Spring 3-10-2022

# Development of Controller Area Network Bus Programming for Quarter-scale Tractor Pulling Sled

Brittani Wacker

*University of Nebraska - Lincoln*

Development of Controller Area Network Bus Programming
for Quarter-scale Tractor Pulling Sled

An Undergraduate Honors Thesis
Submitted in Partial fulfillment of
University Honors Program Requirements
University of Nebraska-Lincoln

by
Brittani Wacker, BS
Agricultural Engineering
College of Engineering

3/10/2022

Faculty Mentor:
Roger Hoy, Ph.D., Biological and Agricultural Engineering

Abstract

Usage of Controller Area Network (CAN) technology has allowed for advancements in many industries, including agriculture. Most importantly, its application is useful for refining data acquisition methods. With support from the Quarter-scale Tractor Design Team at UNL, this project united a CAN bus with the team's current tractor pulling sled. The objectives were to install new instrumentation needed for the CAN bus and to program the updated system utilizing CAN data. The program needed to give the pulling sled functionality and the ability to read and log important pulling data. These goals were all accomplished by implementing new sensors, coding Danfoss hydraulic features, and analyzing input CAN data. Further improvements can be made in the future, but the current Quarter-scale team will benefit from the enhancement in data acquisition.



Keywords: Agricultural Engineering, CAN bus, Danfoss, ISO bus, Small Tractor, Tractor, Tractor Pulling

# Dedication

Completion of this project would not have been possible without the support of the Biological Systems Engineering department, both financially and academically. More specifically I'd like to extend my gratitude to Dr. Roger Hoy, my advisor, for being a constant source of encouragement and wisdom. Dr. Hoy has always had faith in my abilities, even if I did not, and I will always be thankful for the inspiration he gave me.

Beyond the BSE department, I want to thank the members of the Quarter-scale Tractor Team for their support and friendship. Looking back, much of my time at UNL was spent with or around the Quarter-scale team, and I'm grateful I was able to find my place here with them. Of course, this project would not have been an opportunity for me without the work done by this group of people, so I attribute much of my work to their dedication and commitment to the team's success.

# Table of Contents

# Introduction

Advancements in agricultural technology have historically been focused on mechanical advancements of power, such as an increase in horsepower. In the last few decades, improvements in computers and electronics have been crucial to empowering farmers to become more efficient and productive. Access to vast amounts of data has been one of the most important achievements; an example of this is the use of the Controller Area Network (CAN). The use of this data communication framework in large vehicles has become the industry standard for the electronic sharing of information.

Since CAN bus systems are regularly utilized in the automotive and agricultural industry, it is important for engineering students interested in vehicles and machinery to learn its workings. For this project, an opportunity was presented to not only learn the mechanisms behind CAN but also to apply it via an extracurricular project. Working alongside members of the UNL Quarter-scale Tractor Design Team, this project focused on the design of a program for the team's tractor pulling sled to utilize CAN bus technology for improved data collection. Using this data allowed the team to evaluate their tractor designs more effectively.

For many years, the Nebraska Quarter-scale Tractor team has been greatly successful at our annual competition. After designing and testing a lawn mower-sized tractor, which operates using a CAN bus, the team travels to the International Quarter-Scale competition in Peoria, Illinois. The UNL team regularly places high in many of the events, which include design judging, tractor pulling on a dirt track, and evaluations of durability and maneuverability. Their continued success can be attributed to many things, especially their supportive advisor and department. One key to the team's success is the thorough testing conducted using applicable instrumentation, such as the pulling sled. By adequately and accurately simulating the competition conditions via CAN features, the team is more likely to succeed at the competition. Thus, the outcome of this project enables the UNL Quarter-scale team to continue the current trend of success and improve upon their designs in the future.

This project intended to utilize the current pulling sled (shown in Figure 1) as a platform for the implemented programming. To gather data important for calculations, sensors or other instrumentation were added or updated to receive more input information.

## Objectives

The objectives specified for this project were to:

1) Implement new instrumentation required to gather data necessary for the determination of pulling distance, speed, and force.
2) Create programming to receive and transmit CAN messages between the microcontroller and other ECUs, including the visual display, joystick, engine, and connected tractor.


Figure 1: Picture of pulling sled being tested behind a Case IH DX55 tractor.

## Background

The use of Controller Area Networks is an industry standard in several fields for many reasons. A CAN bus is a way for ECUs in an automotive or agricultural vehicle to communicate with one another reliably (Smith, 2021). For one, CAN doesn't require a host computer to allow communication between all ECUs (or nodes) on the bus, which streamlines the data transmission process and reduces the weight of electronic systems. Each ECU can receive, process, and transmit data. This data is released onto the network, where any other ECU may access it (Smith, 2021). Access to CAN data is quite easy; one connection point will allow communication with all connected ECUs at once. Given the benefits above, it is clear why many industries depend on CAN and that the Quarter-scale team also utilizes this technology.

One facet of CAN that is also useful is that messages are all prioritized with a given ID. Data is organized into a hierarchy of importance, allowing the most pertinent information to be communicated first. For this project, the IDs were interpreted using the SAE J1939 standard. Using this standard, CAN messages can be decrypted and translated into useful data. For example, to access a specific piece of data, perhaps the GPS location of a device, there is a standardized PGN (Parameter Group Number) that contains this bit of information (Hennessy, 2019).

In Danfoss Plus+1 graphical programming, receiving and transmitting CAN messages is quite simple This programming interface uses “wires” in place of lines of code to string together actions. Similar to how LabVIEW programming works, several blocks of prewritten code execute different actions, and many of these blocks are CAN-specific in Plus+1, as shown in Figure 2. Using these message blocks and other specified function blocks, one can program any Danfoss microcontroller or other component, which are found throughout the pulling sled.

# Materials & Methods

## Existing Components

Before this project, a majority of the pulling sled was already manufactured by previous team members, which was a constraint of this project. The base framework of the sled was well-developed, which included hydraulic and steering components, a brake system, and the current steel chassis. Design of the weight box system was previously done also, consisting of a

Figure 2: Example of reception of a CAN message in Plus+1 code.

large lead screw that advanced the weight load forward from the back of the pulling sled toward the hitch of the tractor. More recently, Quarter-scale team members transferred a new engine into the sled, which was CAN addressable.

