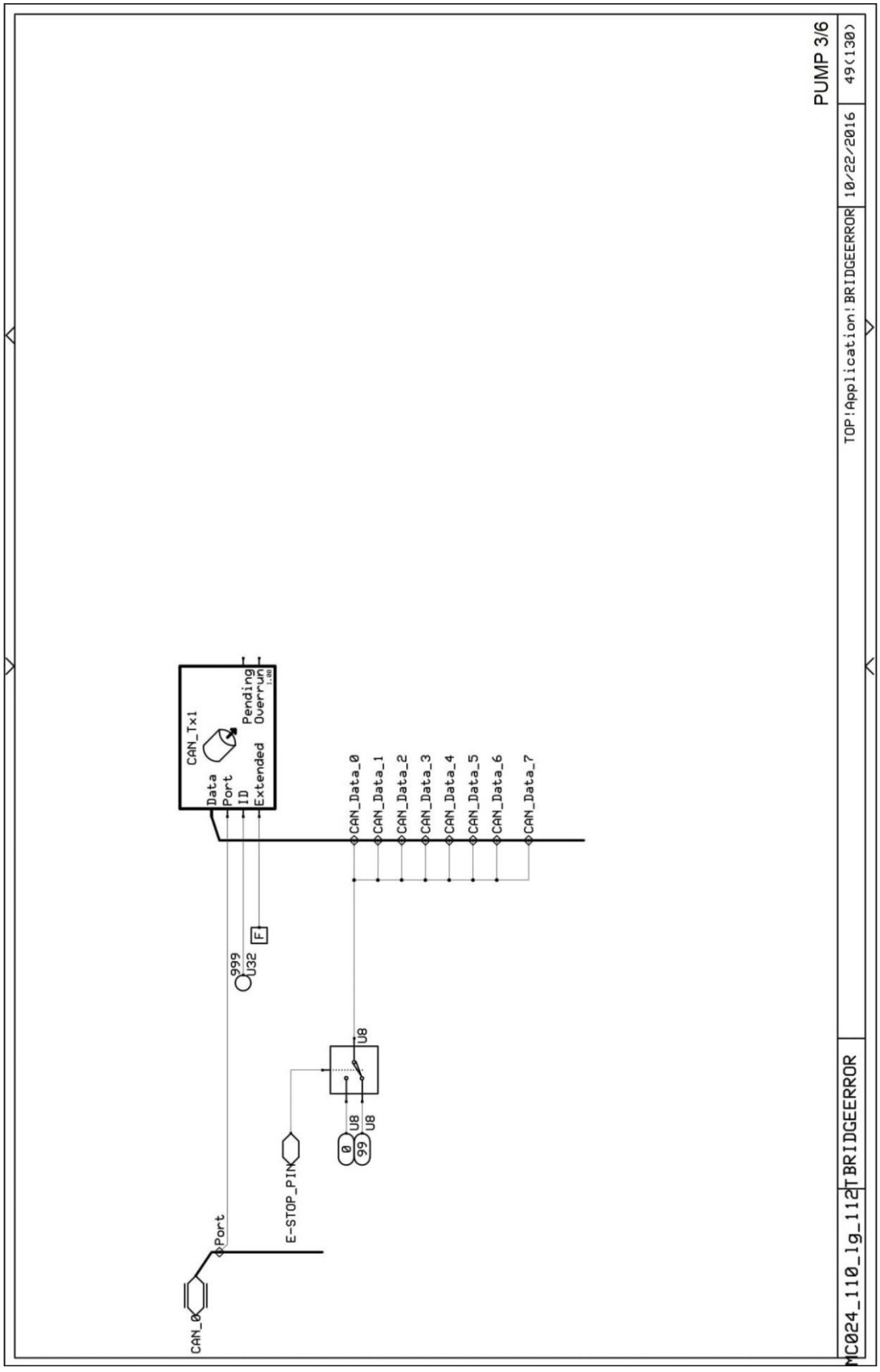

ENGINE 2/4
TOP!Application 10/22/2016 2< 76 >
MC24_10_MC24_12 Application
Inputs
CAN_0
Port
CAN_0
C1p07_DigIn
C1p10_DigIn
CAN_0
C2p12_Status
DigIn
DigIn
Port
FeedBackValue
CHECK POINT
2AMP
EngineCAN
TSC1
ENG SPEED
ENGINE CAN DECODER
CAN 0
START
RUN
Hydraulic_CCO
Hydraulic CCO
PGN 0
Port
Source
Destination
Enable
Data
Status
OR
Pend
TSC1_Tx1_3.0b
EngOvrRdCtrlMode
EngReqSpdCtrlCnd
OvrRdCntlModePri
EngReqSpeedLimit
EngReqTorqLimit
TSC1CtrlPurpose
EngReqTorq_HiRes
208
0
1
3
0
0
0
0
BOOL
T
F
U16
1000
1000
3401
BOOL
F
3
UNIT
T1S
C2p11_MFOut
DigOut
C2p09_MFOut
DigOut
C2p10_MFOut
DigOut
C2p12_MFOut
DigOut
Outputs

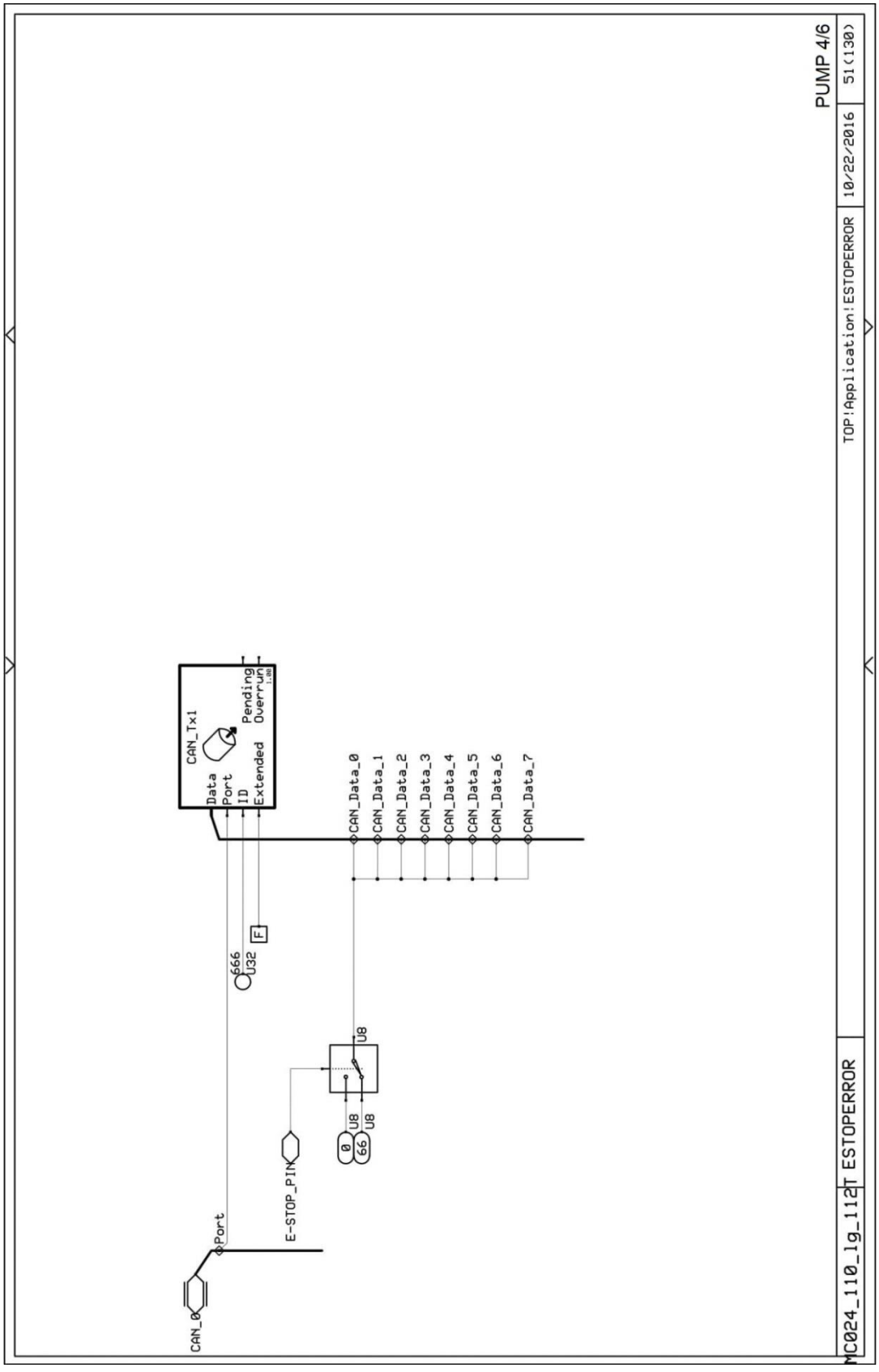

ENGINE 3/4
TOP!Application!Hydraulic_CC00/22/2016
51< 76 >
MC24_10_MC24_12
Hydraulic_CCO
CAN_Rx1
Overrun
Length
Act ID
Rx
Data
Port
Mask
ID
Extended
F
333
U32
333
U32
can_port
CAN_Data_7
CHECK POINT
AD7
BOOL
0
1
F
T
CHECK POINT
ACCO
CCO_out

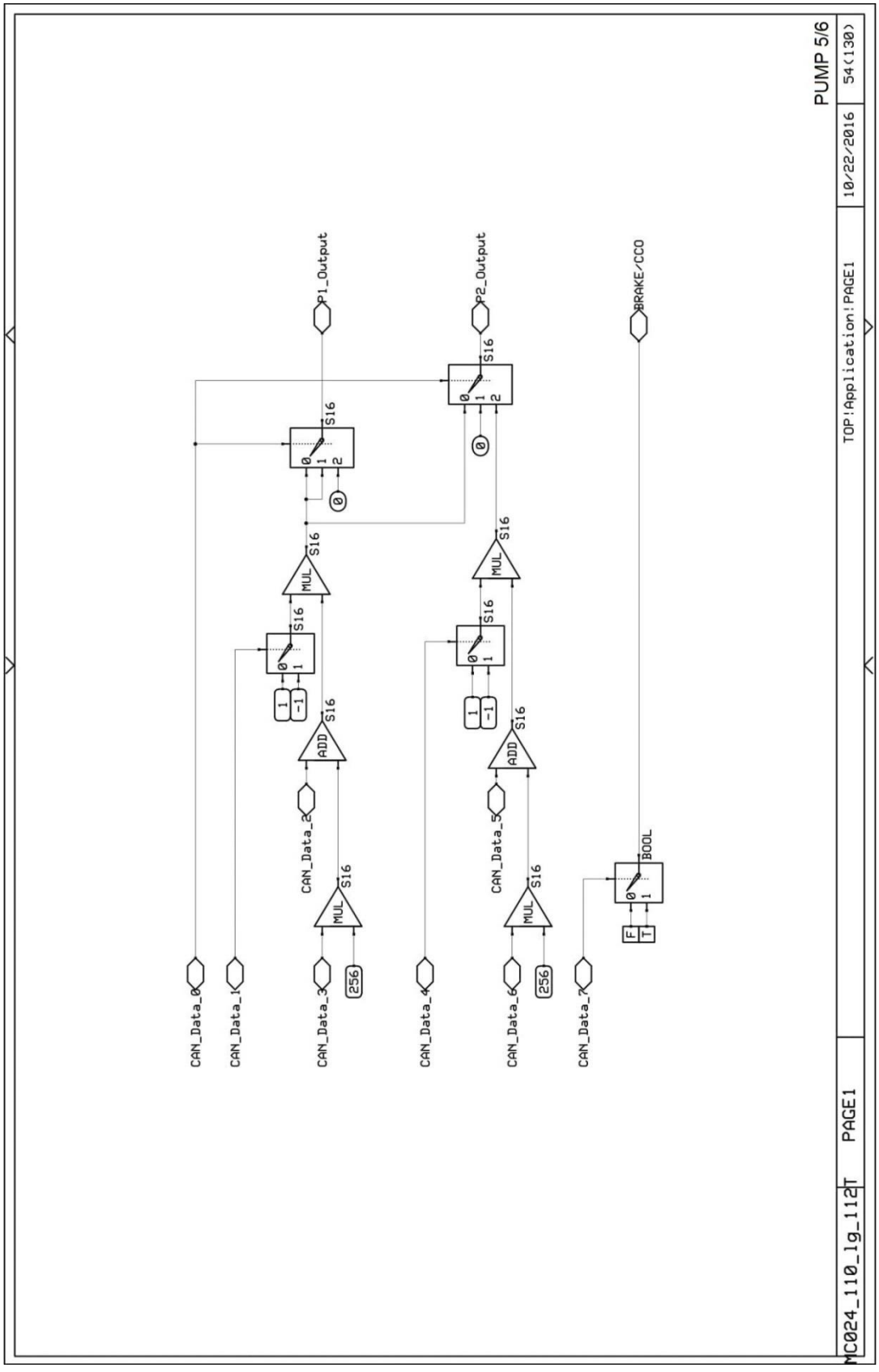

ENGINE 4/4
MC24_10_MC24_12
EngineCAN
TOP!Application!EngineCAN
10/22/2016
49( 76 )
CAN_0
Port
104
U8
CAN_Rx1
Overrun
Length
Act ID
Rx
Data
Port
Mask
ID
Extended
104
U8
F
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
Run
Start
Eng_Spd
TSC1_Enable
BOOL
BOOL
ADD
S16
MUL
S16
256
0
1
F
T

