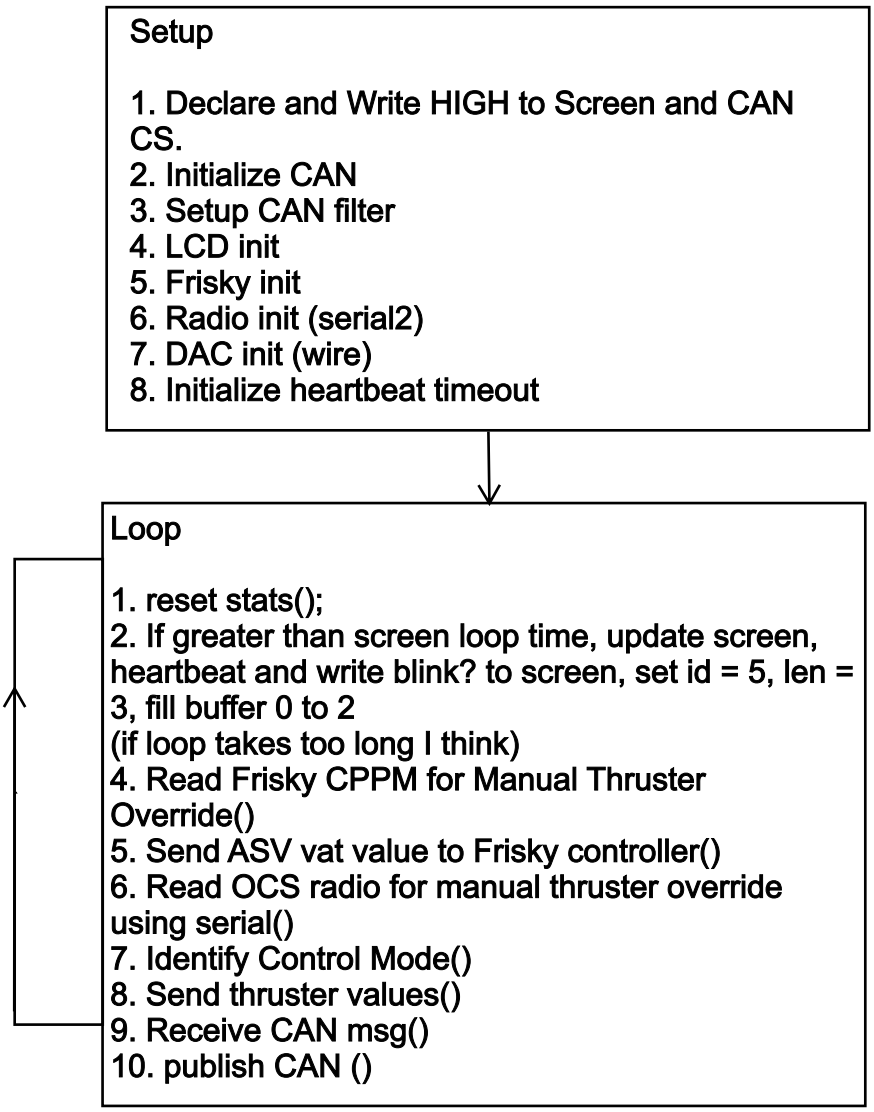

Setup
1. Declare and Write HIGH to Screen and CAN CS.
2. Initialize CAN
3. Setup CAN filter
4. LCD init
5. Frisky init
6. Radio init (serial2)
7. DAC init (wire)
8. Initialize heartbeat timeout
Loop
1. reset stats();
2. If greater than screen loop time, update screen, heartbeat and write blink? to screen, set id = 5, len = 3, fill buffer 0 to 2
(if loop takes too long I think)
4. Read Frisky CPPM for Manual Thruster Override()
5. Send ASV vat value to Frisky controller()
6. Read OCS radio for manual thruster override using serial()
7. Identify Control Mode()
8. Send thruster values()
9. Receive CAN msg()
10. publish CAN ()