The primary feature of the pulling sled was the microcontroller. A Danfoss MC050 acted as the brain of the CAN bus, processing raw data and transmitting it to the user. The main benefit of using this microcontroller was that all 50 pins were configurable through programming. For example, one pin might only serve as an output while another received inputs; others still were configurable to both conditions. This controller was configured and programmed directly through the Plus+1 coding scheme, where the format of the program is determined by the type of controller, seen in Figure 3. To accompany the controller, a Danfoss display screen and joystick were used for operating the sled. The configuration of these controls is shown in Figure 4. The display screen (Danfoss DP600) showed relevant pulling information and engine RPM settings, which were included in this project to accommodate the Quarter-scale team's preferences. A joystick (Danfoss JS6000) was used to actuate the weight box motor, axle lift cylinder, and the ground drive motor, as shown in Figure 5. Installed in the CAN bus were two J1939 access ports; one in the operator's station and one at the front hitch to attach and monitor a tractor, with the latter shown in Figure 6. These useful components were all to be included in this update of the sled.

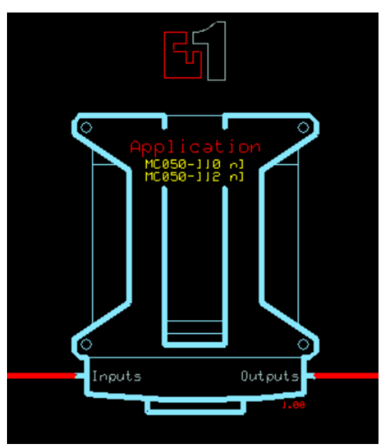


Figure 3: Screen capture of the Plus+1 home page, featuring the chosen MC050 controller.

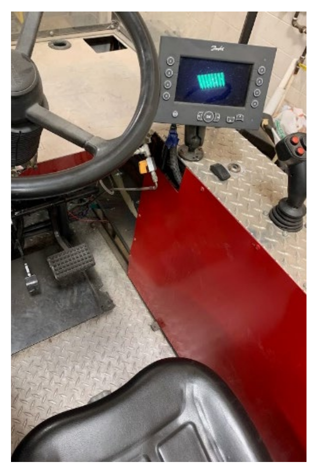

Figure 4: View of operator's station control panel from the operator's seat.

Figure 5: Image of physical JS6000 Danfoss Joystick on pulling sled. The red buttons actuate the lift cylinder and the toggle button moves the weight box. Motion on the Y-axis propels the sled.

Figure 6: Image of the front electrical box, which encases the second J1939 connector for the attached tractor, load cell connector.

The hydraulic system was key since the motion of the weight box and the pulling sled was controlled hydraulically. All the hydraulic features were from Danfoss, thus all the components were programmable using Plus+1 coding. Both the weight box and main drive wheels used a Sauer Danfoss M44 hydraulic motor connected via chain drive to propel each output. These motors received fluid flow from a tandem variable displacement pump (Sauer Danfoss H1T053). A Kubota gas engine powered the hydraulic pump, which was also interfaced with CAN programming. Installed below the weight box rails was a hydraulic cylinder that actuates to lift the front drive axle and lower the sled pan during pulling. Further details of the hydraulic circuit can be found in Appendix A, including a schematic drawing.

Another important feature of the pulling sled was the attached load cell. A load cell uses a change in mechanical strain to determine the amount of force that is being exerted (OEI, 2020). It exported this data as a voltage, which was translated to force by the processing unit in the microcontroller. A picture of the load cell used is shown in Figure 7. This sensor was important for tractor testing since it helped determine how much pulling force the was able tractor to exert. It was important to measure this for replicability of the pull. If the same amount of force was applied each time, then each pull can be recreated and made constant. Changes in performance are much more obvious this way, so the Quarter-scale team can better determine the ability of a specific tractor and adjust accordingly.

## New Components

New components were needed for the new programming to be successful. This included new wiring throughout the sled and more sensors for the acquisition of necessary data. Safety switch mechanisms were also faulty, which were updated. Previously, the Quarter-scale team had used limit switches on both ends of the weight box railing to prevent overrun of the box, and these switches needed replacement. More limit switches were also installed on the front axle lift cylinder to signify when the pan was lowered completely. Further information about the electronics and wiring can be found in Appendix B.

Figure 7: Photo of the current load cell setup used by the Quarter-scale team.

A rotary encoder was necessary to calculate distance traveled, which is crucial information for a tractor pull. To effectively implement this sensor, a device needed to be designed that freely traveled across the ground during pulling. The device was a fabricated "fifth wheel" attached to the sled that moved independently.

One hydraulic component was also needed for the successful update of the pulling sled. A way to "freewheel" or allow the drive axle to freely rotate was important for a proper pulling mode. This way the pulling sled truly became a dead weight on the tractor and did not propel itself or resist motion. The chosen method to do this was to provide relief to the pressure in the hydraulic motor by opening access to another flow circuit during a pull. To achieve this, an electrohydraulic valve was required.

## Programming

To consider this project a success, the program needed to include the functionality of the pulling sled and data processing of sensor inputs. Before writing any useful code, the created program needed to be clear and accessible. Different pages embedded within the program filled this need and helped make a more legible and understandable program. Knowing that future Quarter-scale team members needed the ability to quickly edit or understand the coding, this was the first required step.

The next priority was the engine since the sled would be motionless without it. To program the engine, several function blocks existed to streamline the coding. These blocks of code automatically read a specific CAN message with a determined PGN, shown in Figure 8. Along with another CAN receive function block, the engine can be completely controlled.

With the engine programming completed, the next focus was on other inputs, either from sensors or the joystick. For the most clarity, inputs were handled all in one page of the program and renamed appropriately for use in the coding further on. Inputs from the load cell, rotary encoder, joystick, and other signal pins were all programmed in this section of the code.

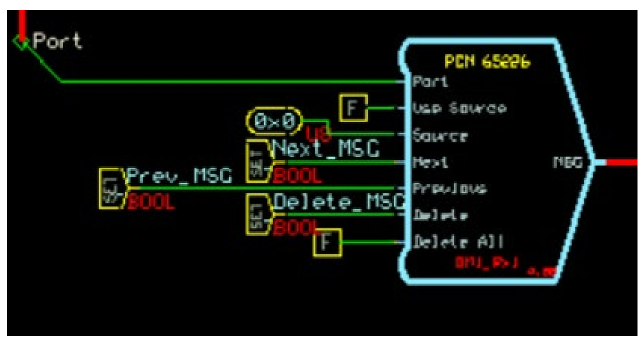


Figure 8: Screenshot of example PGN function block used for engine programming.