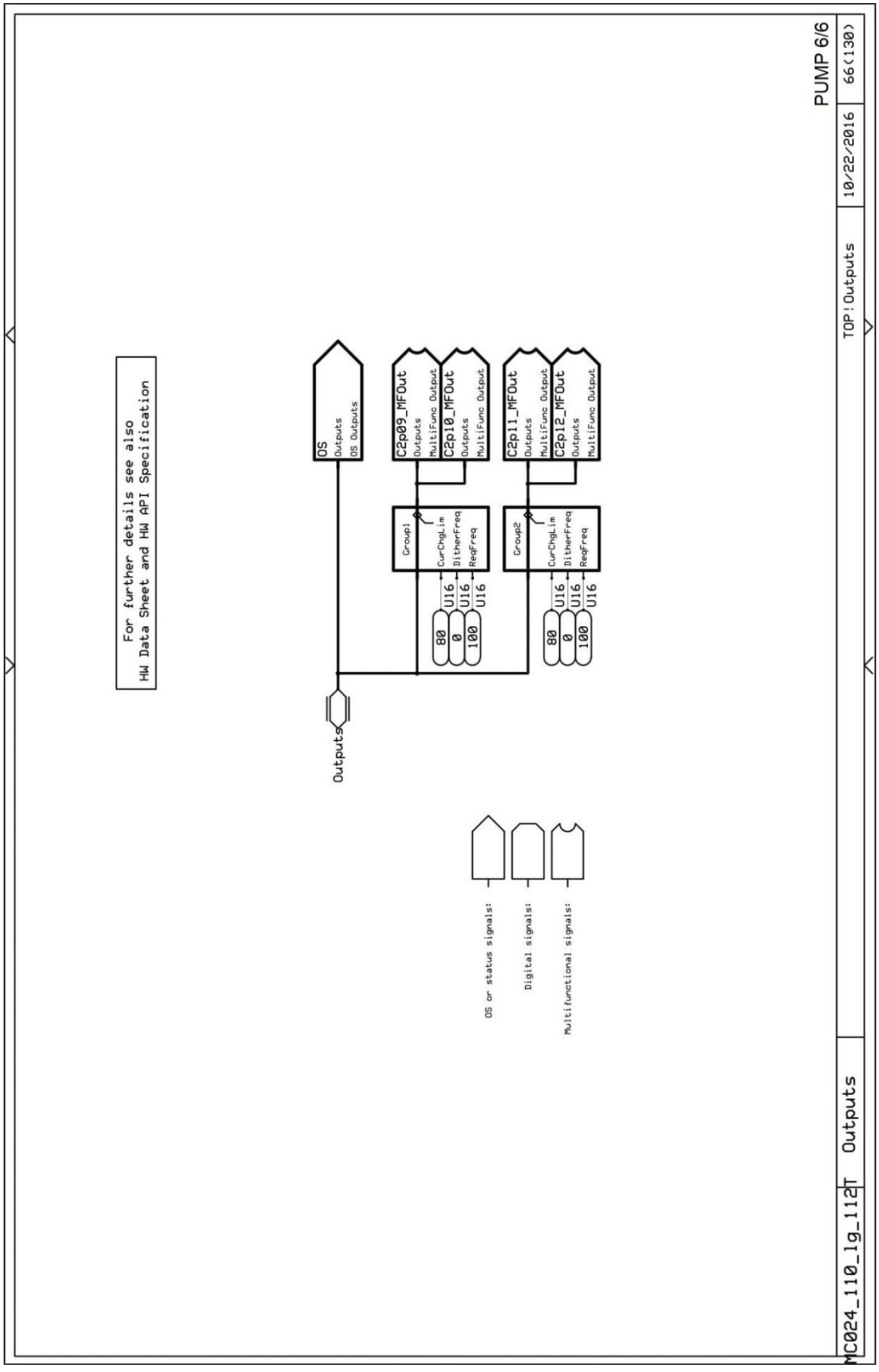

PUMP 1/6
MC024_110_lg_112T
TOP
TOP
10/22/2016
1(130)
Outputs
CAN_0
IO_Bus
Application
MC024-110_lg
MC024-112
Outputs
Inputs
CAN_0
IO_Bus
Inputs
CAN_0
Unit_Config
Mode
2
U8

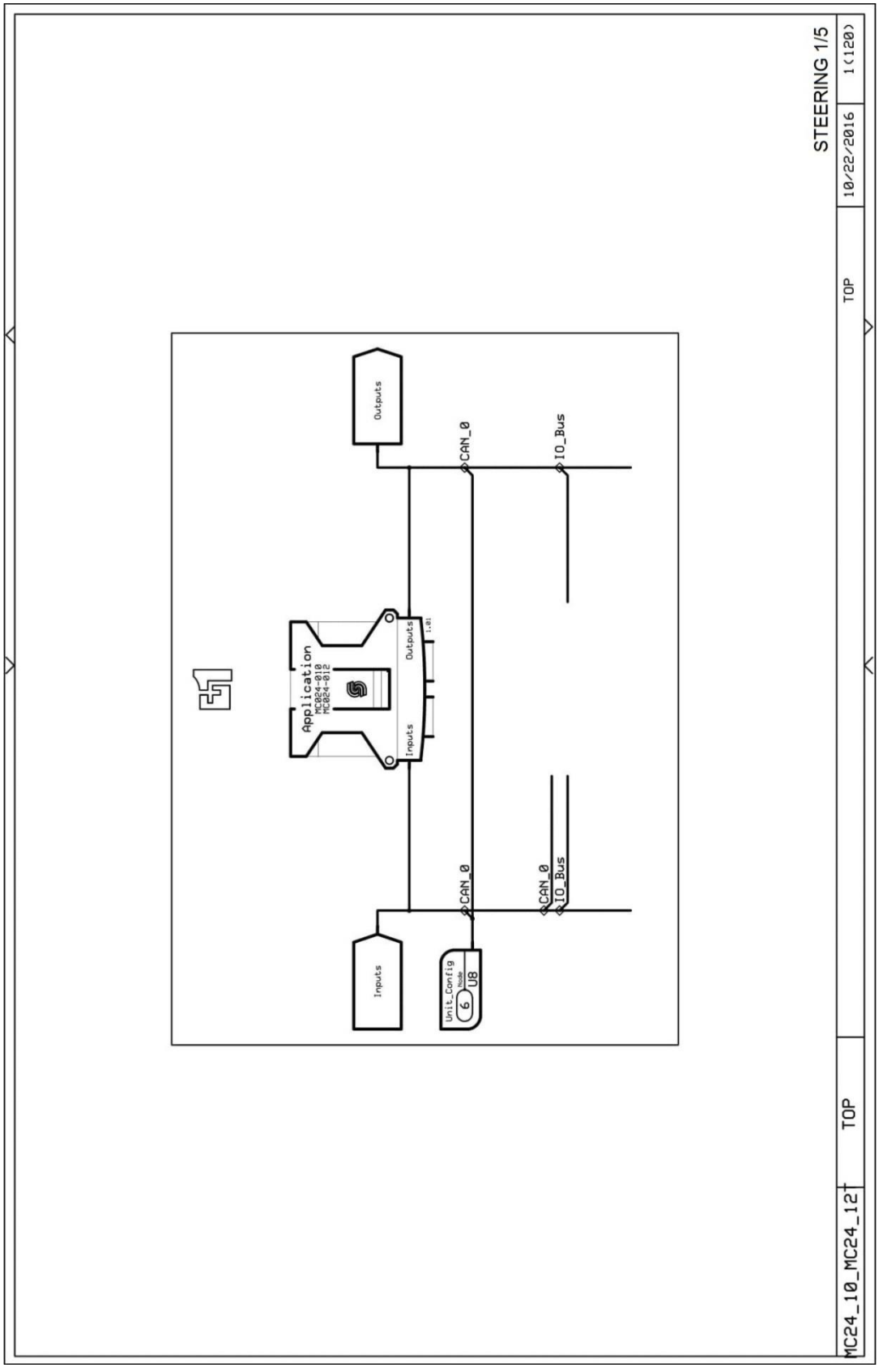

Pump CAN Controller
BRAKE/CCO CONTROL OUTPUT
PAGE1
CAN D0
CAN D1
CAN D2
CAN D3
CAN D4
CAN D5
CAN D6
CAN D7
Pump1
Pump2
BRAKE
CAN_Rx1
Overrun
Length
Act ID
Rx
Data
Port
Mask
ID
Extended
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
CAN_Data_5
CAN_Data_6
CAN_Data_7
H1_12V_CTRL_1
H1_12V_CTRL_2
Pwr
Fdbk_A
Fdbk_B
Setup
Enable
CmdIn
Status
Fault
Diag
Out_A
Out_B
C2p09_MFOut
C2p10_MFOut
C2p11_MFOut
C2p12_MFOut
BRIDGEERROR
ESTOPERROR
E-STOP
ERROR
DigIn
CAN_0
Port
Inputs
Outputs
C2p02_MFIn
C1p10_DigIn
C1p02_Pwr
C2p09_Status
C2p10_Status
C2p11_Status
C2p12_Status
PUMP 2/6
TOP:Application
10/22/2016
2(130)
MC024_110_lg_112
Application

