```
/*****************************************************************************************************************
* This program is free software: you can redistribute it and/or modify it under the terms of the GNU Lesser General    *
* Public License as published by the Free Software Foundation, either version 3 of the License, or 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 Lesser General Public License for more  *
* details. You should have received a copy of the GNU Lesser General Public License along with this program.          *
* If not, see <http://www.gnu.org/licenses/> or <http://blog.robotica.eng.br/gnu-lesser-general-public-license/>.     *
*                                                                                                                    *
* © Copyright 2021 Filippo Pardini - filippo@robotica.eng.br - http://blog.robotica.eng.br/                          *
*****************************************************************************************************************
* Esta biblioteca facilita a utilização da biblioteca padrão <Wire.h> como master, já incluindo as novas facilidades   *
* ligadas ao Timeout (https://www.gitmemory.com/issue/arduino/Arduino/10803/703063770):                              *
* Wire.setWireTimeout(timeout,reset);                                                                                *
* bool Wire.getWireTimeoutFlag();                                                                                    *
* Wire.clearWireTimeoutFlag();                                                                                       *
* byte Wire.endTransmission(bool stop);                                                                              *
*****************************************************************************************************************/

#ifndef I2CRW_h
#define I2CRW_h

#include <Wire.h>

byte endereco;                                                  // Endereço I2C

//_____________________________________________________________________________________________________________________

void I2CRW_begin(byte enderecoI2C,long clock,long timeout,bool reset) // Inicializa a biblioteca
{
  // enderecoI2C => endereço (7bits) do slave
  // clock => frequência de operação I2C, normalmente 100K ou 400K
  // timeout => valor em microssegundos para timeout
  // reset => true - reset do hardware do I2C; false - não faz o reset

  endereco = enderecoI2C;                                       // Endereço I2C
  Wire.begin();                                                 // Inicializa biblioteca Wire padrão
  Wire.setClock((long)clock);                                   // Frequência de operação I2C
  Wire.setWireTimeout(timeout,reset);                           // Timeout I2C em microssegundos
                                                                // com liberação ou não do barramento
  Wire.clearWireTimeoutFlag();                                  // Limpa o flag de timeout
}

//_____________________________________________________________________________________________________________________

byte i2cWrite(byte endreg,byte *dados,byte quantidade,bool libera)    // Grava vários dados em registro
{
  // endreg => endereço do registrador
  // *dados => pointer de um vetor byte onde estão os valores a serem enviados
  // quantidade => número de bytes que são enviados
  // libera => libera ou não o barramento I2C: true - libera; false - não libera

  Wire.beginTransmission(endereco);                             // Inicia a transmissão
  Wire.write(endreg);                                           // Para o endereço do registro
  Wire.write(dados,quantidade);                                 // Transmite os bytes(quantidade) do vetor *dados
  byte codigo = Wire.endTransmission(libera);                   // Encerra a transmissão liberando ou não o
                                                                // barramento I2C, devolvendo o código de retorno:
                                                                // 0 => Sucesso
                                                                // 1 => Dados não cabem no buffer de transmissão
                                                                // 2 => Recebeu NAK na transmissão do endereço
                                                                // 3 => Recebeu NAK na transmissão dos dados
                                                                // 4 => Outros erros
                                                                // 5 => Timeout
  if (codigo)                                                   // Se código != 0 houve falha
  {
    Serial.print(F("\nFalha i2cWrite: "));                      // String na memória flash (F)
    Serial.print(codigo);                                       // Imprime o código
    if (codigo == 5)                                            // Se 5 foi timeout
    {
      Serial.print(F("\n***Timeout***"));                       // Imprime aviso
      Wire.clearWireTimeoutFlag();                              // Limpa o flag de timeout
    }
  }
  else;                                                         // Código = 0 => sucesso
  return codigo;
}

//_____________________________________________________________________________________________________________________

byte i2cWrite(byte endreg,byte dado,bool libera)                // Grava um dado(um byte) em registro
{
  // endreg => endereço do registrador
  // dado => valor a serem enviado
  // libera => libera ou não o barramento I2C: true - libera; false - não libera

  return i2cWrite(endreg,&dado,1,libera);                       // Retorna códigos como acima
}

```

```
//_____________________________________________________________________________________________________________

byte i2cRead(byte endreg,byte *dados,byte quantidade)                 // Lê dados de registro
{
  // endreg => endereço do registrador
  // *dados => pointer de um vetor byte onde para onde serão enviados os valores lidos
  // quantidade => número de bytes que são lidos

  Wire.beginTransmission(endereco);                                   // Inicia a transmissão
  Wire.write(endreg);                                                 // Para o endereço do registro
  byte codigo = Wire.endTransmission(false);                          // Encerra transmissão não liberando o barramento
                                                                      // e devolvendo o código de retorno:
                                                                      // 0 => Sucesso
                                                                      // 1 => Dados não cabem no buffer de transmissão
                                                                      // 2 => Recebeu NAK na transmissão do endereço
                                                                      // 3 => Recebeu NAK na transmissão dos dados
                                                                      // 4 => Outros erros
                                                                      // 5 => Timeout
  if (codigo)                                                         // Se codigo != 0 houve falha
  {
    Serial.print(F("\nFalha i2cRead: "));                             // String na memória flash (F)
    Serial.print(codigo);                                             // Imprime o código
    if (codigo == 5)                                                  // Se 5 foi timeout
    {
      Serial.print(F("\n***Timeout***"));                             // Imprime aviso
      Wire.clearWireTimeoutFlag();                                    // Limpa o flag de timeout
    }
    return codigo;                                                    // Retorna código
  }
  Wire.requestFrom(endereco,quantidade,(byte)true);                   // Após leitura libera o barramento I2C
  for (byte i = 0; i < quantidade; i++)                               // Executa as leituras
  {
    if (Wire.available()) dados[i] = Wire.read();                     // Se disponível faz a leitura de um byte
    else                                                              // Caso contrario
    {
      if (Wire.getWireTimeoutFlag())                                  // Verifica o flag de timeout
      {
        Wire.clearWireTimeoutFlag();                                  // Limpa o flag de timeout
        Serial.print("\n***Timeout***");                              // Imprime aviso
        return 5;                                                     // Código de erro de timeout
      }
    }
  }
  return 0;                                                           // Código = 0 => sucesso
}

//_____________________________________________________________________________________________________________

#endif
```