With all input values addressed, processing these values into useful data was next. For instance, the load cell voltage was converted to a force in pounds in this section. To convert this quantity correctly, a calibration was regularly completed to determine the slope and intercept in the equation relating the output voltage to the exerted force. Finding both values was done by finding the trendline that best fits the plot of voltage versus applied force, shown in Figure 9. Other calculations were also done to convert the output signal of the rotary encoder to a distance travel value, which was dependent on the frequency of the encoder. Other more straightforward values were processed, like a signal from a limit switch indicating that the motion of a hydraulic component should stop. This was executed with a basic conditional code.

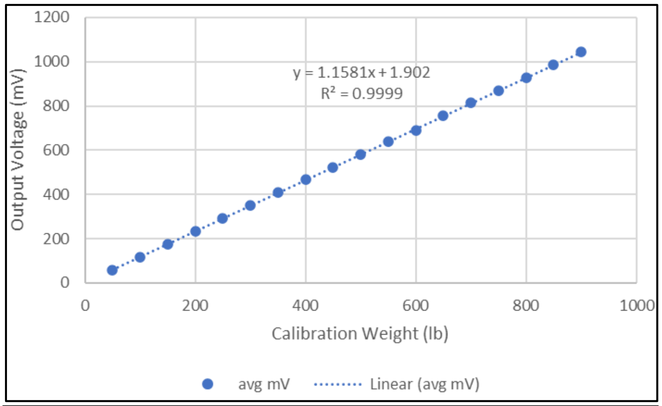


Figure 9: Chart made for calibration of load cell of signal voltage versus applied force. The trendline here was found in March 2021.

Once all the input data was processed, it was used to inform the operator or execute other actions. The sensor data from the load cell and encoder was used to report live pulling feedback to the operator via the display screen. This was done by transmitting specified CAN messages to the display screen. Input from the joystick was sent to function blocks that control the hydraulic cylinder and motors to move the front axle, weight box, or drive wheels.

Beyond programming the microcontroller, another program was created for the display screen. This was also done using Plus+1 on a separate file to upload specifically to the screen. The majority of this program used images and text editors to design the display. Using received CAN message data, aspects of the display were controlled based on certain values. Values like engine RPM were live reported simply for the user to monitor during operation.

Another intention of the program outside of the code itself was the inclusion of a method to export and record data. The programming software used had its own data logging software companion, Plus+1 Service Tool. Checkpoints were added in the code (shown in Figure 10) to flag important data points for logging. A service tool can be programmed to log desired data, or a new one can be created for an individual pull. Logged data can be exported to a Microsoft Excel file as well, which further made adding checkpoints useful.

Figure 10: Example of a checkpoint used in the Plus+1 code for identification in the Service Tool.

## Results & Discussion

### Installed Components

As required by the program, several features were added to the sled to increase functionality for the Quarter-scale team's testing purposes. All components were rewired through a new wiring harness. Finer details on this can be found in Appendix A. The rotary encoder was installed alongside a "fifth-wheel" bike tire mount, which was manufactured and welded during this project. This feature is pictured in Figure 11. Limit switches were installed both on the weight box rails and the axle lift cylinder to prevent overrun of these systems. An example of these switches is seen in Figure 12. The hydraulic valve to enable "tow mode" was installed successfully into the hydraulic circuit. During testing of the updates, this feature performed well and served its purpose. The only downfall to this addition was that the pulling sled must be restarted to reengage the drive wheels.


Figure 11: Image of "fifth wheel" bike tire and suspended mount implemented to employ use of the rotary encoder.

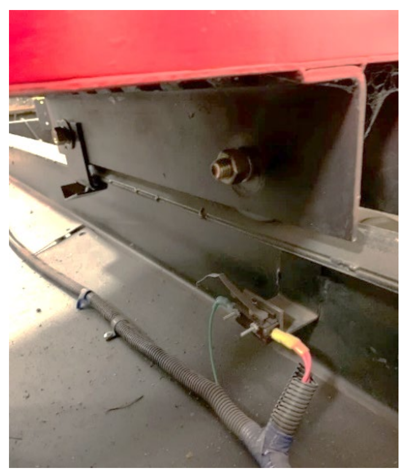
Figure 12: Example setup of a limit switch on the weight box rail. The attachment attached to the weight box will advance toward the switch and trip the system to stop motion of the box.

## Program Capabilities

As intended, the program design gave functionality to the pulling sled and enhanced its testing abilities for the Quarter-scale team. As shown in Figure 13, the top-level organization of the program added clarity and efficiency. In Figure 14, the data received from the engine was well-displayed for the operator on the display. The display also gave live updates during a pull. For example, the progress of the tractor during a pull was simulated by a graphic on the pulling page, which moved an image of a tractor down a “track” according to the input final distance and current distance traveled (Figure 15). Further documentation of the Plus+1 program is available in Appendix C.

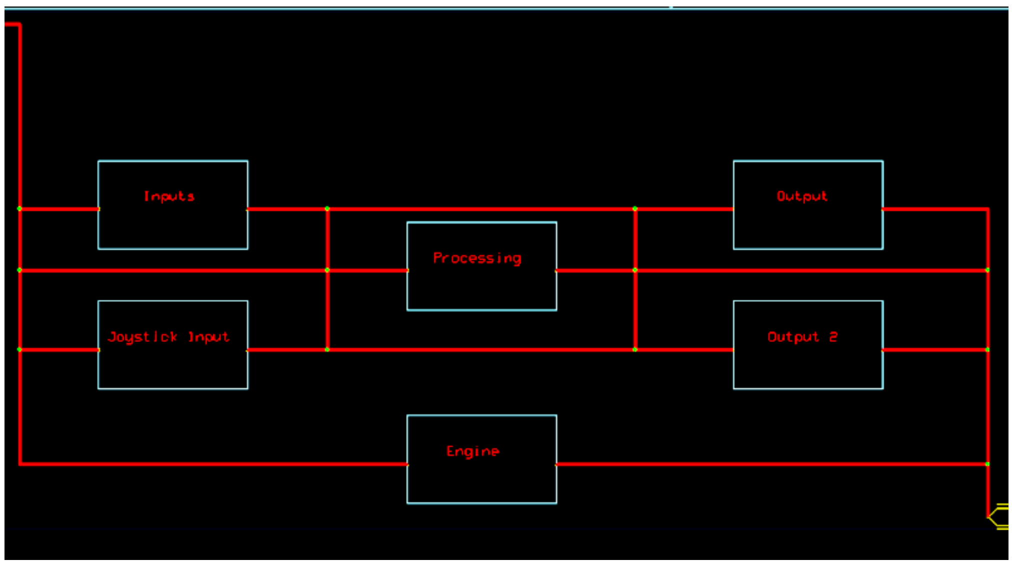


Figure 13: Screen capture of top-level organization of the Plus+1 program.


