GCC Code Coverage Report

			Exec	Total	Coverage
Directory:	./				
Date:	2022-03-20 00:11:12	Lines:	6	25	24.0 %
Legend:	low: < 75.0 % medium: >= 75.0 % high: >= 90.0 %	Branches:	4	12	33.3 %

File	Lines			Branches	
include/byte_utils.hpp	🟩	100.0 %	2 / 2	- %	0 / 0
src/i2c_utils.cpp	□	0.0 %	0 / 17	- %	0 / 0
src/restricted_base.cpp	□	0.0 %	0 / 2	- %	0 / 0
tests/catch_main_app.cpp	🟩	100.0 %	4 / 4	33.3 %	4 / 12

Generated by: GCOVR (Version 4.2)

GCC Code Coverage Report

Directory:	./		Exec	Total	Coverage
Date:	2022-03-20 00:11:12	Lines:	6	25	24.0 %
Legend:	low: < 75.0 % medium: >= 75.0 % high: >= 90.0 %	Branches:	4	12	33.3 %

File	Lines			Branches	
include/byte_utils.hpp	🟩	100.0 %	2 / 2	- %	0 / 0
src/i2c_utils.cpp		0.0 %	0 / 17	- %	0 / 0
src/restricted_base.cpp		0.0 %	0 / 2	- %	0 / 0
tests/catch_main_app.cpp	🟩	100.0 %	4 / 4	33.3 %	4 / 12

Generated by: GCOVR (Version 4.2)

GCC Code Coverage Report

			Exec	Total	Coverage
Directory:	./				
File:	include/byte_utils.hpp	Lines:	2	2	100.0 %
Date:	2022-03-20 00:11:12	Branches:	0	0	- %

```
Line Branch  Exec  Source
   1               // MIT License
   2
   3               // Copyright (c) 2022 Chris Sutton
   4
   5               // Permission is hereby granted, free of charge, to any person obtaining a copy
   6               // of this software and associated documentation files (the "Software"), to deal
   7               // in the Software without restriction, including without limitation the rights
   8               // to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   9               // copies of the Software, and to permit persons to whom the Software is
  10               // furnished to do so, subject to the following conditions:
  11
  12               // The above copyright notice and this permission notice shall be included in all
  13               // copies or substantial portions of the Software.
  14
  15               // THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
  16               // IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
  17               // FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
  18               // AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
  19               // LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
  20               // OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
  21               // SOFTWARE.
  22
  23               #ifndef __BYTE_UTILS_HPP__
  24               #define __BYTE_UTILS_HPP__
  25
  26               #include <stdint.h>
  27
  28
  29               namespace noarch::byte_manip
  30               {
  31
  32               template<std::size_t BYTE_ARRAY_SIZE>
  33                  1  void print_bytes(std::array<uint8_t, BYTE_ARRAY_SIZE> &bytes [[maybe_unused]])
  34               {
  35                   #ifdef USE_RTT
  36                       for (uint16_t idx = 0; idx < bytes.size(); idx++)
  37                       {
  38
  39                               if (idx % 16 == 0)
  40                                   SEGGER_RTT_printf(0, "\n");
  41
  42                               SEGGER_RTT_printf(0, "0x%02x ", +bytes[idx]);
  43
  44                       }
  45                       SEGGER_RTT_printf(0, "\n");
  46                   #endif
  47
  48                  1  }
  49
  50               }   // namespace noarch::byte_manip
  51
  52               #endif // __BYTE_UTILS_HPP__
```

Generated by: GCOVR (Version 4.2)

GCC Code Coverage Report

			Exec	Total	Coverage
Directory:	./				
File:	src/i2c_utils.cpp	Lines:	0	17	0.0 %
Date:	2022-03-20 00:11:12	Branches:	0	0	- %

