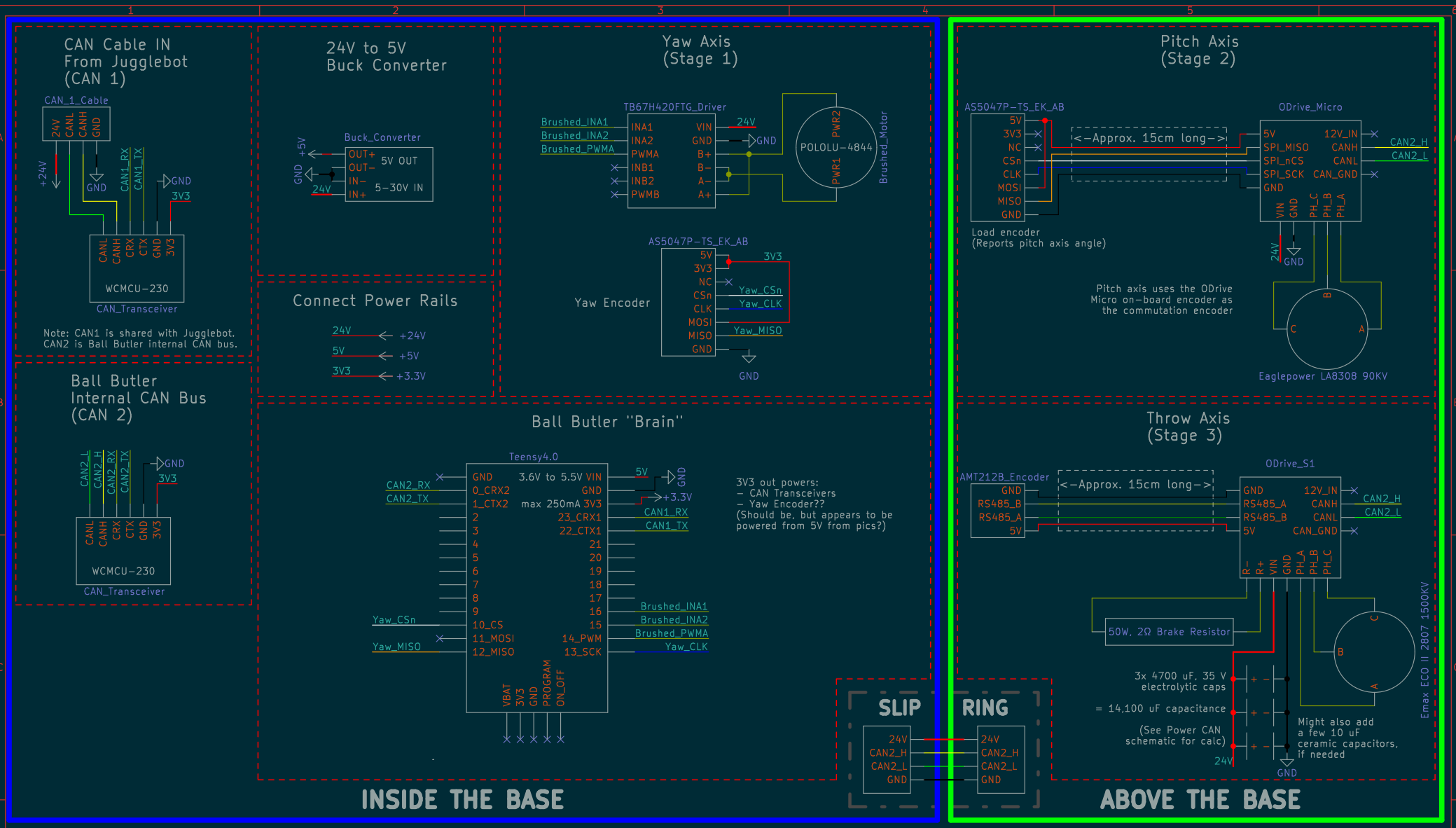


Programming Logic

File: programming_logic.kicad_sch

Power and CAN Schematic

File: Power and CAN.kicad_sch

CAD Model

Select and Press "d" to be taken to the latest CAD model

Sheet: /
File: ball_butler_circuit_diagram.kicad_sch

Title: Ball Butler Schematic

Size: A4	Date: 2025-09-04	**Rev: 2**
KiCad E.D.A. kicad (6.0.9)		Id: 1/3

CAN1 Frames

Throw Command
(To Ball Butler)