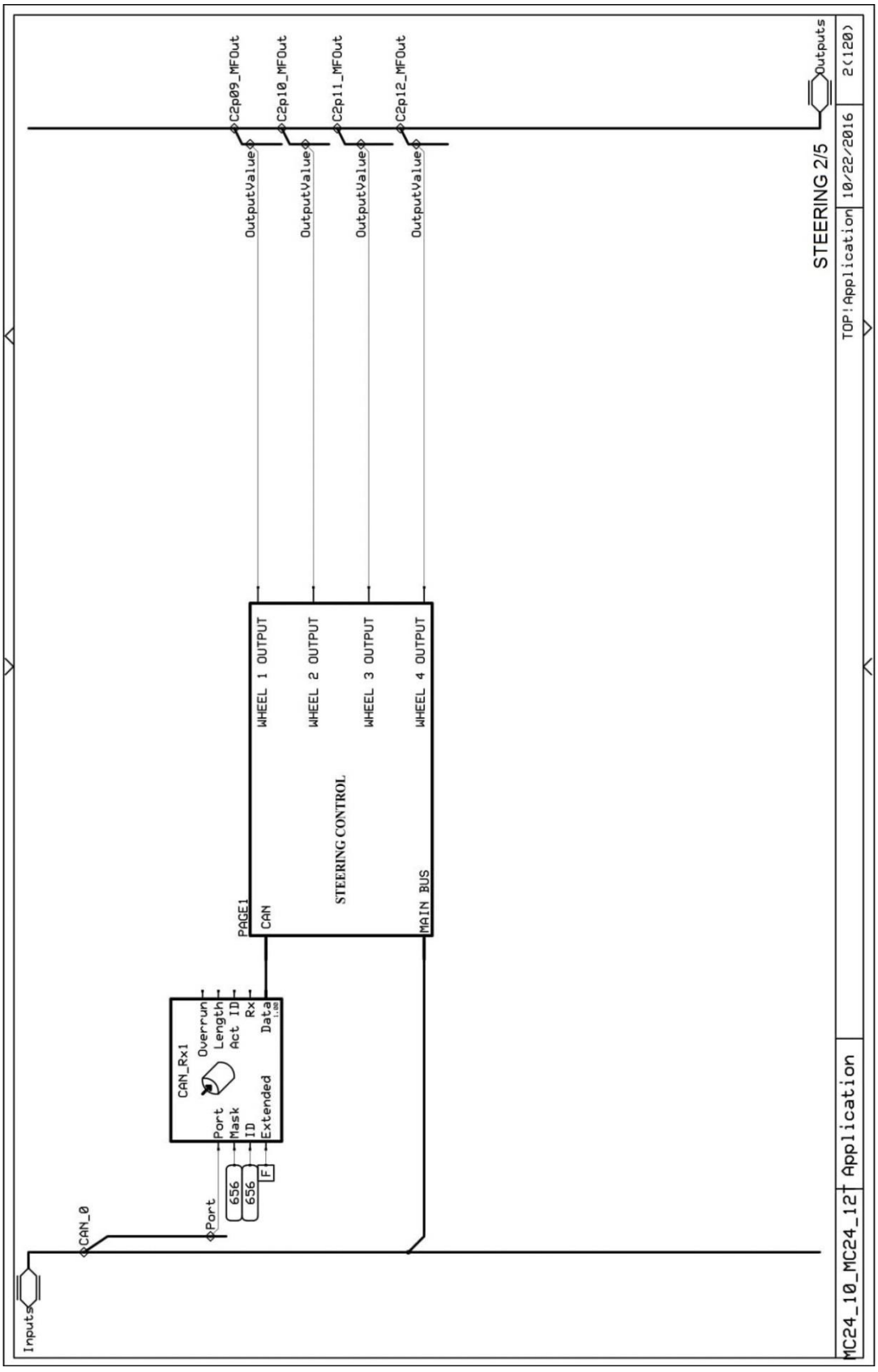

PUMP 3/6
TOP!Application!BRIDGEERROR
10/22/2016
49(130)
MC024_110_lg_112
BRIDGEERROR
CAN_Tx1
Data
Port
ID
Extended
Pending
Overrun
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
CAN_Data_5
CAN_Data_6
CAN_Data_7
999
U32
F
U8
0
99
U8
U8
E-STOP_PIN
Port
CAN_0

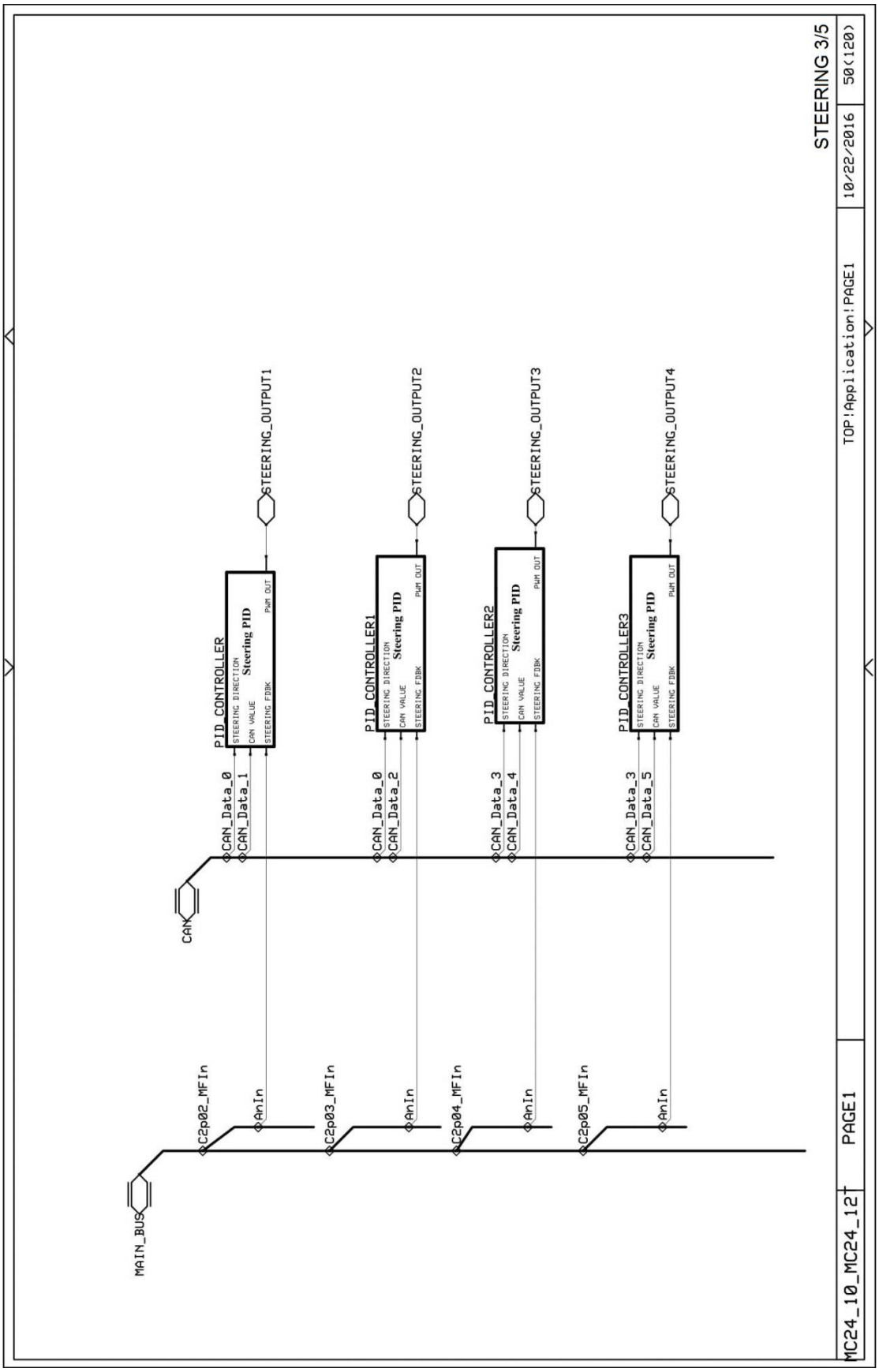

PUMP 4/6
TOP!Application!ESTOPERROR
10/22/2016
51(130)
MC024_110_lg_112
T ESTOPERROR
CAN_Tx1
Data
Port
ID
Extended
Pending
Overrun
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
CAN_Data_5
CAN_Data_6
CAN_Data_7
666
U32
F
U8
U8
U8
0
66
E-STOP_PIN
Port
CAN_0

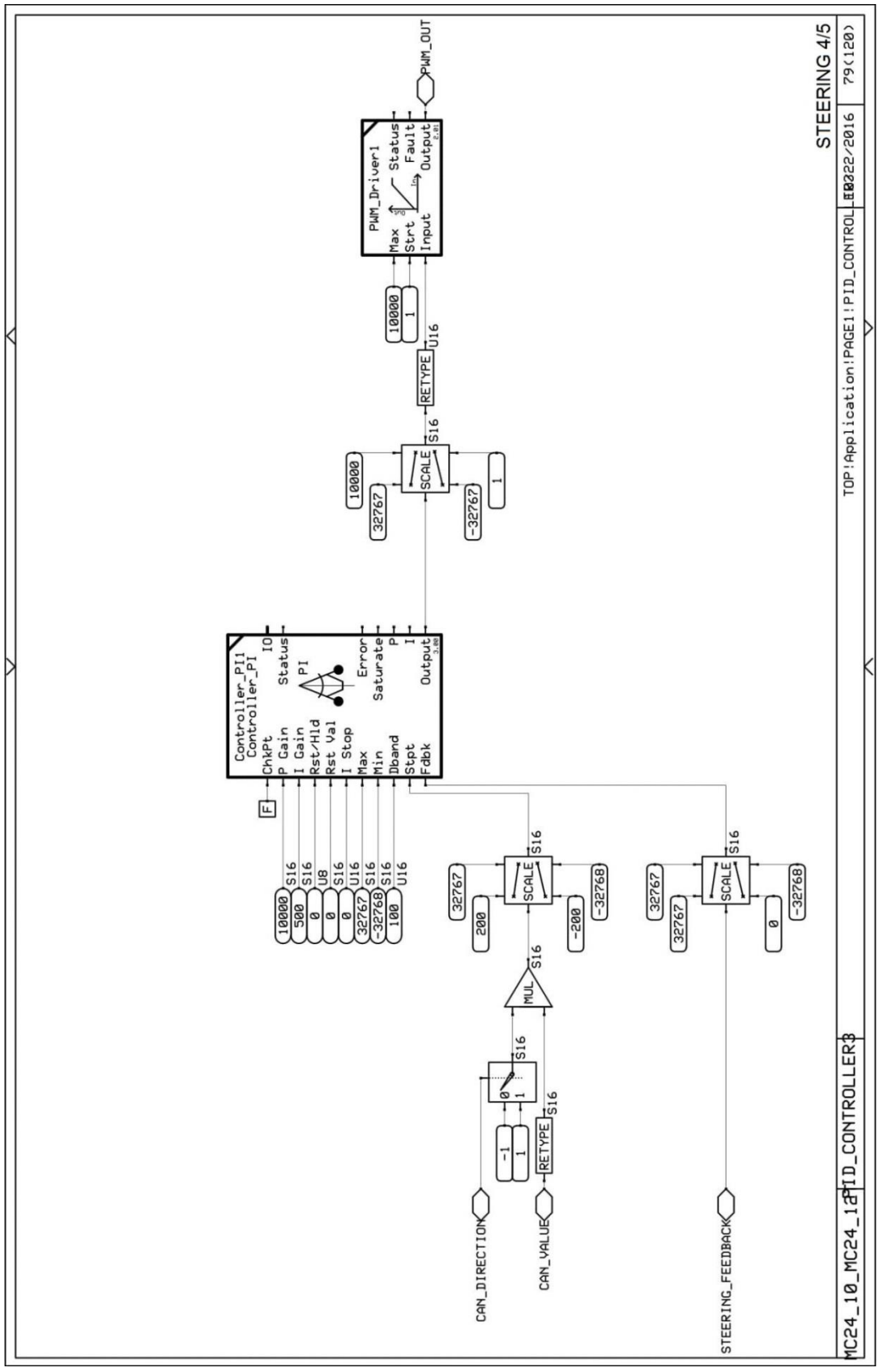

PUMP 5/6
MC024_110_1g_112T
PAGE1
TOP!Application!PAGE1
10/22/2016
54(130)
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
CAN_Data_5
CAN_Data_6
CAN_Data_7
P1_Output
P2_Output
BRAKE/CCO
MUL
ADD
S16
BOOL
256
F
T

