GCC Code Coverage Report

			Exec	Total	Coverage
Directory:	./				
File:	storage/blockdevice/source/SlicingBlockDevice.cpp	Lines:	60	61	98.4 %
Date:	2021-05-06 12:39:05	Branches:	29	34	85.3 %

Line	Branch	Exec	Source
1			/* mbed Microcontroller Library
2			* Copyright (c) 2017 ARM Limited
3			* SPDX-License-Identifier: Apache-2.0
4			*
5			* Licensed under the Apache License, Version 2.0 (the "License");
6			* you may not use this file except in compliance with the License.
7			* You may obtain a copy of the License at
8			*
9			* http://www.apache.org/licenses/LICENSE-2.0
10			*
11			* Unless required by applicable law or agreed to in writing, software
12			* distributed under the License is distributed on an "AS IS" BASIS,
13			* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
14			* See the License for the specific language governing permissions and
15			* limitations under the License.
16			*/
17			
18			#include "blockdevice/SlicingBlockDevice.h"
19			#include "platform/mbed_assert.h"
20			#include "stddef.h"
21			#include <stdio.h>
22			
23			namespace mbed {
24			
25			
26		10	SlicingBlockDevice::SlicingBlockDevice(BlockDevice *bd, bd_addr_t start, bd_addr_t stop)
27			: _bd(bd)
28			, _start_from_end(false), _start(start)
29		10	, _stop_from_end(false), _stop(stop)
30			{
31	✗✓	10	MBED_ASSERT(bd);

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
32          //     // SlicingBlockDevice(bd, 0,0) would use the full block, which does not make sense, it must be a programming eror.
33          //     // SlicingBlockDevice(bd, 100,100) would have no size, which is also a programming error.
34    ✗✓ 10     MBED_ASSERT(start != stop);
35
36    ✓✓ 10     if ((int64_t)_start < 0) {
37        1         _start_from_end = true;
38        1         _start = -_start;
39                  }
40
41    ✓✓ 10     if ((int64_t)_stop <= 0) {
42        1         _stop_from_end = true;
43        1         _stop = -_stop;
44                  }
45       10   }
46
47        6   int SlicingBlockDevice::init()
48            {
49        6       int err = _bd->init();
50    ✗✓  6       if (err) {
51                      return err;
52                  }
53
54        6       bd_size_t size = _bd->size();
55
56                // Calculate from_end values
57    ✓✓  6       if (_start_from_end) {
58        1           _start_from_end = false;
59        1           _start = size - _start;
60                  }
61
62    ✓✓  6       if (_stop_from_end) {
63        1           _stop_from_end = false;
64        1           _stop = size - _stop;
65                  }
66
67                  // Check that block addresses are valid
68    ✓✓  6       if (!is_valid_erase(0, _stop - _start)) {
69        1           return BD_ERROR_DEVICE_ERROR;
70                  }
71
72        5       return 0;
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
73                  }
74
75      1           int SlicingBlockDevice::deinit()
76                  {
77      1               return _bd->deinit();
78                  }
79
80      1           int SlicingBlockDevice::sync()
81                  {
82      1               return _bd->sync();
83                  }
84
85      2           int SlicingBlockDevice::read(void *b, bd_addr_t addr, bd_size_t size)
86                  {
87    ✓✓    2           if (!is_valid_read(addr, size)) {
88      1                 return BD_ERROR_DEVICE_ERROR;
89                      }
90      1               return _bd->read(b, addr + _start, size);
91                  }
92
93      7           int SlicingBlockDevice::program(const void *b, bd_addr_t addr, bd_size_t size)
94                  {
95    ✓✓    7           if (!is_valid_program(addr, size)) {
96      1                 return BD_ERROR_DEVICE_ERROR;
97                      }
98      6               return _bd->program(b, addr + _start, size);
99                  }
100
101     2           int SlicingBlockDevice::erase(bd_addr_t addr, bd_size_t size)
102                 {
103   ✓✓    2           if (!is_valid_erase(addr, size)) {
104     1                 return BD_ERROR_DEVICE_ERROR;
105                     }
106     1               return _bd->erase(addr + _start, size);
107                 }
108
109     2           bool SlicingBlockDevice::is_valid_read(bd_addr_t addr, bd_size_t size) const
110                 {
111   ✓✗✓✓  2           return _bd->is_valid_read(_start + addr, size) && _start + addr + size <= _stop;
112                 }
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
113
114      7  bool SlicingBlockDevice::is_valid_program(bd_addr_t addr, bd_size_t size) const
115         {
116  ✓✗✓✓   7      return _bd->is_valid_program(_start + addr, size) && _start + addr + size <= _stop;
117         }
118
119      8  bool SlicingBlockDevice::is_valid_erase(bd_addr_t addr, bd_size_t size) const
120         {
121  ✓✓✓✓   8      return _bd->is_valid_erase(_start + addr, size) && _start + addr + size <= _stop;
122         }
123
124      1  bd_size_t SlicingBlockDevice::get_read_size() const
125         {
126      1      return _bd->get_read_size();
127         }
128
129      1  bd_size_t SlicingBlockDevice::get_program_size() const
130         {
131      1      return _bd->get_program_size();
132         }
133
134      1  bd_size_t SlicingBlockDevice::get_erase_size() const
135         {
136      1      return _bd->get_erase_size(_start);
137         }
138
139      1  bd_size_t SlicingBlockDevice::get_erase_size(bd_addr_t addr) const
140         {
141      1      return _bd->get_erase_size(_start + addr);
142         }
143
144      1  int SlicingBlockDevice::get_erase_value() const
145         {
146      1      return _bd->get_erase_value();
147         }
148
149      2  bd_size_t SlicingBlockDevice::size() const
150         {
151      2      return _stop - _start;
152         }
153
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
154          1  const char *SlicingBlockDevice::get_type() const
155             {
156          1      return _bd->get_type();
157             }
158
159             } // namespace mbed
```


Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD