```
/*
 * Defines the functionality for the Homing class.
 *
 *  * Version 0.0  LWH created
 *
 * Copyright (c) 2022, LWH brainware. All rights reserved.
 */
#define DEBUG         0

#include <LWHlog.h>
#include "Homing.h"

/*****************************************************************
 * Creates a new Homer instance.
 * Sets the stepper to the given one.
 * Creates two light barriers at the given ports
 * Starts in "NONE" state
 */
Homer::Homer(TTStepper* astepper, int apincoarse, int apinfine) {
  theStepper    = astepper;
  theCoarse     = new GLS(apincoarse);
  theFine       = new GLS(apinfine);
  currentState  = homingState::NONE;
  debug("Homing set up for GLS at pins %i and %i\n", apincoarse, apinfine);
}

/*****************************************************************
 * Updates the light barriers and acts on changes.
 * must be called in the main loop
 */
void Homer::loop(void)  {
    static long homePosition = 0;
    // if not homing, nothing to do here
    if  (currentState == homingState::NONE) return;
    // if homing, update light barriers
    theCoarse->loop();
    theStepper->loop();
    theFine->loop();
    theStepper->loop();

    if  (currentState == homingState::FOUND) {
      //  if stopped after position found, null
      if (! theStepper->isRunning()) {
        currentState  = homingState::NONE;
        long  finalpos  =  theStepper->getCurrentPosition();
        debug("Finally stopped at %li\n", finalpos);
        theStepper->setCurrentPosition(finalpos - homePosition);
        theStepper->turnToPosition(0);
      }
      return;
    }
    if  ((currentState == homingState::CLW_C) || (currentState == homingState::CCLW_C)) {
      //  check the coarse lightbarrier
      if (currentState == homingState::CLW_C) {
        //  if moving clockwise and falling edge found, continue with fine search
//        if (theCoarse->isFree()) {
        if (theCoarse->isCovered()) {
          debug("Freeing edge of turntable found. Searching for motor flap...\n");
          currentState = homingState::CLW_F;
        }
      } else  {
        //  if moving counterclockwise and rising edge found, stop and search clockwise
//        if (theCoarse->isCovered())  {
        if (theCoarse->isFree())  {
          debug("Covering edge of turntable found. Reversing\n");
          currentState = homingState::CLW_C;
          theStepper->searchClw();
        }
      }
    } else  {
      //  check the fine lightbarrier
```

```
      //  this will always happen clockwise, so find the rising edge
      if (theFine->isCovered())  {
        homePosition  = theStepper->getCurrentPosition();
        currentState  = homingState::FOUND;
        debug("Motor flap found. Home position reached at %li\n", homePosition);
        theStepper->stop(false);
      }
    }


}

/*****************************************************************
 * Starts the homing process (if not already running)
 *   make sure flank direction is the same for both lightbarriers,
 *   so if covered, both will run until free.
 */
void Homer::findHome(void)  {
  switch (currentState) {
    case homingState::NONE:
      debug("Finding home position...\n");
      //  if covered, turn clockwise, else counterclockwise
      //  if (glsCoarse.isCovered())  {
      if (theCoarse->isFree())  {
        currentState  = homingState::CLW_C;
        theStepper->searchClw();
      } else  {
        currentState  = homingState::CCLW_C;
        theStepper->searchCClw();
      }
      break;
    default:
      debug("Already searching for home position!\n");
  } // end case
}

/*****************************************************************
 * returns the current homing state
 */
homingState Homer::getState(void) {
  return currentState;
}
/*****************************************************************
 * stops the motor and leaves the homing process
 */
void Homer::abort(void) {
  if (currentState == homingState::NONE) return;
  theStepper->stop(false);  // if not already done
  currentState  = homingState::NONE;
}
```