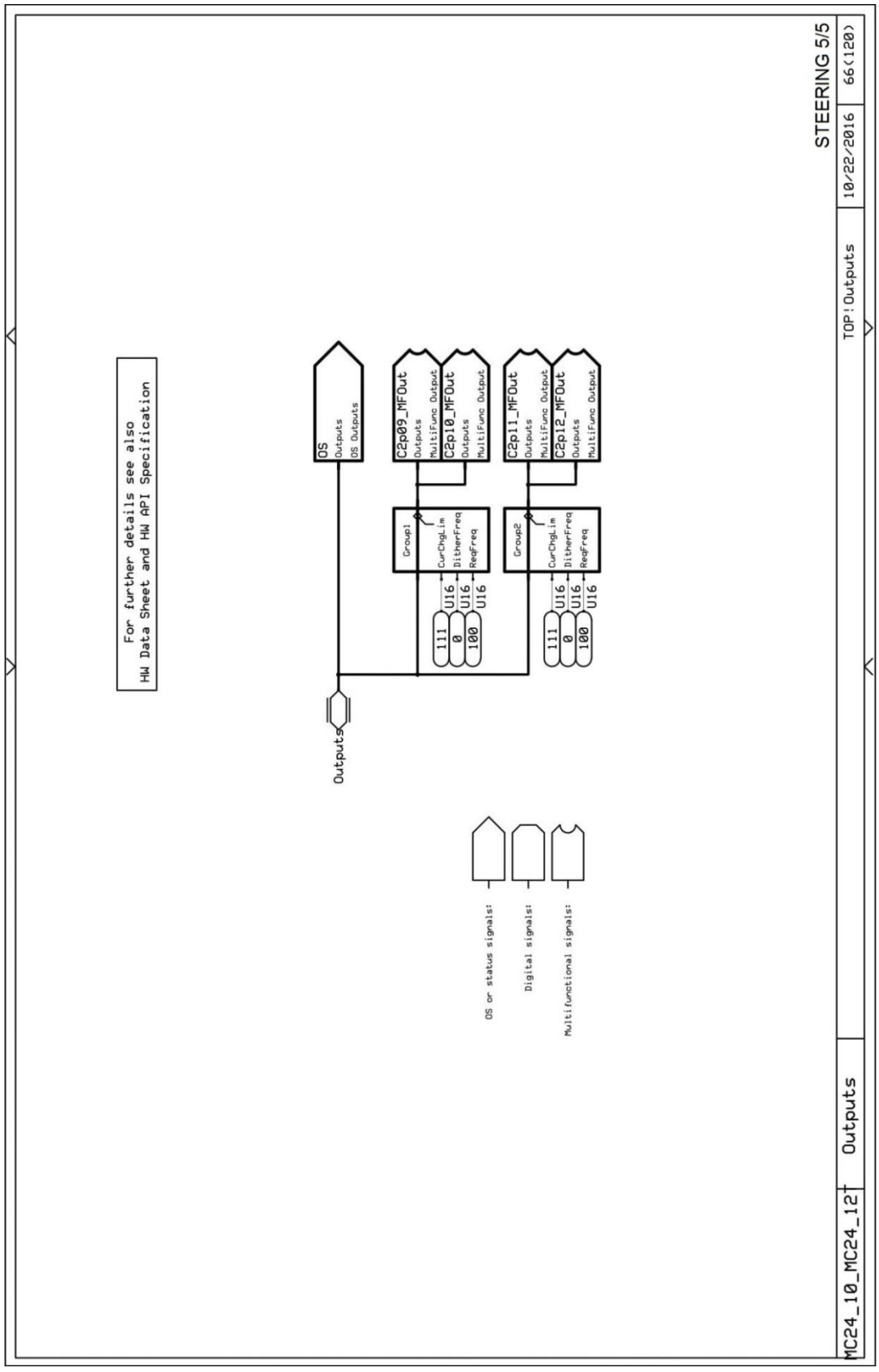

PUMP 6/6
MC024_110_1g_112T Outputs
TOP!Outputs
10/22/2016
66(130)
For further details see also
HW Data Sheet and HW API Specification
OS
Outputs
OS Outputs
C2p09_MFOut
Outputs
MultiFunc Output
C2p10_MFOut
Outputs
MultiFunc Output
C2p11_MFOut
Outputs
MultiFunc Output
C2p12_MFOut
Outputs
MultiFunc Output
Group1
Group2
CurChgLim
DitherFreq
ReqFreq
U16
80
0
100
Outputs
OS or status signals:
Digital signals:
Multifunctional signals:

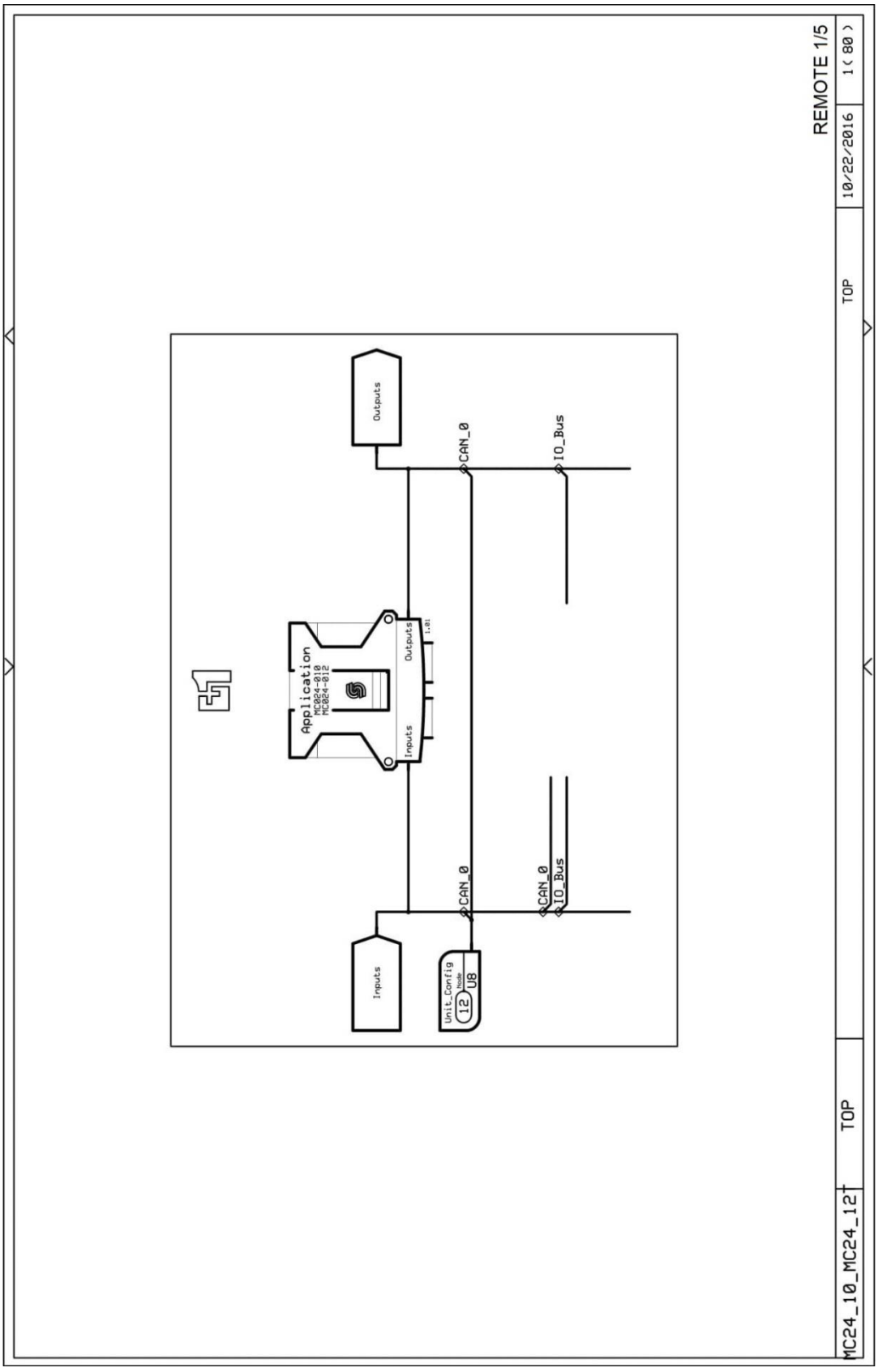

STEERING 1/5
10/22/2016
1(120)
TOP
MC24_10_MC24_12
TOP
Application
MC024-010
MC024-012
Outputs
Inputs
Outputs
Inputs
CAN_0
IO_Bus
CAN_0
IO_Bus
CAN_0
IO_Bus
Unit_Config
Mode
6
U8

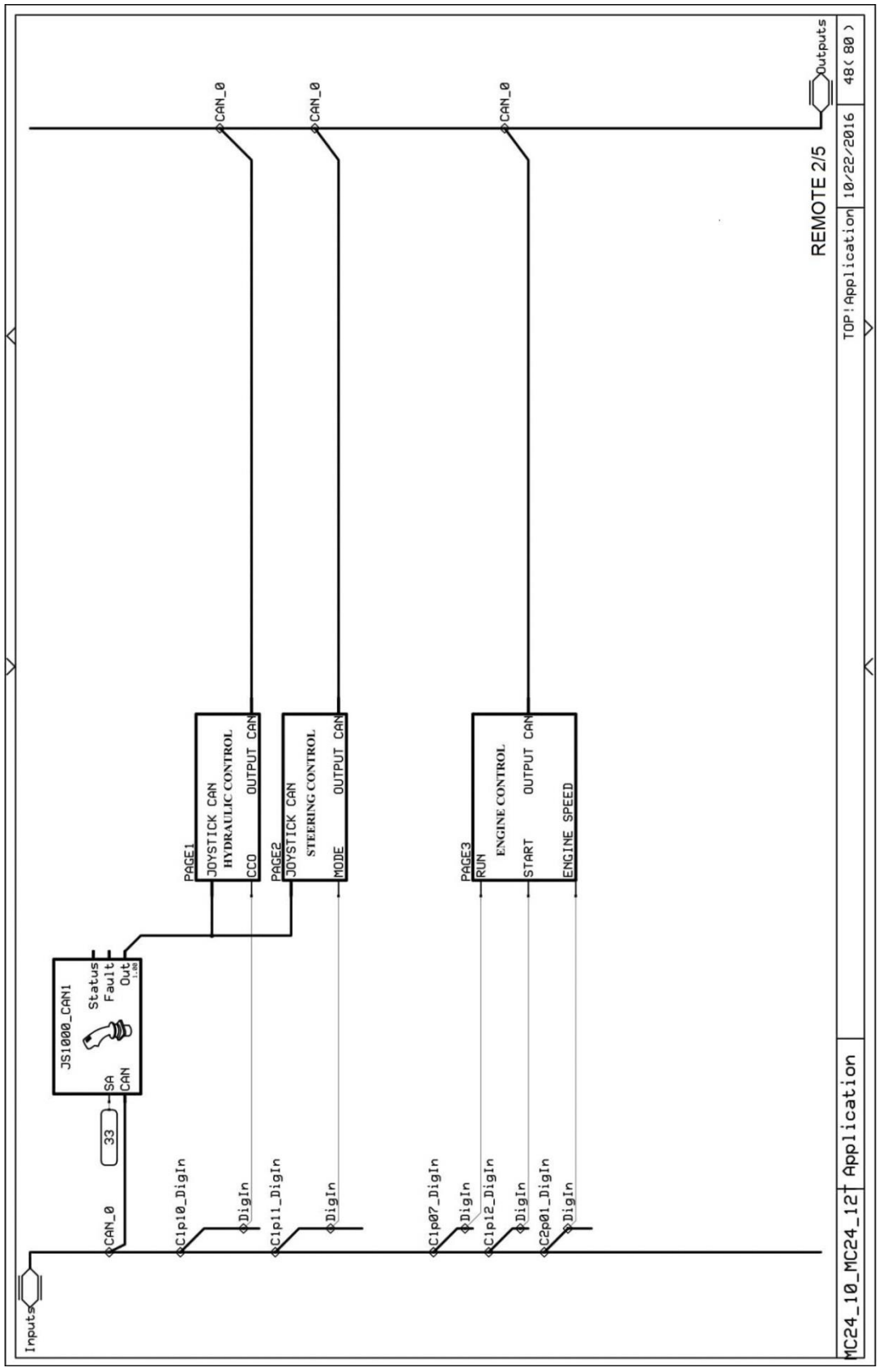

Inputs
CAN_0
Port
656
656
F
Port
Mask
ID
Extended
CAN_Rx1
Overrun
Length
Act ID
Rx
Data
PAGE1
CAN
STEERING CONTROL
WHEEL 1 OUTPUT
WHEEL 2 OUTPUT
WHEEL 3 OUTPUT
WHEEL 4 OUTPUT
MAIN BUS
OutputValue
OutputValue
OutputValue
OutputValue
C2p09_MFOut
C2p10_MFOut
C2p11_MFOut
C2p12_MFOut
Outputs
STEERING 2/5
TOP!Application
10/22/2016
2(120)
MC24_10_MC24_12†
Application