Frame Start	<---- Arbitration ----> (bits)											CAN Stuff	<--- Data ---> (bytes)								CAN Stuff	Frame End
	10	9	8	7	6	5	4	3	2	1	0		7	6	5	4	3	2	1	0		
	Command ID							Throw Num					t		z		y		x			

Command ID = 0X7D0
Throw Num indexes as a rolling sequence
(0-31. Loops back to 0 on overflow).

(x, y, z, t) describe the landing/target state of the ball.
x, y, z are 0.1 mm/LSB, giving a maximum value of ~6.5 m
t is 1 ms/LSB, giving a maximum time of ~1.09 min

Throw Feedback
(To Jugglebot)

Frame Start	<---- Arbitration ----> (bits)											CAN Stuff	<--- Data ---> (bytes)								CAN Stuff	Frame End
	10	9	8	7	6	5	4	3	2	1	0		7	6	5	4	3	2	1	0		
	Command ID							Throw Num					CRC16		v_z		v_y		v_x			

Command ID = 0X7D1
Throw Num reports back the commanded Throw Number

(v_x, v_y, v_z) describe the landing state of the ball
and allows Jugglebot to orient itself correctly.
These values are 0.2 mm/LSB, giving a max value of ~13.1 m/s

Checksum:
CRC16 (int16) requires CRC-16/CCITT-FALSE over `Throw Command` data (bytes 0-7)
[Allows Jugglebot to confirm catch vel pairing with throw command. Redundant with Throw Num]

Ball Butler Pose
(From Jetson)

Frame A

Frame Start	<---- Arbitration ----> (bits)											CAN Stuff	<--- Data ---> (bytes)								CAN Stuff	Frame End
	10	9	8	7	6	5	4	3	2	1	0		7	6	5	4	3	2	1	0		
	Command ID							-					CRC16		z		y		x			

Frame B

Frame Start	<---- Arbitration ----> (bits)											CAN Stuff	<--- Data ---> (bytes)								CAN Stuff	Frame End
	10	9	8	7	6	5	4	3	2	1	0		7	6	5	4	3	2	1	0		
	Command ID							-					q3		q2		q1		q0			