Figure 14: Picture of engine view on display screen.


Figure 15: Picture of pulling screen.

Physically, the intended programming operated smoothly. Starting the engine was quick and simple, which will accommodate many different operators. The joystick-operated drive system took acclimation, but the fine control of the pulling sled's motion was effective. Moving the weight box was slow but was reasonably paced for attempting to move thousands of pounds of dead weight. The limit switches were installed and operated as they should, preventing motion beyond the range of the system. The use of the load cell required frequent calibration, but otherwise, it reported useful data to the screen. The new fifth wheel also was useful and traveled along the ground smoothly using the suspension spring built into it.

## Future Improvements

In the near future, the most important improvement for the pulling sled will be automating the pulling process in a way that starts and stops the pull at the touch of a button. Currently, the main persisting inconvenience is that the inherited chassis of the pulling sled is too heavy for a Quarter-scale tractor to effectively mimic a competition-like pull. To address this weight issue, a further improvement will be to slowly lower the pan to ease the tractor into the start of a pull rather than begin the pull with the pan fully on the ground. The team has tried several methods of balancing this issue, one of which included ballasting the pulling sled on the rear end to relieve the attached tractor when starting the pull. None of these methods have been successful, but being able to program a "push start" feature will allow the tractor to gain momentum and establish traction without being overloaded. The operator during a pull should be able to focus on the incoming data rather than on the tractor.

## Conclusion

Upon completion of this project, it is clear that the UNL Quarter-scale team will benefit from the updates made to their pulling sled. As accuracy is important to the team, it was important for the program and instrumentation implemented to yield useful data and to operate the pulling sled in an easily accessible manner. After a review of the results, this project has accomplished this goal. Sensors were implemented and yielded valuable insight to pulling; the program gave function to the sled and communicated CAN messages among the ECUs. Looking forward, the Quarter-scale team will benefit from this project for years to come.

## References

DP. (2021). PLUS+1 GUIDE Software User Manual. Danfoss Plus+1. Retrieved from https://www.danfoss.com/en-us/products/dps/software/software-and-tools/plus1-software/#tab-software-tools-and-applications

Hennessy, B. (2019). An Introduction to J1939 and DBC files. Kvaser AB. Retrieved from https://www.kvaser.com/developer-blog/an-introduction-j1939-and-dbc-files/

International Organization for Standardization. (2015) 11898: Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signaling. Geneva, Switzerland: ISO. https://www.iso.org/standard/63648.html

KAB. (nd). SAE J1939 Introduction. Kvaser AB. Retrieved from https://www.kvaser.com/about-can/higher-layer-protocols/j1939-introduction/

Smith, G. (2021). What Is CAN Bus (Controller Area Network) and How It Compares to Other Vehicle Bus Networks? DEWESoft. Retrieved from https://dewesoft.com/daq/what-is-can-bus#what-is-can-bus

OEI. (2020). Load Cells & Force Sensors: What is a Load Cell & How Does it Work. Omega Engineering Inc. Retrieved from https://www.omega.com/en-us/resources/load-cells

# Appendix

## A: Hydraulic Schematic

Table A1: Identification of hydraulic components and models in the schematic (Figure A2).

| ID | FUNCTION | MODEL |
|---|---|---|
| **1** | Tandem Hydrostat Pump | Sauer Danfoss H1T053 Tandem Pump |
| **2** | Tandem Charge Pump | Charge Pump 83009499_ID |
| **3** | Ground Drive | Sauer Danfoss M44 motor |
| **4** | Box Drive | Sauer Danfoss M44 motor |
| **5** | Steering | Steering Pump EATON 211 |
| **6** | Pan Valve Block | Valve block 700744750 97xx |
| **7** | Return Manifold | |
| **8** | Cooler | SAE 14 MODINE 1A016217 OIL COOLER |
| **9** | Return Filter | |
| **10** | Charge Filter | |
| **11** | Steering Cylinder | |
| **12** | Pan Cylinder | |
| **13** | Tow Valve | |

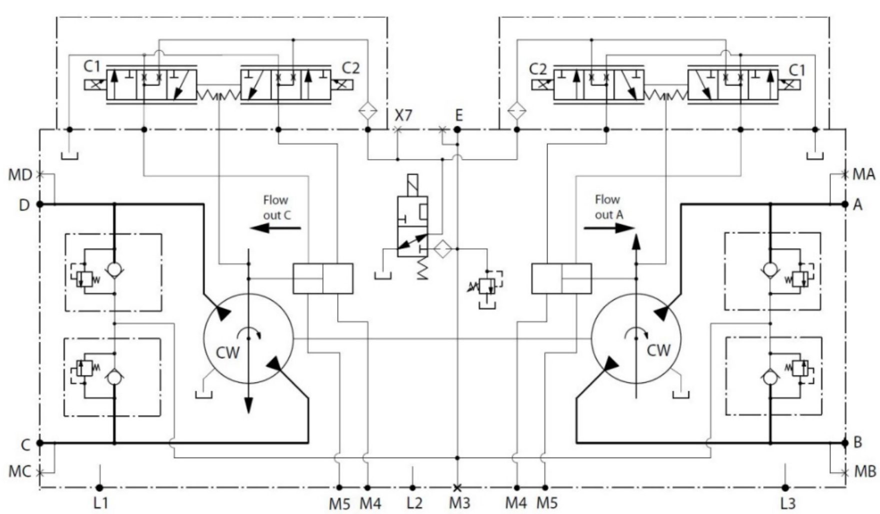


Figure A1: Hydraulic flow schematic of tandem variable displacement pump (ID 1 on Table A1).

