```
/*
* TI eQEP driver interface API
*
* Copyright (C) 2013 Nathaniel R. Lewis - http://nathanielrlewis.com/
*
* 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 2 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, write to the Free Software
* Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
*
*
* This code is changed by Jelle Spijker (C) 2014.
* Introducing polling with threading.
*
*/

#pragma once

#include <iostream>
#include <stdint.h>
#include <string>
#include "BBB.h"

#define eQEP0 "/sys/devices/ocp.3/48300000.epwmss/48300180.eqep"
#define eQEP1 "/sys/devices/ocp.3/48302000.epwmss/48302180.eqep"
#define eQEP2 "/sys/devices/ocp.3/48304000.epwmss/48304180.eqep"

namespace Hardware
{
    // Class which defines an interface to my eQEP driver
    class eQEP:
        public BBB
    {
        // Base path for the eQEP unit
        std::string path;
    public:
        // Modes of operation for the eQEP hardware
        typedef enum
        {
            // Absolute positioning mode
            eQEP_Mode_Absolute = 0,

            // Relative positioning mode
            eQEP_Mode_Relative = 1,

            // Error flag
            eQEP_Mode_Error = 2,
        } eQEP_Mode;

        // Default constructor for the eQEP interface driver
        eQEP(std::string _path, eQEP_Mode _mode);

        // Reset the value of the encoder
        void set_position(int32_t position);

        // Get the position of the encoder, pass poll as true to poll the pin, whereas passing false reads the immediate value
        int32_t get_position(bool _poll = true);

        // Thread of the poll
        int WaitForPositionChange(CallbackType callback);
        void WaitForPositionChangeCancel() { this->threadRunning = false; }

        // Set the polling period
        void set_period(uint64_t period);

        // Get the polling period of the encoder
        uint64_t get_period();

        // Set the mode of the eQEP hardware
        void set_mode(eQEP_Mode mode);

        // Get the mode of the eQEP hardware
        eQEP_Mode get_mode();

    private:
        friend void* threadedPolleqep(void *value);

    };

    void* threadedPolleqep(void *value);

}
```