```
//
    ================================================================================
    =========
/** @file adc.h
 *    This file contains a very simple A/D converter driver. The driver is
     hopefully
 *    thread safe in FreeRTOS due to the use of a mutex to prevent its use by
     multiple
 *    tasks at the same time. There is no protection from priority inversion,
     however,
 *    except for the priority elevation in the mutex.
 *
 *  Revisions:
 *    @li 01-15-2008 JRR Original (somewhat useful) file
 *    @li 10-11-2012 JRR Less original, more useful file with FreeRTOS mutex
     added
 *    @li 10-12-2012 JRR There was a bug in the mutex code, and it has been fixed
 *
 *  License:
 *    This file is copyright 2012 by JR Ridgely and released under the Lesser GNU
 *    Public License, version 2. It intended for educational use only, but its
     use
 *    is not limited thereto. */
/*    THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
 *    AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 *    IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 *    ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
 *    LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
     CONSEQUEN-
 *    TIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE
     GOODS
 *    OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
     HOWEVER
 *    CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
     LIABILITY,
 *    OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE
     USE
 *    OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. */
//
    ================================================================================
    =========

// This define prevents this .H file from being included multiple times in a .CPP
    file
#ifndef _AVR_ADC_H_
#define _AVR_ADC_H_

#include "emstream.h"                          // Header for serial ports and
    devices
#include "FreeRTOS.h"                          // Header for the FreeRTOS RTOS
#include "task.h"                              // Header for FreeRTOS task functions
#include "queue.h"                             // Header for FreeRTOS queues
#include "semphr.h"                            // Header for FreeRTOS semaphores


//--------------------------------------------------------------------------------
```

```
    ------
/** @brief   This class @b will run the A/D converter on an AVR processor.
 *  @details This header file declares our two main functions: read_once and
     read_oversampled.
 *           It also introduces an overloaded operator.
 */

class adc
{
    protected:
        /// The ADC class uses this pointer to the serial port to say hello
        emstream* ptr_to_serial;

    public:
        // The constructor sets up the A/D converter for use. The "= NULL" part
            is a
        // default parameter, meaning that if that parameter isn't given on the
            line
        // where this constructor is called, the compiler will just fill in
            "NULL".
        // In this case that has the effect of turning off diagnostic printouts
        adc (emstream* = NULL);

        // This function reads one channel once, returning the result as an
            unsigned
        // integer; it should be called from within a normal task, not an ISR
        uint16_t read_once (uint8_t);

        // This function reads the A/D lots of times and returns the average.
            Doing so
        // implements a crude sort of low-pass filtering that can help reduce
            noise
        uint16_t read_oversampled (uint8_t, uint8_t);

}; // end of class adc


// This operator prints the A/D converter (see file adc.cpp for details). It's
    not
// a part of class adc, but it operates on objects of class adc
emstream& operator << (emstream&, adc&);

#endif // _AVR_ADC_H_
```