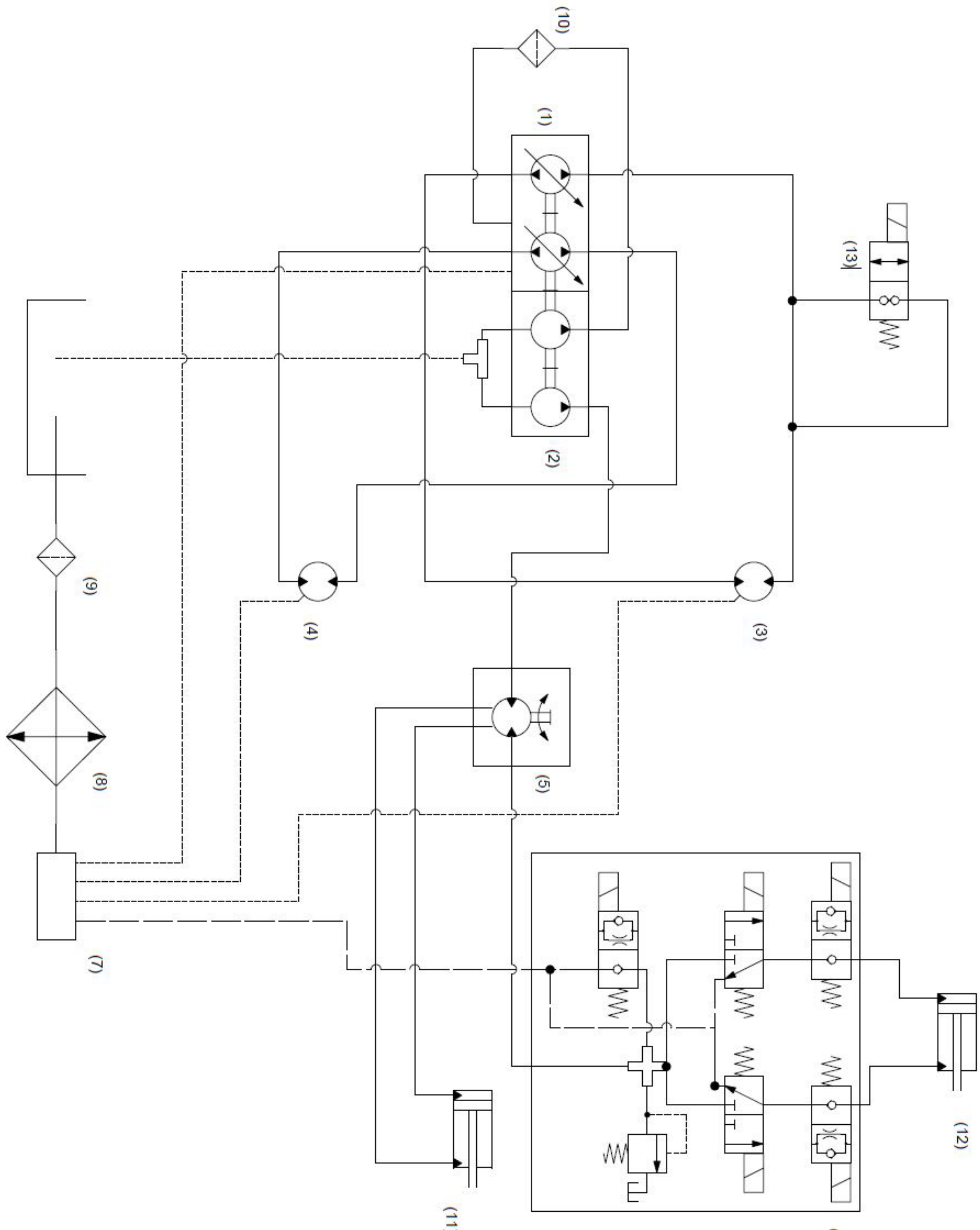


Figure A2: Schematic drawing of hydraulic components used on the pulling sled. This includes filters, motors, the reservoir, and all other components listed in Table A1.

## B: Wiring Diagram

Table B1: Designation of microcontroller connector pins for wiring reference and program usage.

| MC050 - Microcontroller | | |
|---|---|---|
| Pin | Description | Program Use |
| 1 | Ground | |
| 2 | 12V Power | |
| 3 | CAN Hi | Receive/transmit CAN messages |
| 4 | CAN Low | Receive/transmit CAN messages |
| 5 | CAN Shield | |
| 8 | 5V Power Out | |
| 10 | Box Home Snap Switch | Input – Stop Box Motor |
| 11 | Box End Snap Switch | Input – Stop Box Motor |
| 12 | Optional Key/Button Run Input | |
| 13 | Optional Key/Button Start Input | |
| 14 | Pan Up Switch Input | Input – Stop Pan Cylinder |
| 15 | Pan Down Switch Input | Input – Stop Pan Cylinder |
| 18 | 5th Wheel Encoder | Input – Ground Travel Distance |
| 19 | 5th Wheel Encoder | Input – Ground Travel Distance |
| 23 | Ground Speed Gear Tooth | Input – Travel Speed |
| 24 | Box Lead Screw Gear Tooth | Input – Box Speed |
| 25 | Load Cell Voltage | Input – Pull Force |
| 31 | Pan Solenoid 1 Extend | Output – Pan Cylinder Control |
| 32 | Pan Solenoid 2 Return | Output – Pan Cylinder Control |
| 36 | Pan Solenoid 3 Retract | Output – Pan Cylinder Control |
| 34 | Engine Run Out | Output – Engine Control |
| 35 | Engine Start Out | Output – Engine Control |
| 39 | Drive Wheel Swash Solenoid A | Output – Ground Drive Motor Control |
| 40 | Drive Wheel Swash Solenoid B | Output – Ground Drive Motor Control |
| 41 | Box Swash Solenoid A | Output – Box Motor Control |
| 42 | Box Swash Solenoid B | Output – Box Motor Control |
| 43 | Tow Mode Digital Out | Output – Tow Mode Control |

Table B2: Designation of connector pins for other CAN nodes, including display screen, joystick, and J1939 port connectors.

| DP600 – Display Screen (12 pin connector) | |
|---|---|
| Pin | Description |
| 1 | Ground |
| 2 | Power |
| 6 | CAN Hi |
| 7 | CAN Low |
| JS6000 Joystick | |
| 1 | Ground |
| 2 | Power |
| 3 | CAN Hi |
| 4 | CAN Low |
| J1939 Connectors (2 total) | |
| A | Ground |
| B | 12V Power |
| C | CAN Hi |
| D | CAN Low |