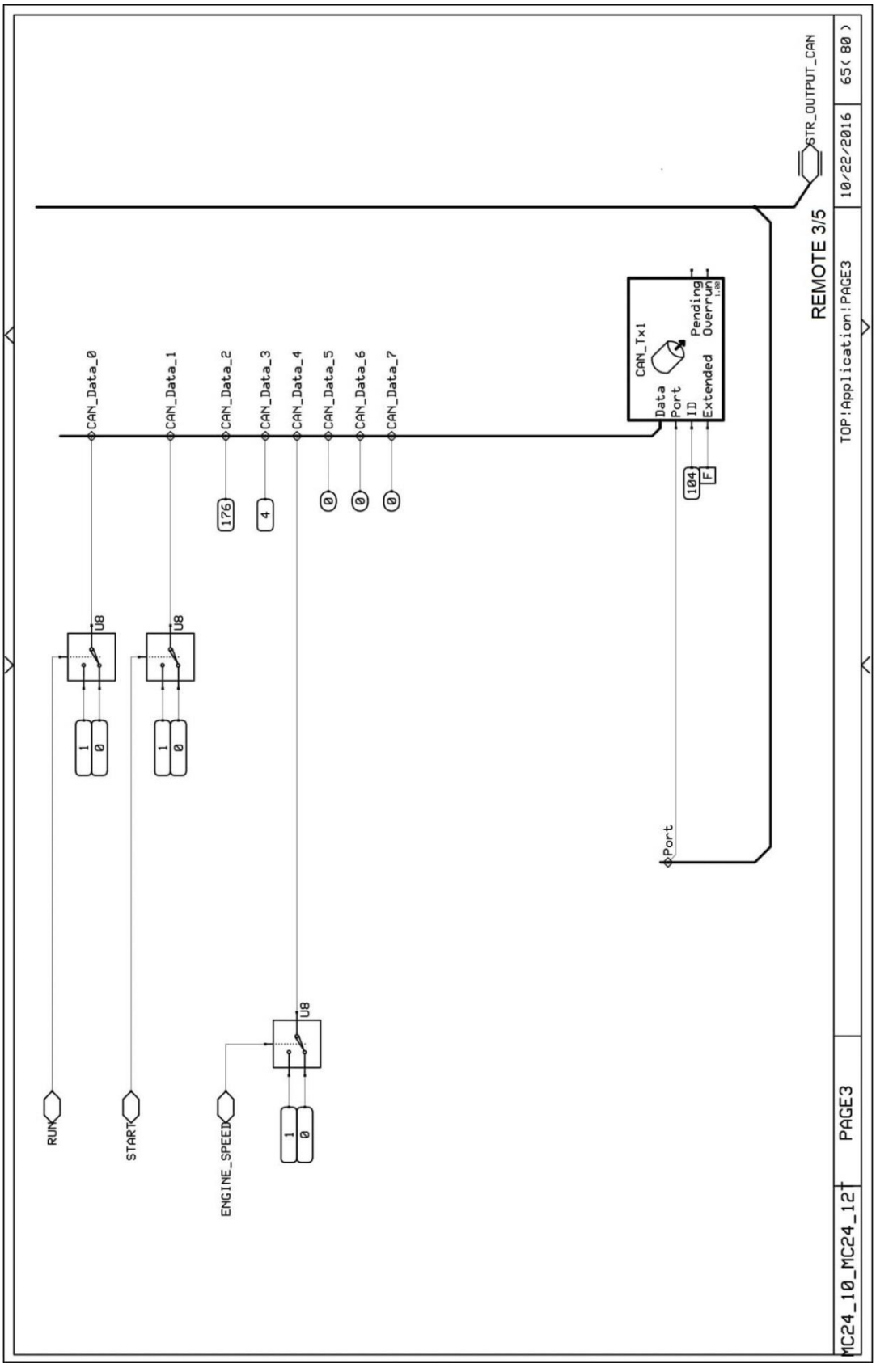

STEERING 3/5
MC24_10_MC24_12
PAGE1
TOP!Application!PAGE1
10/22/2016
50(120)
MAIN_BUS
C2p02_MFIn
C2p03_MFIn
C2p04_MFIn
C2p05_MFIn
AnIn
CAN
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
CAN_Data_5
PID_CONTROLLER
PID_CONTROLLER1
PID_CONTROLLER2
PID_CONTROLLER3
STEERING DIRECTION
CAN VALUE
STEERING FDBK
Steering PID
PWM OUT
STEERING_OUTPUT1
STEERING_OUTPUT2
STEERING_OUTPUT3
STEERING_OUTPUT4

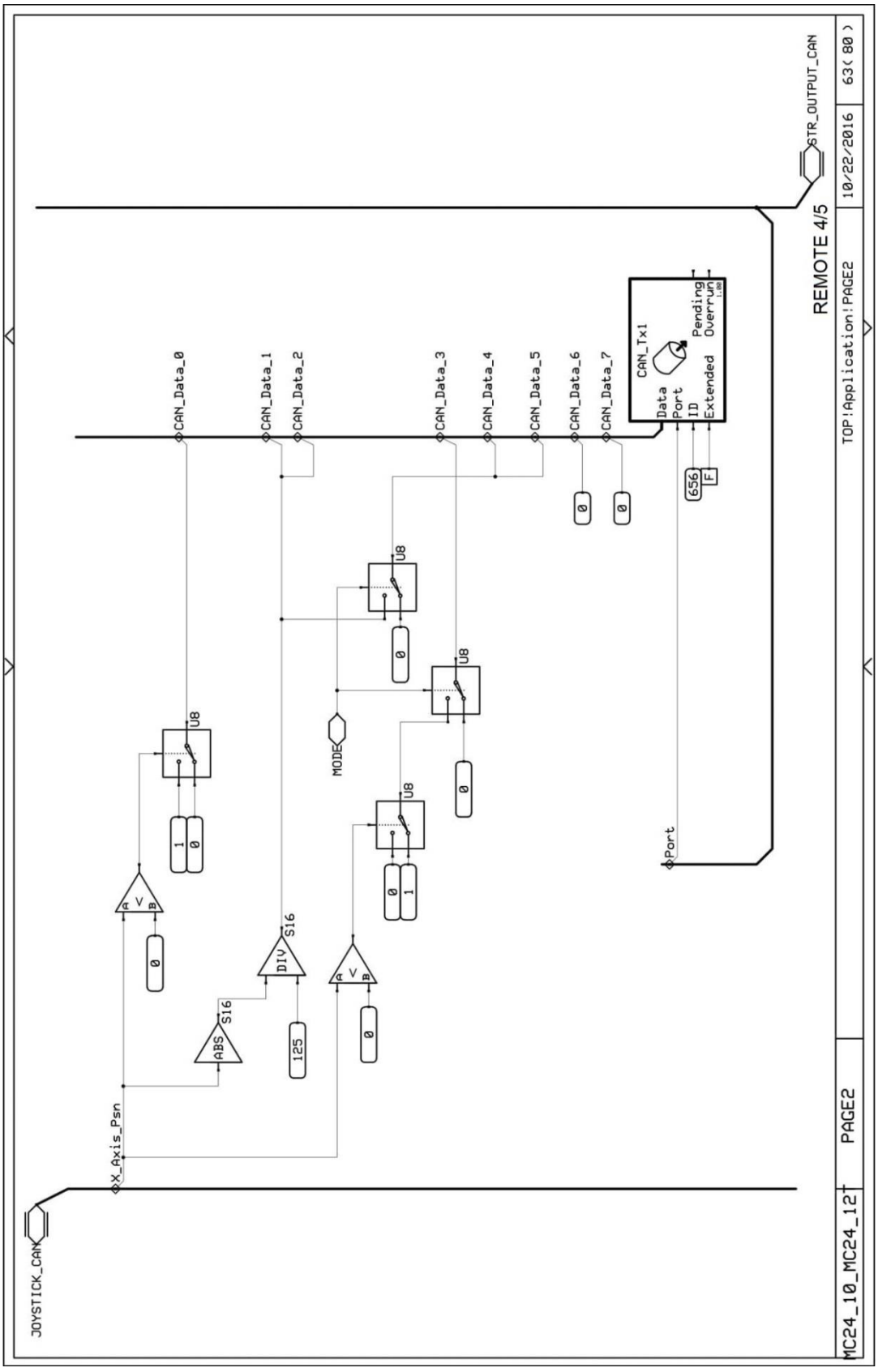

STEERING 4/5
TOP!Application!PAGE1!PID_CONTROL
MC24_10_MC24_12
PID_CONTROLLER3
PWM_OUT
PWM_Driver1
Status
Fault
Output
Max
Strt
Input
10000
1
RETYPE
U16
S16
SCALE
32767
-32767
Controller_PI1
Controller_PI
ChkPt
P Gain
I Gain
Rst/Hld
Rst Val
I Stop
Max
Min
Dband
Stpt
Fdbk
Error
Saturate
P
I
Output
IO
Status
PI
F
500
0
U8
-32768
100
200
-200
MUL
-1
CAN_DIRECTION
CAN_VALUE
STEERING_FEEDBACK

