Class B Library Help

Table of Contents

1 Introduction

Introduction

This document describes the Class B Safety Software Library routines that detect the occurrence of Faults in a single channel CPU. These routines have been developed in accordance with the IEC 60730 standard to support the Class B certification process. These routines can be directly integrated with the end user’s application to test and verify the critical functionalities of a controller without affecting the end user’s application. The Class B safety software routines can be called periodically at start-up or run time to test the following components:

• CPU Registers

• CPU Program Counter

• Invariable Memory

• Variable Memory

• Clock

2 Software License Agreement

MICROCHIP IS WILLING TO LICENSE THE ACCOMPANYING SOFTWARE AND DOCUMENTATION TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE FOLLOWING TERMS. TO ACCEPT THE TERMS OF THIS LICENSE, CLICK "I ACCEPT" AND PROCEED WITH THE DOWNLOAD OR INSTALL. IF YOU DO NOT ACCEPT THESE LICENSE TERMS, CLICK "I DO NOT ACCEPT," AND DO NOT DOWNLOAD OR INSTALL THIS SOFTWARE.

NON-EXCLUSIVE SOFTWARE LICENSE AGREEMENT

This Nonexclusive Software License Agreement ("Agreement") is a contract between you, your heirs, successors and assigns ("Licensee") and Microchip Technology Incorporated, a Delaware corporation, with a principal place of business at 2355 W. Chandler Blvd., Chandler, AZ 85224-6199, and its subsidiary, Microchip Technology (Barbados) II Incorporated (collectively, "Microchip") for the accompanying Microchip software including, but not limited to, Graphics Library Software, IrDA Stack Software, MCHPFSUSB Stack Software, Memory Disk Drive File System Software, mTouch(TM) Capacitive Library Software, Smart Card Library Software, TCP/IP Stack Software, MiWi(TM) DE Software, Security Package Software, and/or any PC programs and any updates thereto (collectively, the "Software"), and accompanying documentation, including images and any other graphic resources provided by Microchip ("Documentation").

1. Definitions. As used in this Agreement, the following capitalized terms will have the meanings defined below:

a. "Microchip Products" means Microchip microcontrollers and Microchip digital signal controllers.

b. "Licensee Products" means Licensee products that use or incorporate Microchip Products.

c. "Object Code" means the Software computer programming code that is in binary form (including related documentation, if any), and error corrections, improvements, modifications, and updates.

d. "Source Code" means the Software computer programming code that may be printed out or displayed in human readable form (including related programmer comments and documentation, if any), and error corrections, improvements, modifications, and updates.

e. "Third Party" means Licensee’s agents, representatives, consultants, clients, customers, or contract manufacturers.

f. "Third Party Products" means Third Party products that use or incorporate Microchip Products.

2. Software License Grant. Microchip grants strictly to Licensee a non-exclusive, non-transferable, worldwide license to:

a. use the Software in connection with Licensee Products and/or Third Party Products;

b. if Source Code is provided, modify the Software; provided that Licensee clearly notifies Third Parties regarding the source of such modifications;

c. distribute the Software to Third Parties for use in Third Party Products, so long as such Third Party agrees to be bound by this Agreement (in writing or by "click to accept") and this Agreement accompanies such distribution;

d. sublicense to a Third Party to use the Software, so long as such Third Party agrees to be bound by this Agreement (in writing or by "click to accept");

e. with respect to the TCP/IP Stack Software, Licensee may port the ENC28J60.c, ENC28J60.h, ENCX24J600.c, and ENCX24J600.h driver source files to a non-Microchip Product used in conjunction with a Microchip ethernet controller;

f. with respect to the MiWi (TM) DE Software, Licensee may only exercise its rights when the Software is embedded on a Microchip Product and used with a Microchip radio frequency transceiver or UBEC UZ2400 radio frequency transceiver which are integrated into Licensee Products or Third Party Products.

For purposes of clarity, Licensee may NOT embed the Software on a non-Microchip Product, except as described in this

Section.

