```c
/**
 * @file init.c
 * @brief Perform initialization and start handler tasks
 * Copyright (C) 2017 Ethan Wells
 *
 * This program is free software: you can redistribute it and/or modify it
 * under the terms of the GNU General Public License as published by the Free
 * Software Foundation, either version 3 of the License, or(at your option) any
 * later version.
 *
 * This program is distributed in the hope that it will be useful, but WITHOUT
 * ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
 * FOR A PARTICULAR PURPOSE. See the GNU General Public License for more
 * details.
 *
 * You should have received a copy of the GNU General Public License along
 * with this program. If not, see <https://www.gnu.org/licenses/>
 */

#include "../include/robot.h"

static inline float lMogoRecalc(int p) {
        return p * 1.1;
} /* lMogoRecalc */

static inline float lineRecalc(int v) {
        return (float)(v > 16);
} /* lineRecalc */

void initializeIO() {
        watchdogInit();
} /* initializeIO */

/**
 * Notify both through the terminal and an lcd
 *
 * @param buffer the text to display
 */
void notice(const char *buffer) {
        #ifdef DEBUG_MODE
                print(buffer);
        #endif /* ifdef DEBUG_MODE */
        lcdSetText(uart1, 2, buffer);
        delay(5);
} /* notice */
```

```
void init() {
        // LCD initialization
        lcdInit(uart1);
        lcdSetBacklight(uart1, true);

        #ifdef DEBUG_MODE
                print("\nInitializing... ");
        #endif /* ifdef DEBUG_MODE */
        lcdSetText(uart1, 1, "Initializing...");

        // Set up the analog sensors
        gyro        = newGyro(1, true, 200);
        gyro.child  = new(Sensor);
        *gyro.child = newGyro(2, true, 195);
        notice("gyroscopes, ");
        Sensor *mogoAngle = new(Sensor);
        *mogoAngle        = newAnalog(3, true);
        mogoAngle->child  = new(Sensor);
        *mogoAngle->child = newAnalog(4, true);
        notice("mobile goal angle, ");
        Sensor *liftPot = new(Sensor);
        *liftPot = newAnalog(5, false);
        sensorRefresh(liftPot);
        liftPot->zero = liftPot->value;
        notice("lift pot, ");

        for (int i = 0; i < 3; i++) {
                line[i]          = newAnalog(i + 6, false);
                line[i].inverted = true;
                line[i].recalc   = &lineRecalc;
        }
        notice("line sensors");

        // Set up the digital sensors
        Sensor *driveCoder[2] = { new(Sensor), new(Sensor) };
        *driveCoder[0] = newQuad(4, 5, true);
        notice("left drive quad, ");
        *driveCoder[1] = newQuad(8, 9, true);
        notice("right drive quad, ");
        liftLimit[0] = newDigital(12, true);
        liftLimit[1] = newDigital(11, true);
        notice("lift limit switches, ");
        sonic  = new(Sensor);
        *sonic = newSonic(6, 7);
        notice("ultrasonic, ");
```

```
// The IMEs
if (imeInitializeAll() < 1) {
        print("\n\nexiting program...\n\n");
        exit(0);
}
Sensor *manipS = new(Sensor);
*manipS = newIME(0, true);
sensorReset(manipS);

// Initialize and set up all of the motors, servos, etc

// intake motor
intake = motorCreate(3, true);
notice("intake motor, ");

// intake manipulater motor
manip = motorCreate(4, true);
manip.child = new(Motor);
*manip.child = motorCreate(8, false);
manip.sensor = manipS;

// lift motors
lift                      = motorCreate(5,  true); // bottom left
lift.child                = new(Motor);
*lift.child               = motorCreate(6, false); // top left
lift.child->child         = new(Motor);
*lift.child->child        = motorCreate(7, false); // bottom right
lift.sensor               = liftPot;
notice("lift motors, ");

// mobile goal intake motors
mogo                 = motorCreate(1, false); // left
mogo.recalc          = &lMogoRecalc;
mogo.deadband        = 6;
mogo.child           = new(Motor);
*mogo.child          = motorCreate(10, true); // right
mogo.child->deadband = 6;
mogo.sensor          = mogoAngle;
notice("mobile goal motors, ");


// left drive motors
drive[0]        = motorCreate(2, true);
drive[0].sensor = driveCoder[0];

// right drive motors
```

```
        drive[1]        = motorCreate(9, false);
        drive[1].sensor = driveCoder[1];
        notice("drive motors, ");

        lcdSetText(uart1, 1, "Ready!");
        #ifdef DEBUG_MODE
                print("\n\n");
        #endif /* ifdef DEBUG_MODE */
        setTeamName("709S");
        notice("done!");

        // Start the LCD task
        LCDHandle = GO(lcdTask, NULL);
} /* init */
```