Pose data to be sent from Jetson at 1-10 Hz

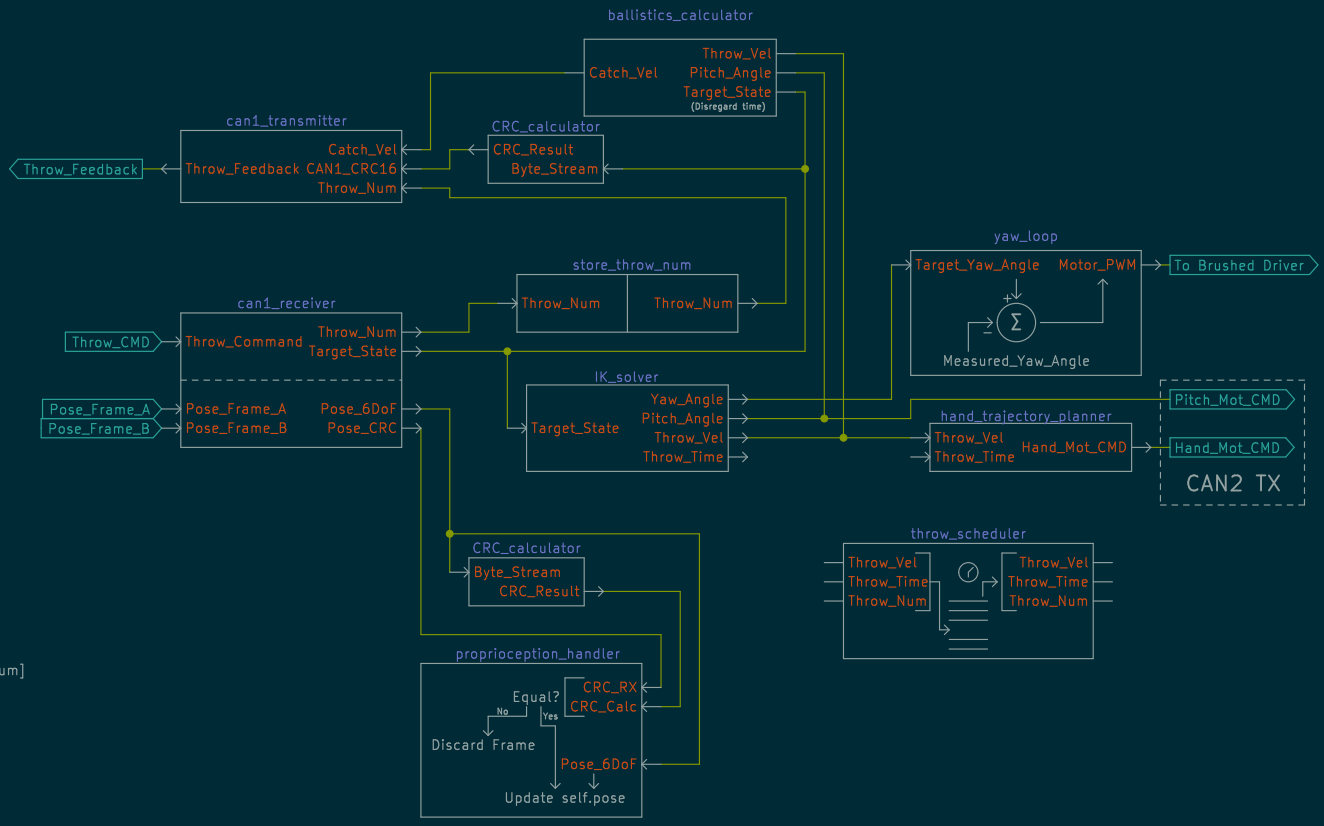
Command IDs:
Frame A = 0x7D2
Frame B = 0x7D3

(x, y, z, q0 .. q4) describes the current pose of the 'Base Box Lid'.

Position:
x, y, z (all int16) are scaled by 0.1 mm, giving a maximum value of ~6.5 m

Orientation:
q0 .. q4 (all int16) describe the orientation to approx 0.005°

Checksum:
CRC16 (int16) requires CRC-16/CCITT-FALSE over Frame A bytes 0-5 + Frame B bytes 0-7
[Ensures the correct pair of frames is matched]