3. Documentation License Grant. Microchip grants strictly to Licensee a non-exclusive, non-transferable, worldwide license to use the Documentation in support of Licensee's authorized use of the Software

4. Third Party Requirements. Licensee acknowledges that it is Licensee’s responsibility to comply with any third party license terms or requirements applicable to the use of such third party software, specifications, systems, or tools. This includes, by way of example but not as a limitation, any standards setting organizations requirements and, particularly with respect to the Security Package Software, local encryption laws and requirements. Microchip is not responsible and will not be held responsible in any manner for Licensee’s failure to comply with such applicable terms or requirements.

5. Open Source Components. Notwithstanding the license grant in Section 1 above, Licensee further acknowledges that certain components of the Software may be covered by so-called "open source" software licenses ("Open Source Components"). Open Source Components means any software licenses approved as open source licenses by the Open Source Initiative or any substantially similar licenses, including without limitation any license that, as a condition of distribution of the software licensed under such license, requires that the distributor make the software available in source code format. To the extent required by the licenses covering Open Source Components, the terms of such license will apply in lieu of the terms of this Agreement. To the extent the terms of the licenses applicable to Open Source Components prohibit any of the restrictions in this Agreement with respect to such Open Source Components, such restrictions will not apply to such Open Source Component.

6. Licensee Obligations. Licensee will not: (a) engage in unauthorized use, modification, disclosure or distribution of Software or Documentation, or its derivatives; (b) use all or any portion of the Software, Documentation, or its derivatives except in conjunction with Microchip Products, Licensee Products or Third Party Products; or (c) reverse engineer (by disassembly, decompilation or otherwise) Software or any portion thereof. Licensee may not remove or alter any Microchip copyright or other proprietary rights notice posted in any portion of the Software or Documentation. Licensee will defend, indemnify and hold Microchip and its subsidiaries harmless from and against any and all claims, costs, damages, expenses (including reasonable attorney's fees), liabilities, and losses, including without limitation: (x) any claims directly or indirectly arising from or related to the use, modification, disclosure or distribution of the Software, Documentation, or any intellectual property rights related thereto; (y) the use, sale and distribution of Licensee Products or Third Party Products; and (z) breach of this Agreement.

7. Confidentiality. Licensee agrees that the Software (including but not limited to the Source Code, Object Code and library files) and its derivatives, Documentation and underlying inventions, algorithms, know-how and ideas relating to the Software and the Documentation are proprietary information belonging to Microchip and its licensors ("Proprietary Information"). Except as expressly and unambiguously allowed herein, Licensee will hold in confidence and not use or disclose any Proprietary Information and will similarly bind its employees and Third Party(ies) in writing. Proprietary Information will not include information that: (i) is in or enters the public domain without breach of this Agreement and through no fault of the receiving party; (ii) the receiving party was legally in possession of prior to receiving it; (iii) the receiving party can demonstrate was developed by the receiving party independently and without use of or reference to the disclosing party's Proprietary Information; or (iv) the receiving party receives from a third party without restriction on disclosure. If Licensee is required to disclose Proprietary Information by law, court order, or government agency, License will give Microchip prompt notice of such requirement in order to allow Microchip to object or limit such disclosure. Licensee agrees that the provisions of this Agreement regarding unauthorized use and nondisclosure of the Software, Documentation and related Proprietary Rights are necessary to protect the legitimate business interests of Microchip and its licensors and that monetary damage alone cannot adequately compensate Microchip or its licensors if such provisions are violated. Licensee, therefore, agrees that if Microchip alleges that Licensee or Third Party has breached or violated such provision then Microchip will have the right to injunctive relief, without the requirement for the posting of a bond, in addition to all other remedies at law or in equity.

8. Ownership of Proprietary Rights. Microchip and its licensors retain all right, title and interest in and to the Software and Documentation including, but not limited to all patent, copyright, trade secret and other intellectual property rights in the Software, Documentation, and underlying technology and all copies and derivative works thereof (by whomever produced). Licensee and Third Party use of such modifications and derivatives is limited to the license rights described in this Agreement.