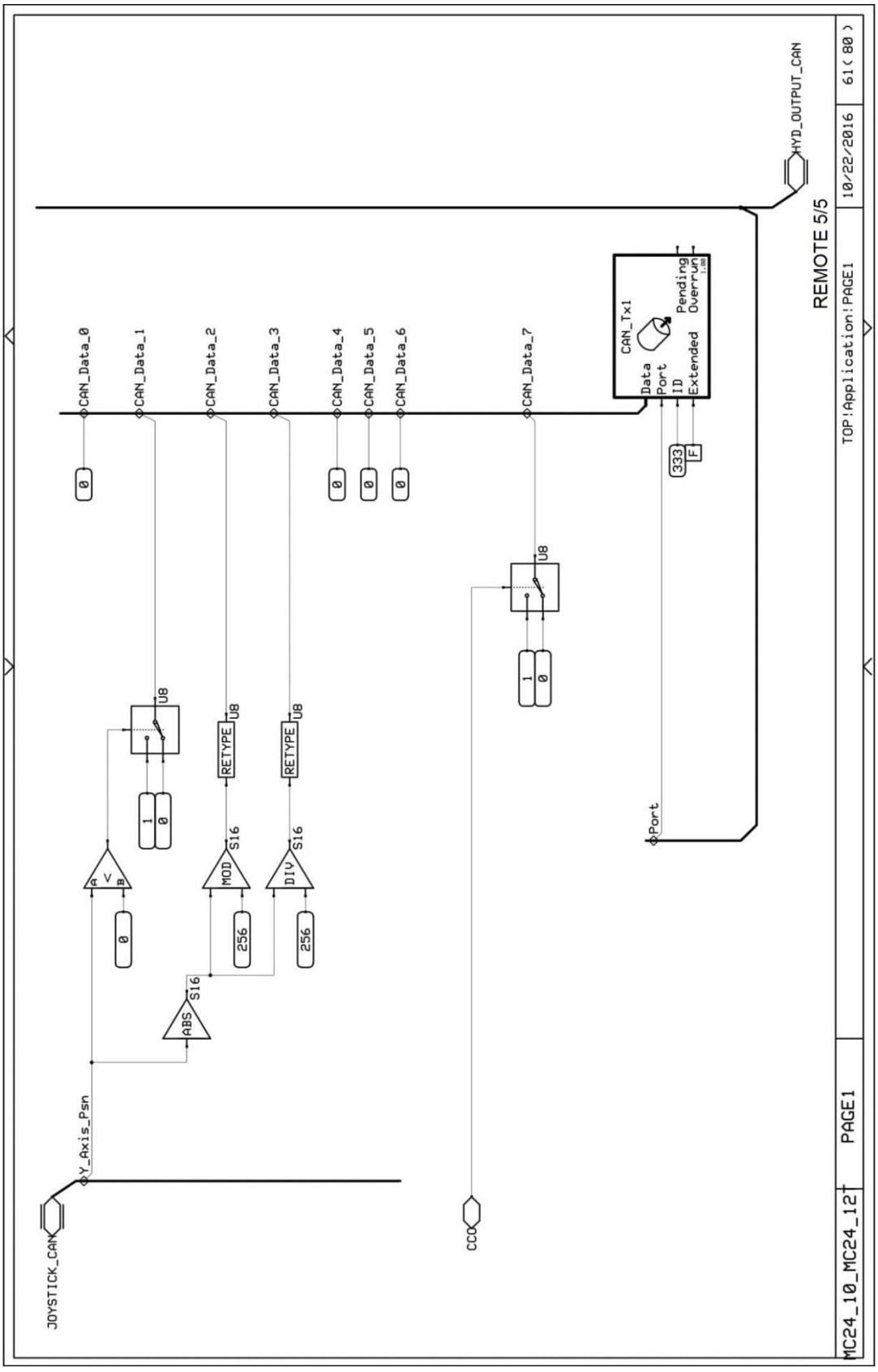

STEERING 5/5
MC24_10_MC24_12†
Outputs
TOP!Outputs
10/22/2016
66(120)
For further details see also
HW Data Sheet and HW API Specification
OS
Outputs
OS Outputs
C2p09_MFOut
C2p10_MFOut
C2p11_MFOut
C2p12_MFOut
Outputs
MultiFunc Output
Group1
Group2
CurChgLim
DitherFreq
ReqFreq
U16
111
0
100
Outputs
OS or status signals:
Digital signals:
Multifunctional signals:

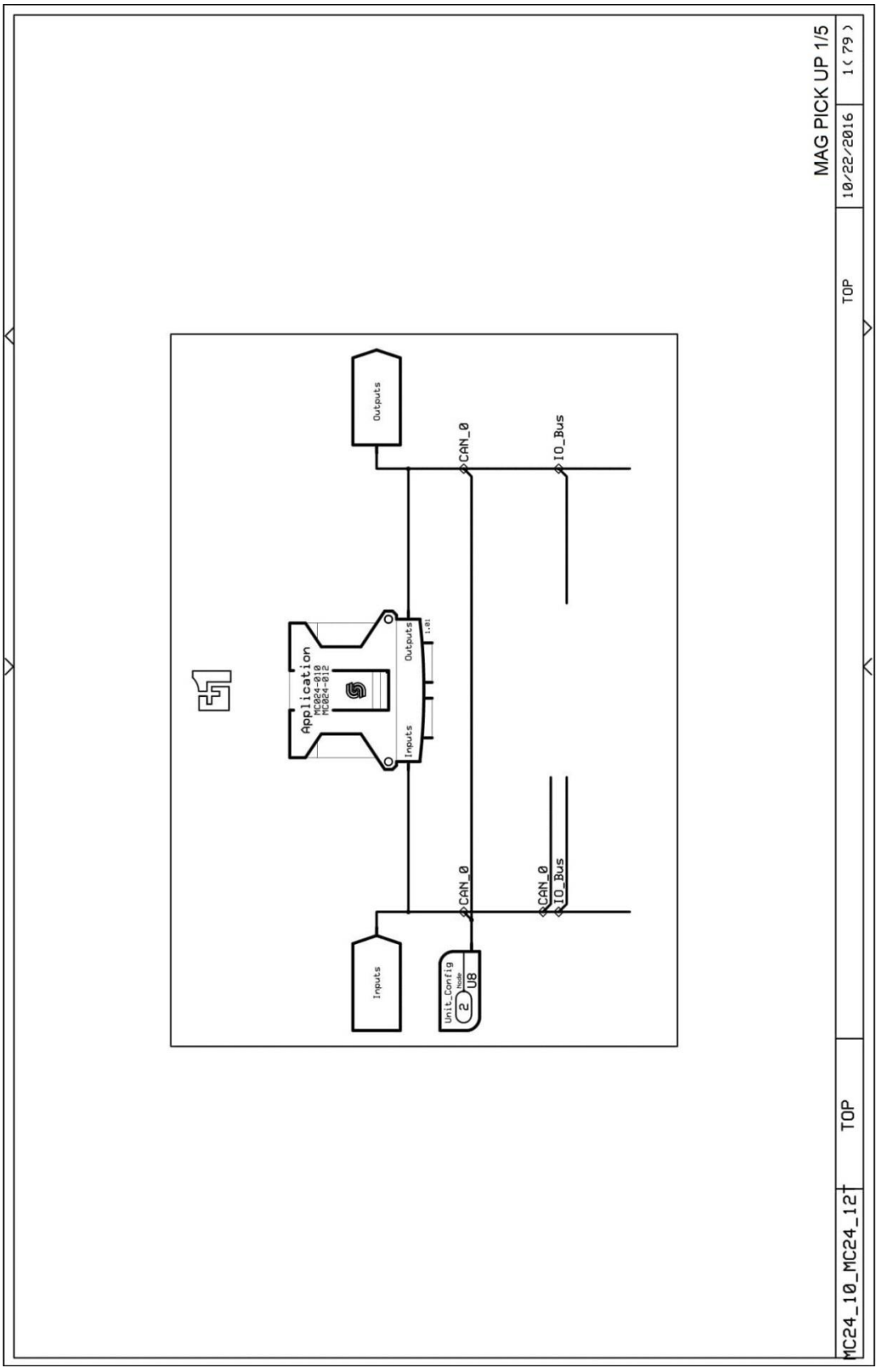

REMOTE 1/5
MC24_10_MC24_12
TOP
TOP
10/22/2016
1 ( 80 )
Application
MC024-010
MC024-012
Inputs
Outputs
I.01
Outputs
CAN_0
IO_Bus
Inputs
CAN_0
Unit_Config
Mode
I2
U8
CAN_0
IO_Bus

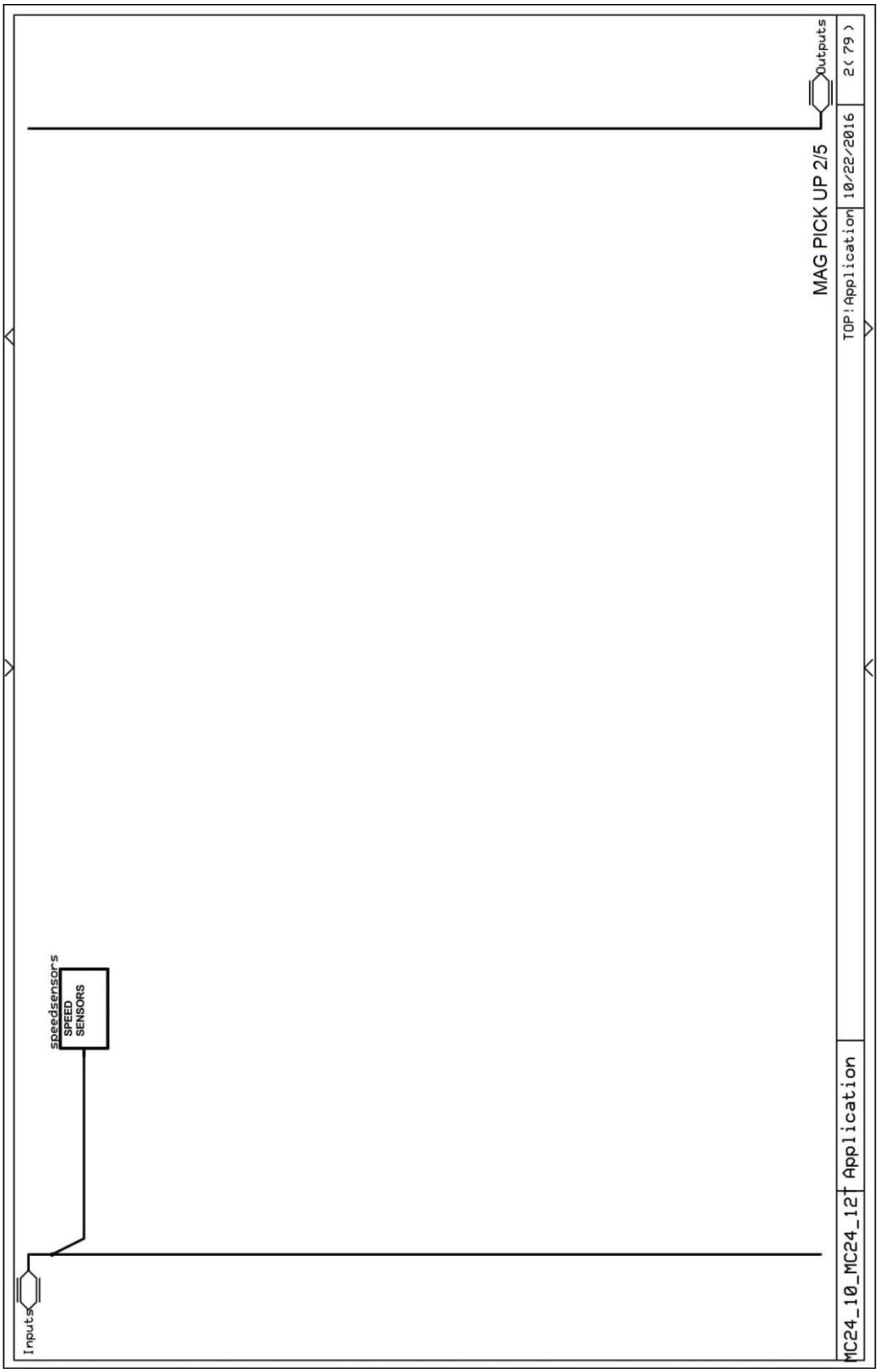

Inputs
CAN_0
JS1000_CAN1
Status
Fault
Out
SA
CAN
33
C1p10_DigIn
DigIn
C1p11_DigIn
DigIn
C1p07_DigIn
DigIn
C1p12_DigIn
DigIn
C2p01_DigIn
DigIn
PAGE1
JOYSTICK CAN
HYDRAULIC CONTROL
CCO
OUTPUT CAN
PAGE2
JOYSTICK CAN
STEERING CONTROL
MODE
OUTPUT CAN
PAGE3
RUN
ENGINE CONTROL
START
OUTPUT CAN
ENGINE SPEED
CAN_0
CAN_0
CAN_0
Outputs
REMOTE 2/5
MC24_10_MC24_12 Application
TOP!Application 10/22/2016
48( 80 )