1. Reset stats();
   reset_posb_stats();
   reset_ocs_stats();
   reset_rc_stats();
   reset_sbc_stats();
   reset_batt1_stats();
   reset_batt2_stats();

if individual timeout is greater than stat_timeout, reset stats

2. LCD functions
   Display stats
   Display heartbeat

4. Send ASV Batt value to Frisky
   1. read lower batt value
   2. convert to DAC, and send

5. Receive OCS radio control
   1. Check for START_BYTE

3. Read Frisky control values
   1. Get control mode
   2. Get RSSI (signal strength)
   3. Get direction
      for forward, side and yaw, rc.get_ch

6. Get control mode
   1. If SBC or POKB not alive, set to manual RC, if not, manual OCS.
   2. If RC & OCS down, set to station keep
   3. If control mode not autonomous, set to rc
   4. If ocs alive, not autonomous, set to ocs
   5. If not, set autonomous

Some weird stuff going on with the control modes here

Control Architecture

Frisky	OCS	Kill	State
x	x	Kill	kill
Manual	x	Not kill	Manual RC
Station keep	x	Not kill	Station Keep
Autonomous / Timeout	Manual	Not kill	Manual OCS
Autonomous	Autonomous / Timeout	Not kill	Autonomous
Timeout	Autonomous	Not kill	Autonomous

State	From	CAN TOPIC
Manual RC	RC (telemetry)	CAN_manual_thruster
Manual OCS	OCS (telemetry)	CAN_manual_thruster
Station Keep	SBC	CAN_thruster
Autonomous	SBC	CAN_thruster

For state, highest priority = Frisky, then OCS

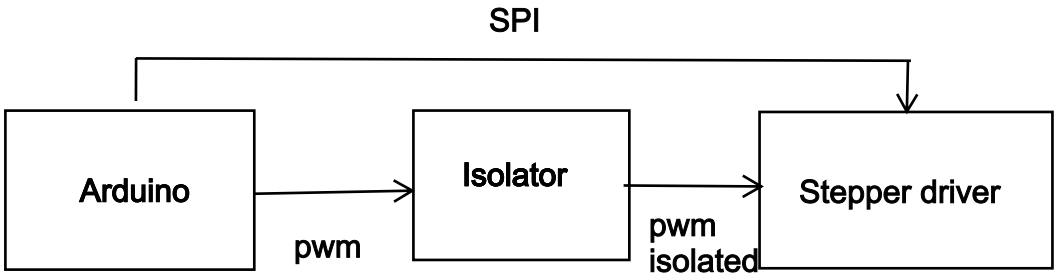
In event of failure

SBC	POKB	POSB	Frisky	OCS	Kill	State
x	x	x	x	x	Kill/Timeout	kill
x	x	Timeout	x	x	x	kill
Timeout	x	Alive	Not Timeout	x	Not kill	Manual RC
Timeout	x	Alive	Timeout	x	Not kill	Manual OCS
x	Timeout	Alive	Not Timeout	x	Not kill	Manual RC
x	Timeout	Alive	Timeout	x	Not kill	Manual OCS
Alive	Alive	Alive	Timeout	Timeout	Not kill	Station Keep
Timeout	x	Alive	Timeout	Timeout	Not kill	GG
x	Timeout	Alive	Timeout	Timeout	Not kill	GG

For failure, priority order
1. POSB
2. Frisky -> manual RC
3. OCS -> manual OCS
4. SBC, POKB -> Station keep

Send control information through CAN_control_link

RSSI: OCS (in 1dB)	RSSI: Frisky (in 1dB)	Control Mode: 0x01 - Autonomous 0x02 - Frisky (Manual) 0x03 - OCS (Manual) 0x04 - Station Keep


SPI
Arduino
pwm
Isolator
pwm
isolated
Stepper driver