CAN2 Notes

Pitch Motor Command

Pitch Motor Node ID = 1

"Set_Input_Pos" (0x0C) for Pitch ODrive (Micro)
Vel_FF and Torque_FF = 0

Hand Motor Command

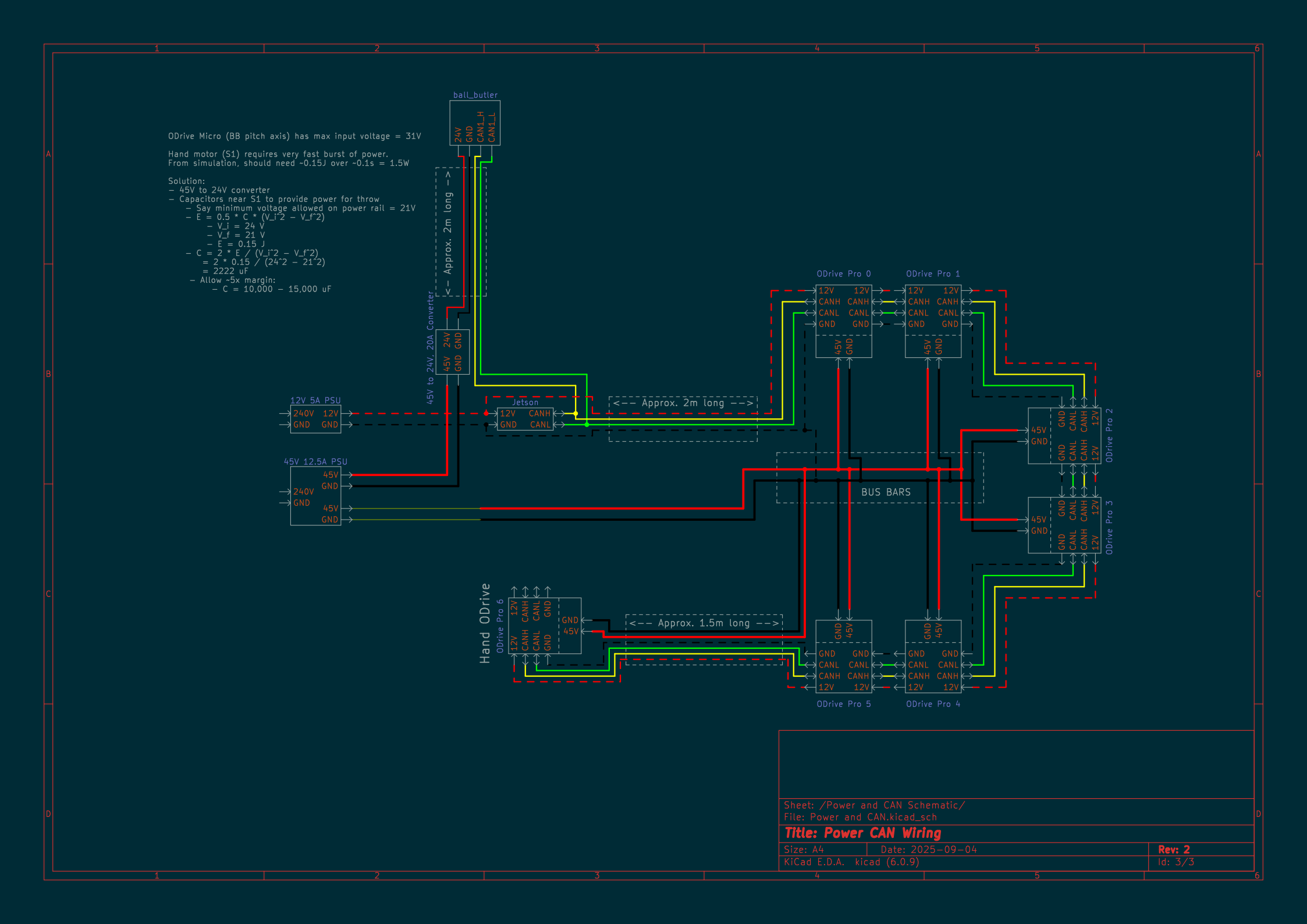
Hand Motor Node ID = 2

"Set_Input_Pos" (0x0C) for Hand ODrive (S1)
Input_Pos, Vel_FF and Torque_FF as calculated by planner

To Add/Think About

- ODrive Error handling?
- Command Scheduling

Sheet: /Programming Logic/		
File: programming_logic.kicad_sch		
Title: Programming Logic		
Size: A4	Date: 2025-09-04	**Rev: 2**
KiCad E.D.A. kicad (6.0.9)		Id: 2/3



Sheet: /Power and CAN Schematic/
File: Power and CAN.kicad_sch

Title: Power CAN Wiring

Size: A4	Date: 2025-09-04	**Rev: 2**
KiCad E.D.A. kicad (6.0.9)		Id: 3/3