Table B3: Designation of Kubota engine connector pins and their connection points.

| **Kubota VIC-1 (Main Engine Connector)** | | |
|---|---|---|
| Pin Designation | Connected To: | |
| A - Ignition | Pin 34 | MC050 |
| F -Start | Pin 35 | MC050 |
| C - Fuel Pump Gnd | - | Fuel Pump - |
| D - Fuel Pump Pwr | + | Fuel Pump + |
| B - Charge | Amber Lamp | Optional LED Indicator |
| G - MIL | Red Lamp | Optional LED Indicator |
| N - CAN Hi | | |
| P -CAN Lo | | |
| **Kubota FPM (Fuel Pressure Manifold)** | | |
| Pin | Connected To: | *Jumper harness needed for fuel pressure manifold. Connectors included with engine accessories.* |
| 1 | A | |
| 2 | B | |
| 3 | C | |
| 4 | D | |

C: Samples of Plus+1 Program

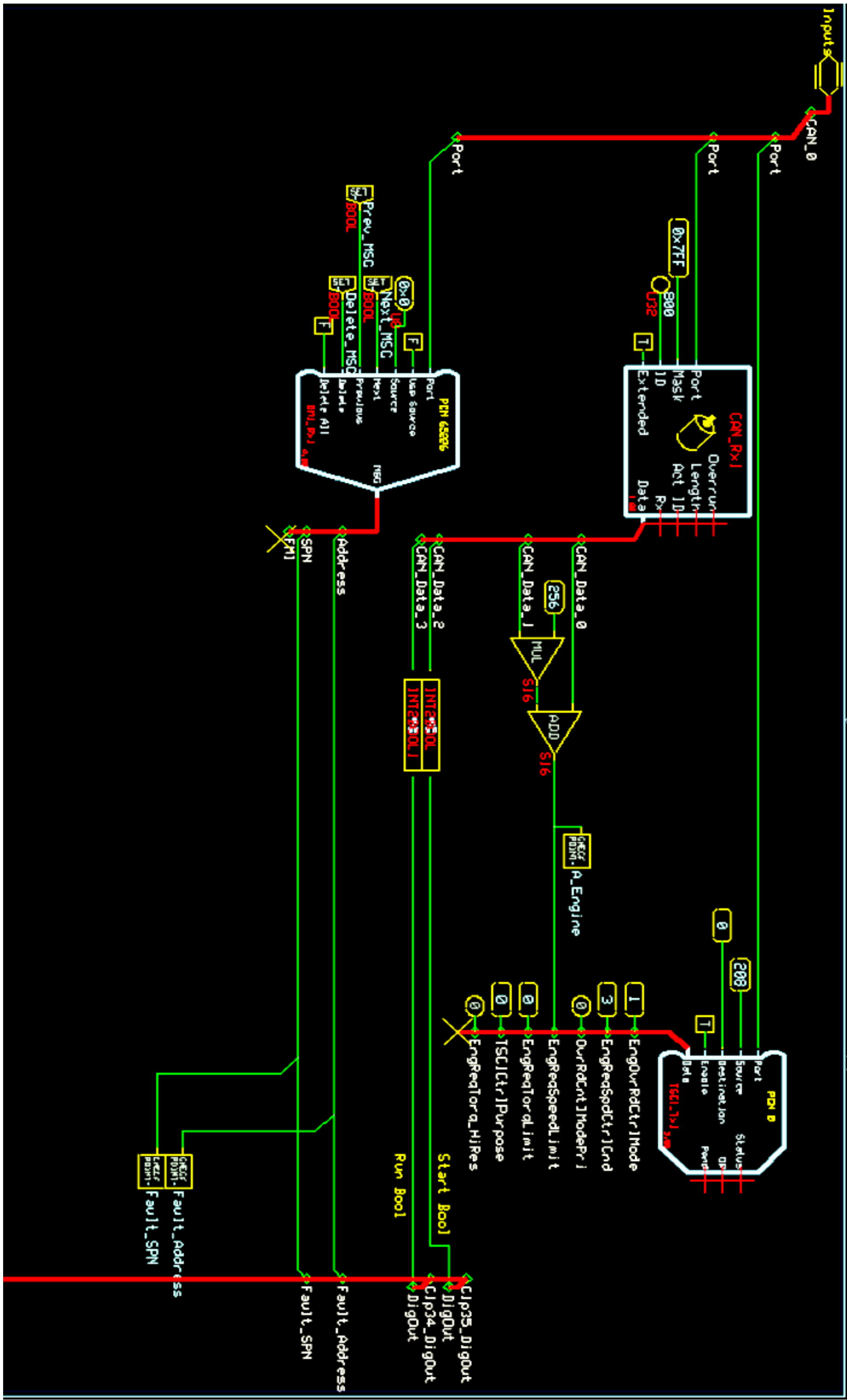


Figure C1: Screen capture of PLUS+1 engine operation programming.