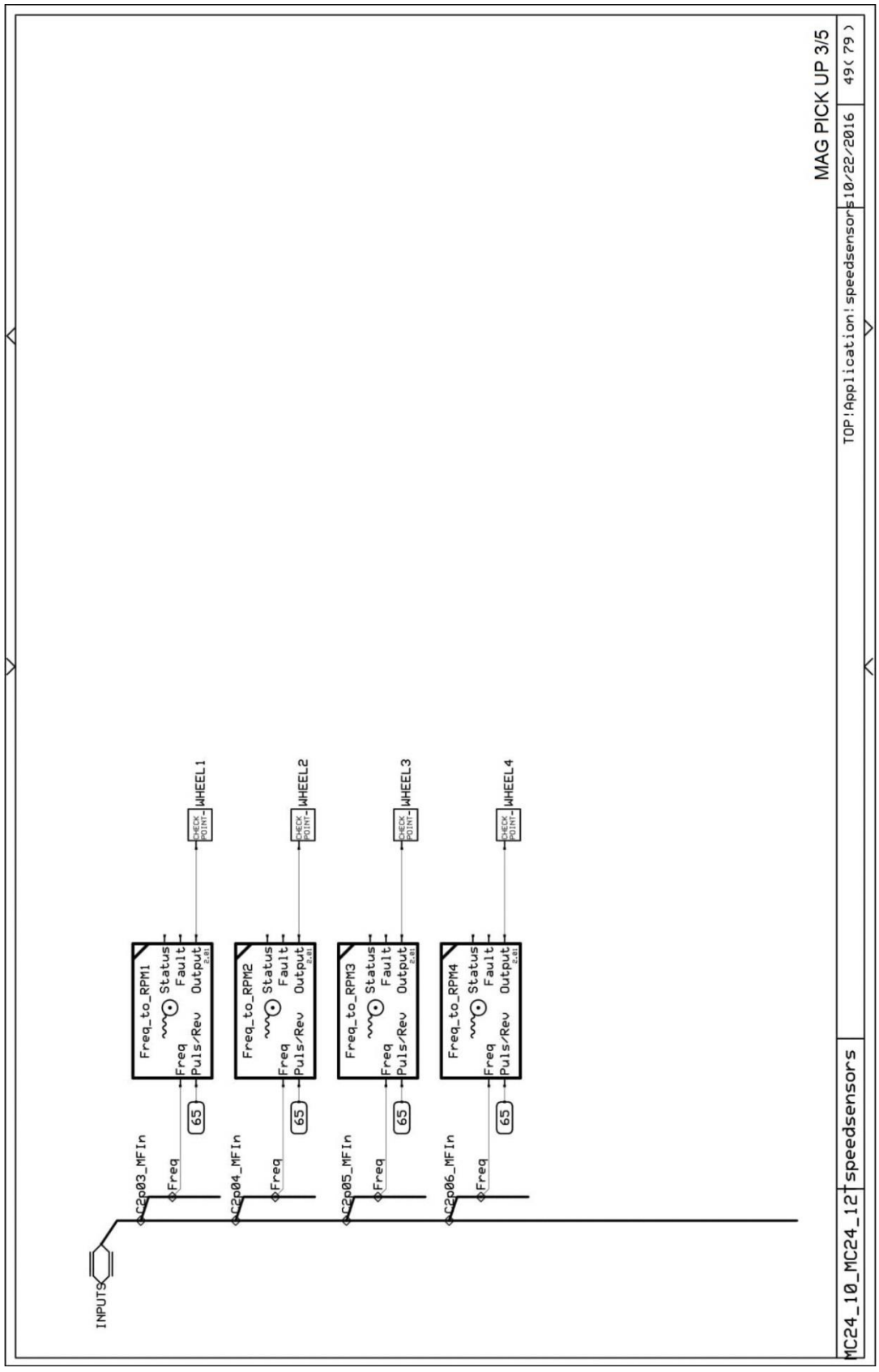

REMOTE 3/5
MC24_10_MC24_12
PAGE3
TOP!Application!PAGE3
10/22/2016
65( 80 )
STR_OUTPUT_CAN
CAN_Tx1
Data
Port
ID
Extended
Pending
Overrun
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
CAN_Data_5
CAN_Data_6
CAN_Data_7
176
4
0
0
0
104
F
U8
U8
U8
RUN
START
ENGINE_SPEED
Port

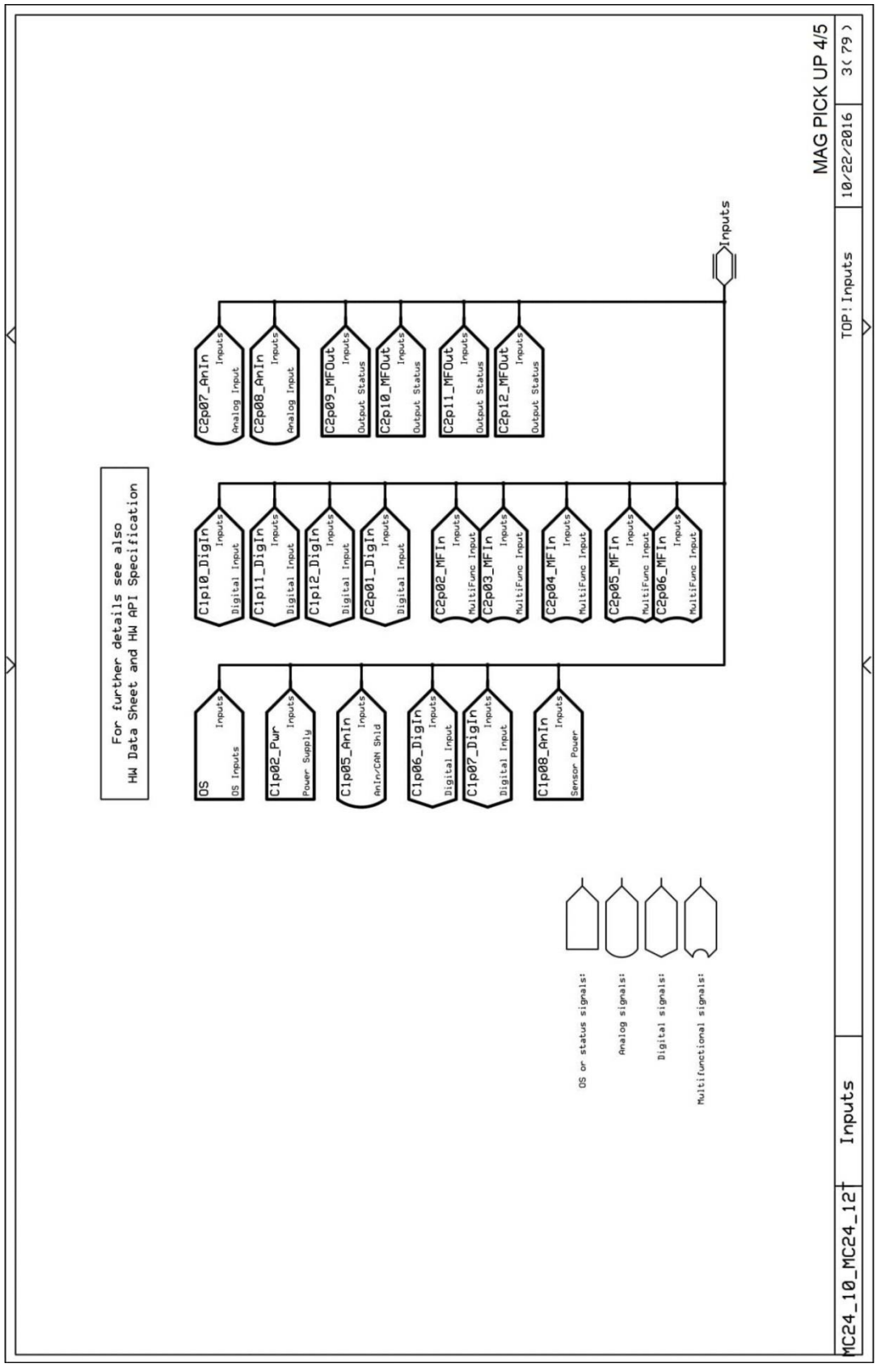

REMOTE 4/5
JOYSTICK_CAN
STR_OUTPUT_CAN
MC24_10_MC24_12
PAGE2
TOP!Application!PAGE2
10/22/2016
63( 80 )
X_Axis_Psn
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
CAN_Data_5
CAN_Data_6
CAN_Data_7
CAN_Tx1
Data
Port
ID
Extended
Pending
Overrun
Port
MODE
U8
S16
ABS
DIV
125
656
F

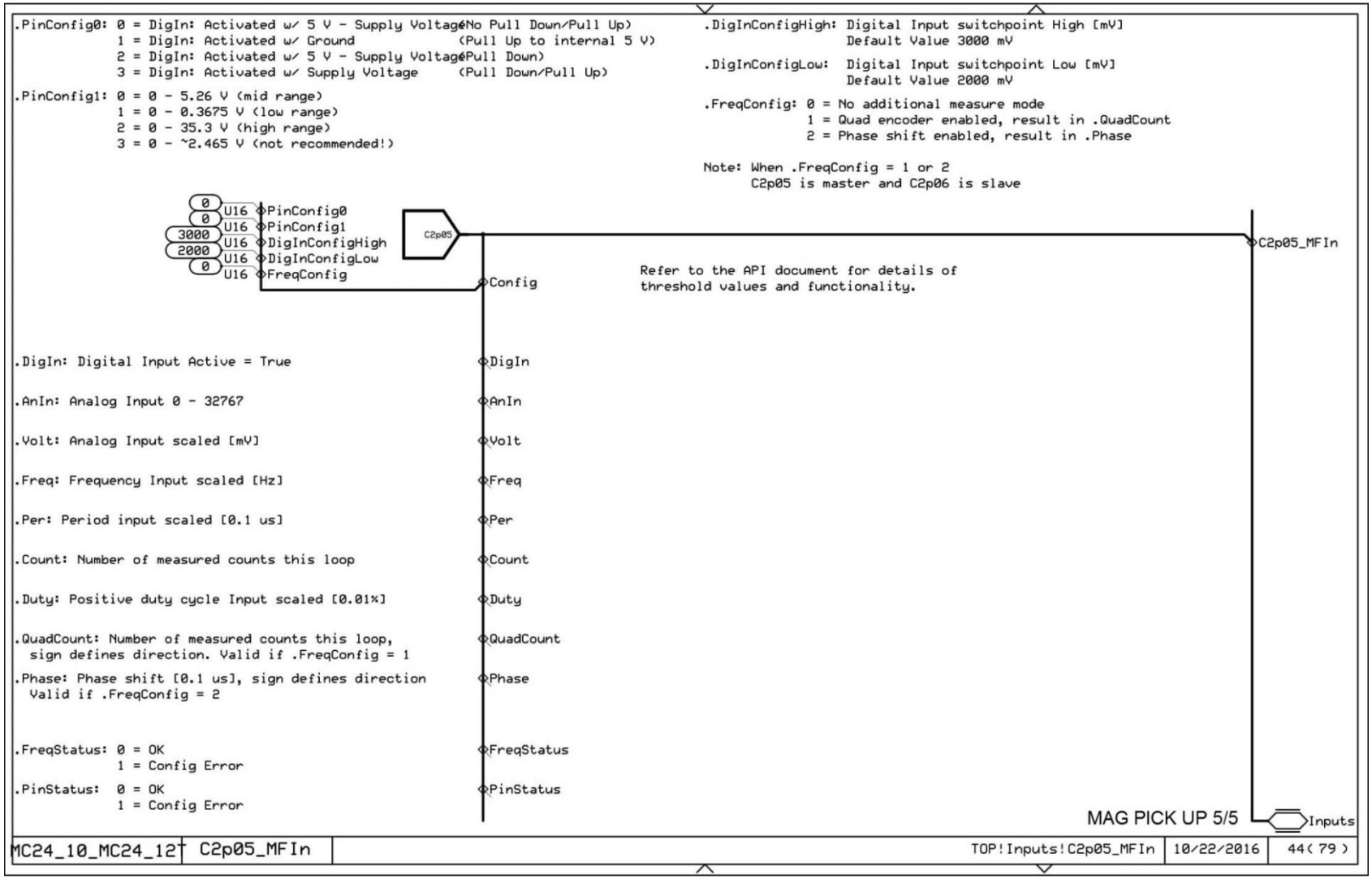

REMOTE 5/5
MC24_10_MC24_12
PAGE1
TOP!Application!PAGE1
10/22/2016
61 ( 80 )
JOYSTICK_CAN
HYD_OUTPUT_CAN
CCO
Y_Axis_Psn
CAN_Data_0
CAN_Data_1
CAN_Data_2
CAN_Data_3
CAN_Data_4
CAN_Data_5
CAN_Data_6
CAN_Data_7
CAN_Tx1
Data
Port
ID
Extended
Pending
Overrun
333
F
Port
U8
RETYPE
S16
MOD
DIV
ABS
A
B
V
0
1
256