Line Branch Exec Source

```
// MIT License

// Copyright (c) 2022 Chris Sutton

// Permission is hereby granted, free of charge, to any person obtaining a copy
// of this software and associated documentation files (the "Software"), to deal
// in the Software without restriction, including without limitation the rights
// to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
// copies of the Software, and to permit persons to whom the Software is
// furnished to do so, subject to the following conditions:

// The above copyright notice and this permission notice shall be included in all
// copies or substantial portions of the Software.

// THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
// IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
// FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
// AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
// LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
// OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
// SOFTWARE.

#include <i2c_utils.hpp>
#include <timer_manager.hpp>

namespace stm32::i2c
{

// we can't unit test this without mocking
Status send_addr(I2C_TypeDef* i2c_handle [[maybe_unused]], uint8_t addr [[maybe_unused]], MsgType type [[maybe_unused]])
{
#if not defined(X86_UNIT_TESTING_ONLY)

 // Set the master to operate in 7-bit addressing mode. Clear ADD10 bit[11]
 i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_ADD10);

 // Set the address for the slave device. Set SADD bits[7:1].
 // The bits SADD[9],SADD[8] and SADD[0] are don't care.
 i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_SADD);
 i2c_handle->CR2 = i2c_handle->CR2 | (addr << 0);

 if (type == MsgType::PROBE) // generate START with AUTO-END enabled
 {
  // Master requests a write transfer
  i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_RD_WRN);

  // Enable AUTOEND Mode. A STOP condition is automatically sent when NBYTES data are transferred.
  i2c_handle->CR2 = i2c_handle->CR2 | I2C_CR2_AUTOEND;

 }
 else if (type == MsgType::WRITE) // generate START with AUTO-END disabled
 {
  // Master requests a write transfer
  i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_RD_WRN);

  // Disable RELOAD Mode. The transfer is completed after the NBYTES data transfer (STOP or RESTART follows).
  i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_RELOAD);

  // Disable AUTOEND Mode. TC flag is set when NBYTES data are transferred, stretching SCL low.
  i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_AUTOEND);
 }
 else if (type == MsgType::READ) // generate REPEATED START
 {

  // Master requests a read transfer
  i2c_handle->CR2 = i2c_handle->CR2 | (I2C_CR2_RD_WRN);

  // Disable RELOAD Mode. The transfer is completed after the NBYTES data transfer (STOP or RESTART follows).
  i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_RELOAD);

  // Disable AUTOEND Mode. TC flag is set when NBYTES data are transferred, stretching SCL low.
  i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_AUTOEND);
 }

 // Generate the restart/start condition
 generate_start_condition(i2c_handle);

 // give slave a chance to respond
 stm32::TimerManager::delay_microsecond(1000);

 // check if addr was not recognised by slave device
 if ( (i2c_handle->ISR & I2C_ISR_NACKF) == I2C_ISR_NACKF )
 {
  return Status::NACK;
 }
#endif
 // otherwise slave device is happy
 return Status::ACK;

}



Status receive_byte(I2C_TypeDef* i2c_handle [[maybe_unused]], uint8_t &rx_byte [[maybe_unused]])
{
#if not defined(X86_UNIT_TESTING_ONLY)
 rx_byte = i2c_handle->RXDR & I2C_RXDR_RXDATA;
#endif
 return Status::ACK;

}


Status send_byte(I2C_TypeDef* i2c_handle [[maybe_unused]], uint8_t tx_byte [[maybe_unused]])
{
#if not defined(X86_UNIT_TESTING_ONLY)
 i2c_handle->TXDR = tx_byte;

 // wait for TX FIFO to be transmitted before continuing
 while (((i2c_handle->ISR & I2C_ISR_TXE) == I2C_ISR_TXE) == false)
 {
  // do nothing
  stm32::TimerManager::delay_microsecond(10);
 }
 // check if slave device responded with NACK
 if (((i2c_handle->ISR & I2C_ISR_NACKF) == I2C_ISR_NACKF) == true)
 {
  return Status::NACK;
 }
#endif
 return Status::ACK;
}

void generate_stop_condition(I2C_TypeDef* i2c_handle)
{
 i2c_handle->CR2 = i2c_handle->CR2 | (I2C_CR2_STOP);
}

void generate_start_condition(I2C_TypeDef* i2c_handle)
{
 i2c_handle->CR2 = i2c_handle->CR2 | (I2C_CR2_START);
}

void set_numbytes(I2C_TypeDef* i2c_handle, uint32_t nbytes)
{
 i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_NBYTES);
 i2c_handle->CR2 = i2c_handle->CR2 | (nbytes << I2C_CR2_NBYTES_Pos);
}

void send_ack(I2C_TypeDef* i2c_handle)
{
 i2c_handle->CR2 = i2c_handle->CR2 & ~(I2C_CR2_NACK);
}

void send_nack(I2C_TypeDef* i2c_handle)
{
 i2c_handle->CR2 = i2c_handle->CR2 | (I2C_CR2_NACK);
}

} // namespace stm32::i2c
```

Generated by: GCOVR (Version 4.2)

GCC Code Coverage Report

			Exec	Total	Coverage
Directory:	./				
File:	src/restricted_base.cpp	Lines:	0	2	0.0 %
Date:	2022-03-20 00:11:12	Branches:	0	0	- %

Line Branch Exec Source

```
// MIT License

// Copyright (c) 2022 Chris Sutton

// Permission is hereby granted, free of charge, to any person obtaining a copy
// of this software and associated documentation files (the "Software"), to deal
// in the Software without restriction, including without limitation the rights
// to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
// copies of the Software, and to permit persons to whom the Software is
// furnished to do so, subject to the following conditions:

// The above copyright notice and this permission notice shall be included in all
// copies or substantial portions of the Software.

// THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
// IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
// FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
// AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
// LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
// OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
// SOFTWARE.

#include <restricted_base.hpp>

void invalid_allocation_error_handler()
{
    while(true)
    {
        //
    }
}




// void* RestrictedBase::operator new(size_t size [[maybe_unused]]) noexcept
// {
//     while(true)
//     {
//         // forbidden
//     }
//     // just to prevent compiler errors
//     void *p;
//     return p;

// }

// void RestrictedBase::operator delete(void* ptr) noexcept
// {
//     while(true)
//     {
//         // forbidden
//     }

// }
```

GCC Code Coverage Report

			Exec	Total	Coverage
Directory:	./				
File:	tests/catch_main_app.cpp	Lines:	4	4	100.0 %
Date:	2022-03-20 00:11:12	Branches:	4	12	33.3 %

```
 Line Branch  Exec  Source
    1
    2                #include <catch2/catch_all.hpp>
    3
    4                #include <byte_utils.hpp>
    5
    6                1  TEST_CASE("Test Embedded Utils", "[embedded_utils]")
    7                {
    8                    std::array<uint8_t, 8> bytes;
    9                1      noarch::byte_manip::print_bytes(bytes);
       ✓✗✓✗
   10  ✓✗✓✗       1      REQUIRE(true);
       ✗✗✗✗
   11                1  }
```

Generated by: GCOVR (Version 4.2)