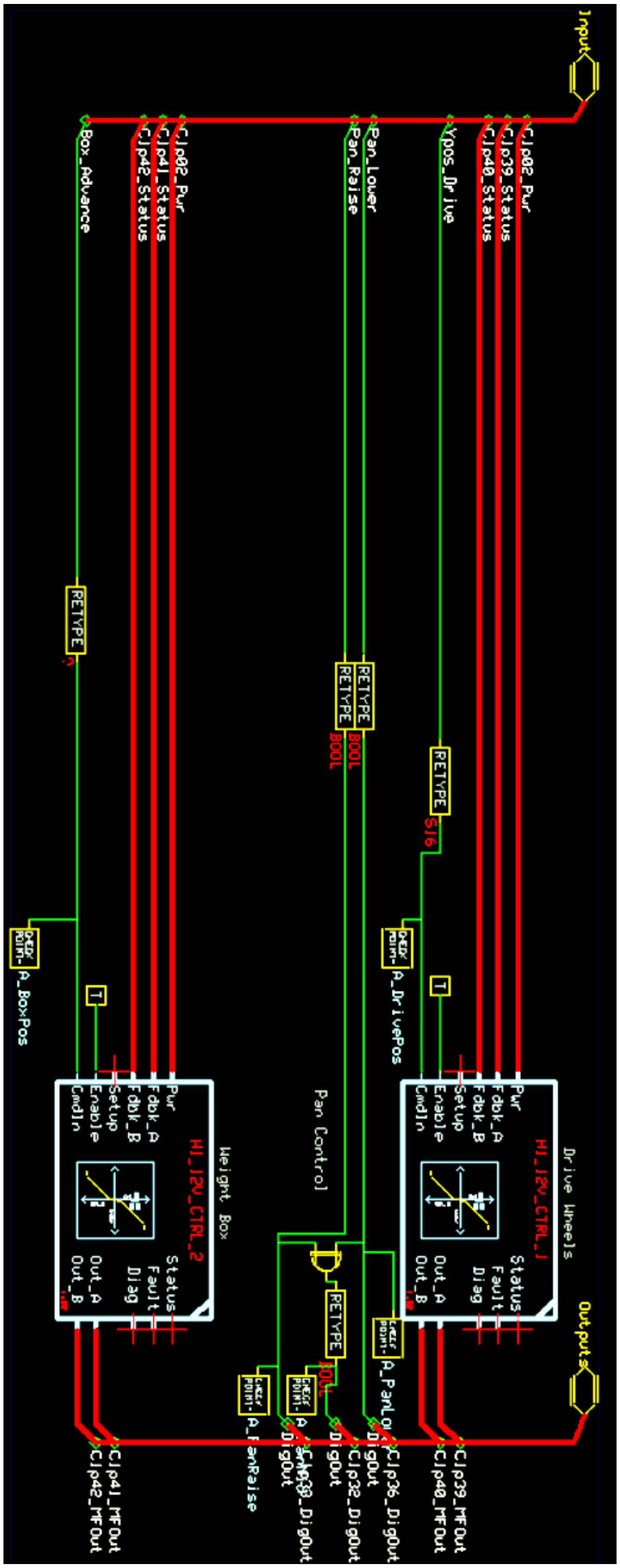


Figure C2: Screen capture of graphical code for operation of both M44 hydraulic motors.

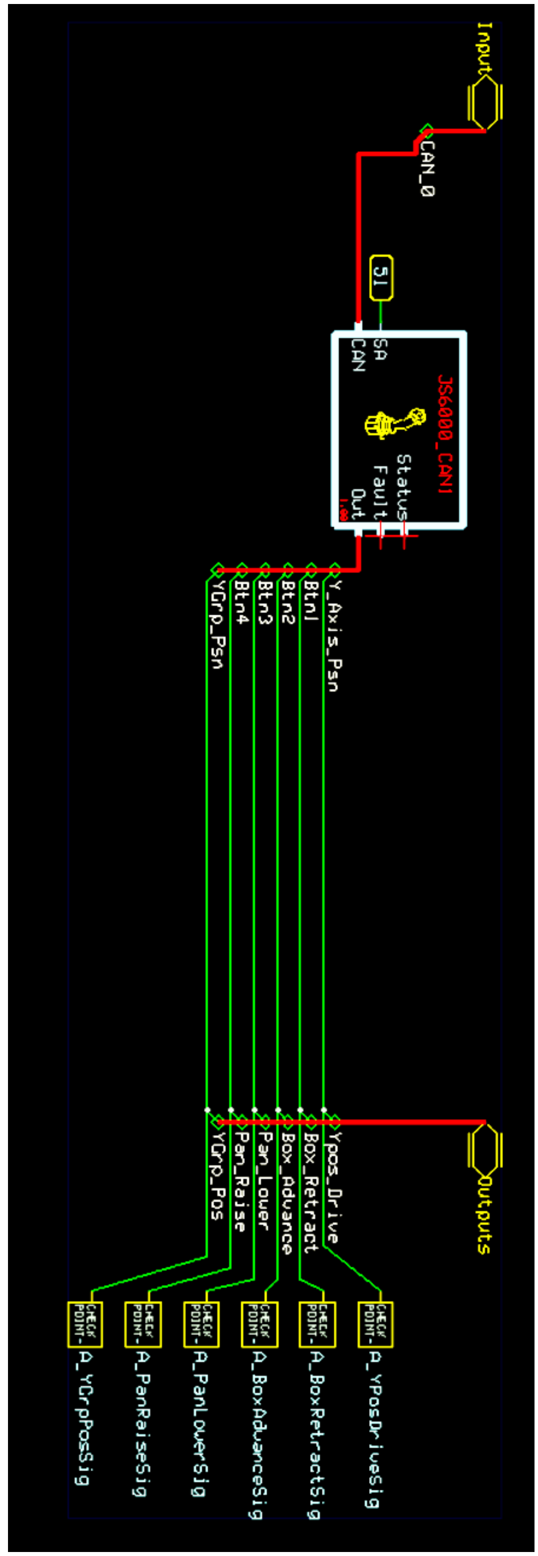


Figure C3: Screen capture of graphical coding to rename input values for easier reference in other program pages.