MAG PICK UP 1/5
10/22/2016
1 ( 79 )
TOP
MC24_10_MC24_12
TOP
Application
MC024-010
MC024-012
Inputs
Outputs
1.01
Outputs
CAN_0
IO_Bus
Inputs
CAN_0
Unit_Config
Node
2
U8
CAN_0
IO_Bus


Inputs
speedsensors
SPEED
SENSORS
Outputs
MAG PICK UP 2/5
MC24_10_MC24_12
Application
TOP!Application
10/22/2016
2( 79 )


MAG PICK UP 3/5
MC24_10_MC24_12
speedsensors
TOP!Application!speedsensors
10/22/2016
49( 79 )
INPUTS
G2p03_MFIn
G2p04_MFIn
G2p05_MFIn
G2p06_MFIn
Freq
65
Freq_to_RPM1
Freq_to_RPM2
Freq_to_RPM3
Freq_to_RPM4
Status
Fault
Freq
Puls/Rev
Output
CHECK POINT-
WHEEL1
WHEEL2
WHEEL3
WHEEL4


MAG PICK UP 4/5
MC24_10_MC24_12
Inputs
TOP!Inputs
10/22/2016
3( 79 )
For further details see also
HW Data Sheet and HW API Specification
Inputs
OS
OS Inputs
C1p02_Pwr
Power Supply
C1p05_AnIn
AnIn/CAN Shld
C1p06_DigIn
Digital Input
C1p07_DigIn
Digital Input
C1p08_AnIn
Sensor Power
C1p10_DigIn
Digital Input
C1p11_DigIn
Digital Input
C1p12_DigIn
Digital Input
C2p01_DigIn
Digital Input
C2p02_MFIn
MultiFunc Input
C2p03_MFIn
MultiFunc Input
C2p04_MFIn
MultiFunc Input
C2p05_MFIn
MultiFunc Input
C2p06_MFIn
MultiFunc Input
C2p07_AnIn
Analog Input
C2p08_AnIn
Analog Input
C2p09_MFOut
Output Status
C2p10_MFOut
Output Status
C2p11_MFOut
Output Status
C2p12_MFOut
Output Status
OS or status signals:
Analog signals:
Digital signals:
Multifunctional signals:


.PinConfig0: 0 = DigIn: Activated w/ 5 V - Supply Voltage (No Pull Down/Pull Up)
1 = DigIn: Activated w/ Ground (Pull Up to internal 5 V)
2 = DigIn: Activated w/ 5 V - Supply Voltage (Pull Down)
3 = DigIn: Activated w/ Supply Voltage (Pull Down/Pull Up)
.PinConfig1: 0 = 0 - 5.26 V (mid range)
1 = 0 - 0.3675 V (low range)
2 = 0 - 35.3 V (high range)
3 = 0 - ~2.465 V (not recommended!)
.DigInConfigHigh: Digital Input switchpoint High [mV]
Default Value 3000 mV
.DigInConfigLow: Digital Input switchpoint Low [mV]
Default Value 2000 mV
.FreqConfig: 0 = No additional measure mode
1 = Quad encoder enabled, result in .QuadCount
2 = Phase shift enabled, result in .Phase
Note: When .FreqConfig = 1 or 2
C2p05 is master and C2p06 is slave
0
0
3000
2000
0
U16 PinConfig0
U16 PinConfig1
U16 DigInConfigHigh
U16 DigInConfigLow
U16 FreqConfig
C2p05
Config
C2p05_MFIn
Refer to the API document for details of
threshold values and functionality.
.DigIn: Digital Input Active = True
DigIn
.AnIn: Analog Input 0 - 32767
AnIn
.Volt: Analog Input scaled [mV]
Volt
.Freq: Frequency Input scaled [Hz]
Freq
.Per: Period input scaled [0.1 us]
Per
.Count: Number of measured counts this loop
Count
.Duty: Positive duty cycle Input scaled [0.01%]
Duty
.QuadCount: Number of measured counts this loop,
sign defines direction. Valid if .FreqConfig = 1
QuadCount
.Phase: Phase shift [0.1 us], sign defines direction
Valid if .FreqConfig = 2
Phase
.FreqStatus: 0 = OK
1 = Config Error
FreqStatus
.PinStatus: 0 = OK
1 = Config Error
PinStatus
MAG PICK UP 5/5
Inputs
MC24_10_MC24_12
C2p05_MFIn
TOP!Inputs!C2p05_MFIn
10/22/2016
44( 79 )

# 6. GASOLINE FUEL SYSTEM

## [1] FUEL LINE

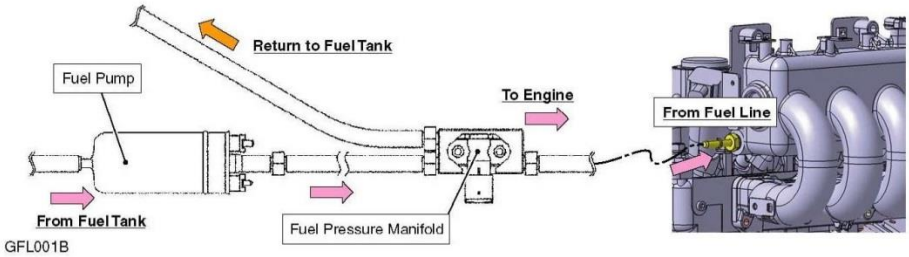



**Figure 1-24**

## [2] FUEL PUMP


**Figure 1-25**

**Installation Requirements**

1. Maximum vibration levels not to exceed 3 G.
2. Maximum allowable suction pressure to be ≤ 10 kPa. (Below atmospheric pressure)
3. Pump body not to be secured by excessive clamping force causing any deformation of pump housing.
4. Teflon tape not to be utilized during fitting assembly.
5. Pump should not be mounted in high temperature environments to prevent cavitation vapor lock conditions.



Kubota, 2014. Kubota Engines Application Manual (WG1605-E3, WG2503-E3 Model) Third edition.

## [3] FUEL PRESSURE MANIFOLD

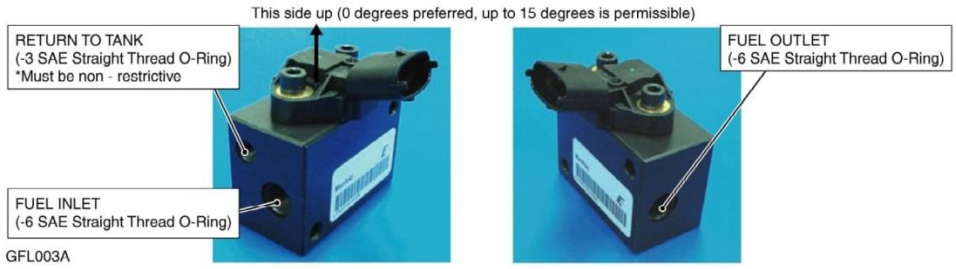



**Figure 1-26**

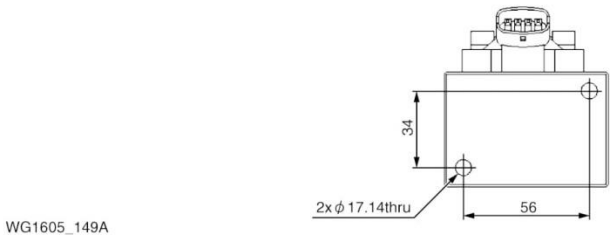



**Figure 1-27**

Fuel Pressure Manifold enables gasoline fuel system to meet the EPA evaporative emissions requirements implemented for industrial mobile LSI engines in 2007. The ECU equipped to provide Pulse Width Modulation control of the fuel pump. This control feature and fuel pressure manifold combination prevent the build-up of excess heat in the gasoline tank which in turn reduces the evaporative emissions.

**Installation Requirements**

1. 0 degrees preferred, up to 15 degrees is permissible, over 15 degrees is not permissible.
2. Device must be mounted direct (no rubber isolation) to frame rail in relatively cool area (not to exceed 85 °C)
3. The volume of fuel between the fuel pump and this device must be a minimum of 25 $cm^3$, yet can not exceed 75 $cm^3$ without approval.



Kubota, 2014. Kubota Engines Application Manual (WG1605-E3, WG2503-E3 Model) Third edition.

## [4] WIRING FOR FUEL PRESSURE MANIFOLD

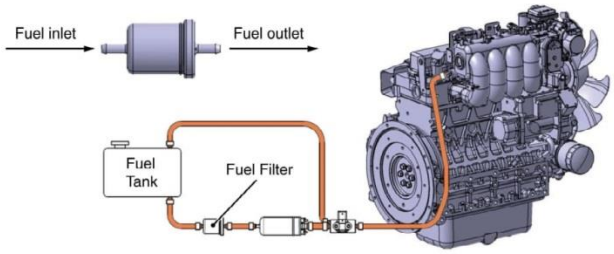
FEA must prepare the relay wire harness between the wire harness and the fuel pressure manifold (FPM).
Wire kind : SAE-J1128-TXL or equivalent
Wire size : AWG18 or equivalent
Heat resistant convoluted tube ( > 120 °C) shall be convolved.


GFL004B

**Figure 1-28**

## [5] FUEL FILTER

Kubota supplies fuel filter (12581-4301Δ).
Fuel filter shall be installed between fuel tank and fuel pump.




WG_133A

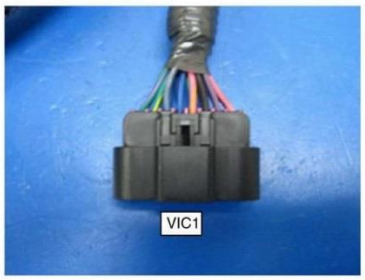
**Figure 1-29**



Kubota, 2014. Kubota Engines Application Manual (WG1605-E3, WG2503-E3 Model) Third edition.

# 10. VEHICLE INTERFACE CONNECTORS

## [1] VEHICLE INTERFACE CONNECTORS 1 (VIC1)





"A" - ignition input
"B" - charge indication
"C" - fuel pump -
"D" - fuel pump +
"E" - AUX PWM2
"F" - start command
"G" - MIL
"H" - fuel select input

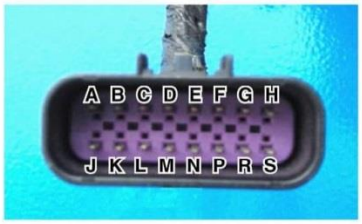
"J" - FPP2
"K" - FPP1
"L" - 5V RTN
"M" - 5V EXT
"N" - CAN1 +
"P" - CAN1 -
"R" - 5V RTN2
"S" - 5V EXT2

FEA needs to prepare the mating parts as below.
Manufacturer : DELPHI
Connector : 15326868
Terminal : 15326269
Seal : 15366060
Lock : 15358785

VIC001B

**Figure 1-43**

## [2] VEHICLE INTERFACE CONNECTORS 2 (VIC2)


("A" - not used)
("B" - AUX PWM5 RECIRC)
("C" - AUX PWM5)
"D" - TACH
2 pulse / rev (GCP)
any pulse / rev (4G)
("E" - GROUND)
("F" - AUX PWM1)
("G" - AUX ANA PU2)
("H" - AUX ANA PU3)

"J" - GOV SELECT 2
"K" - GOV SELECT 1
("L" - AUX DIG2)
("M"- AUX DIG3)
("N" - VS+)
("P" - VS-)
("R" - AUX ANA PD1)
("S" - 12V RELAYED POWER)

In case FEA uses VIC2, FEA needs to prepare the mating parts as below.
Manufacturer : DELPHI
Connector : 15326863
Terminal : 12191819
Seal : 15366060
Lock : 15358785

VIC002C

**Figure 1-44**



Kubota, 2014. Kubota Engines Application Manual (WG1605-E3, WG2503-E3 Model) Third edition.