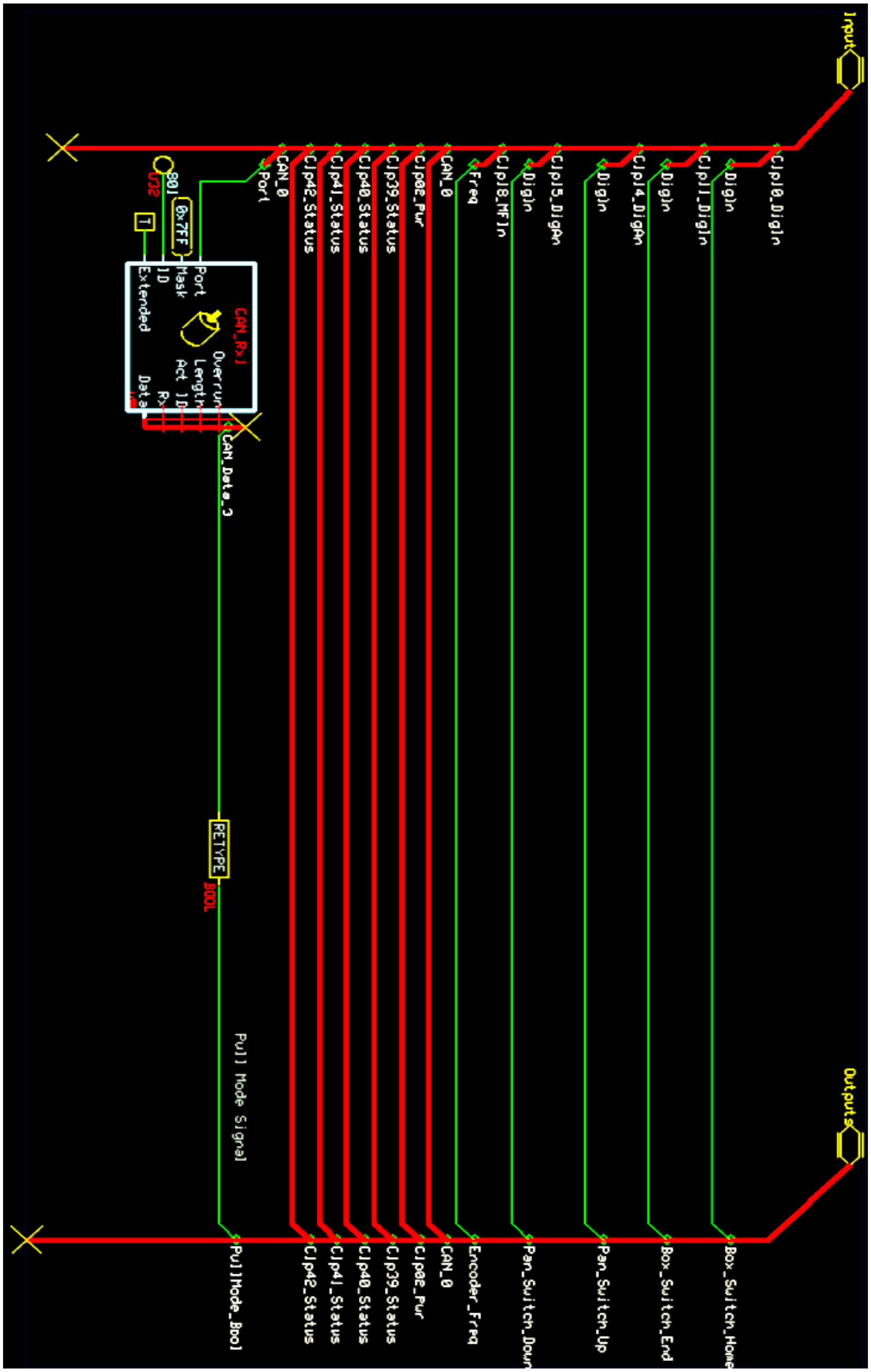


Figure C4: Screen capture of graphical code that connects joystick input values to their respective output names for use in other program screens.

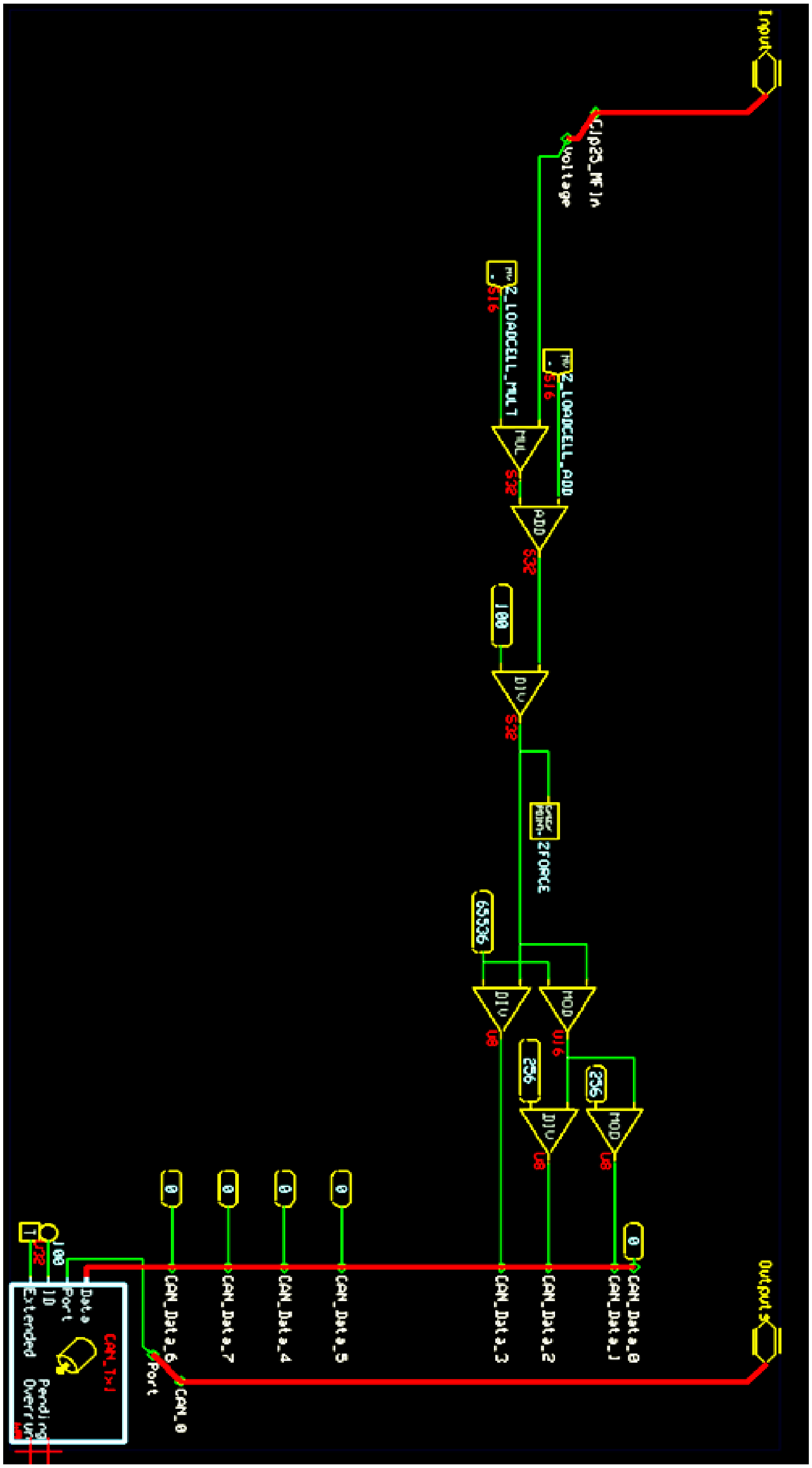


Figure C5: Screen capture of code for load cell voltage conversion to CAN message carrying pull force.