9. Termination of Agreement. Without prejudice to any other rights, this Agreement terminates immediately, without notice by Microchip, upon a failure by Licensee or Third Party to comply with any provision of this Agreement. Upon termination, Licensee and Third Party will immediately stop using the Software, Documentation, and derivatives thereof, and immediately

destroy all such copies.

10. Warranty Disclaimers. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. MICROCHIP AND ITS LICENSORS ASSUME NO RESPONSIBILITY FOR THE ACCURACY, RELIABILITY OR APPLICATION OF THE SOFTWARE OR DOCUMENTATION. MICROCHIP AND ITS LICENSORS DO NOT WARRANT THAT THE SOFTWARE WILL MEET REQUIREMENTS OF LICENSEE OR THIRD PARTY, BE UNINTERRUPTED OR ERROR-FREE. MICROCHIP AND ITS LICENSORS HAVE NO OBLIGATION TO CORRECT ANY DEFECTS IN THE SOFTWARE.

11. Limited Liability. IN NO EVENT WILL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER ANY LEGAL OR EQUITABLE THEORY FOR ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES INCLUDING BUT NOT LIMITED TO INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS. The aggregate and cumulative liability of Microchip and its licensors for damages hereunder will in no event exceed $1000 or the amount Licensee paid Microchip for the Software and Documentation, whichever is greater. Licensee acknowledges that the foregoing limitations are reasonable and an essential part of this Agreement.

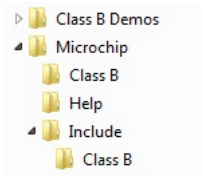
12. General. THIS AGREEMENT WILL BE GOVERNED BY AND CONSTRUED UNDER THE LAWS OF THE STATE OF ARIZONA AND THE UNITED STATES WITHOUT REGARD TO CONFLICTS OF LAWS PROVISIONS. Licensee agrees that any disputes arising out of or related to this Agreement, Software or Documentation will be brought exclusively in either the U.S. District Court for the District of Arizona, Phoenix Division, or the Superior Court of Arizona located in Maricopa County, Arizona. This Agreement will constitute the entire agreement between the parties with respect to the subject matter hereof. It will not be modified except by a written agreement signed by an authorized representative of Microchip. If any provision of this Agreement will be held by a court of competent jurisdiction to be illegal, invalid or unenforceable, that provision will be limited or eliminated to the minimum extent necessary so that this Agreement will otherwise remain in full force and effect and enforceable. No waiver of any breach of any provision of this Agreement will constitute a waiver of any prior, concurrent or subsequent breach of the same or any other provisions hereof, and no waiver will be effective unless made in writing and signed by an authorized representative of the waiving party. Licensee agrees to comply with all import and export laws and restrictions and regulations of the Department of Commerce or other United States or foreign agency or authority. The indemnities, obligations of confidentiality, and limitations on liability described herein, and any right of action for breach of this Agreement prior to termination, will survive any termination of this Agreement. Any prohibited assignment will be null and void. Use, duplication or disclosure by the United States Government is subject to restrictions set forth in subparagraphs (a) through (d) of the Commercial Computer-Restricted Rights clause of FAR 52.227-19 when applicable, or in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, and in similar clauses in the NASA FAR Supplement. Contractor/manufacturer is Microchip Technology Inc., 2355 W. Chandler Blvd., Chandler, AZ 85224-6199.

If Licensee has any questions about this Agreement, please write to Microchip Technology Inc., 2355 W. Chandler Blvd., Chandler, AZ 85224-6199 USA. ATTN: Marketing.

License Rev. No. 05-012412

3 Getting Started

The folder structure of Class B Software Library is shown below:



The library specific folders are the following:

• The ...\Class B Demo folder will contain the library demo projects.

• The ...\Microchip folder will contain the library components.

• The ...\Help sub-folder under ..\Microchip folder will contain this document (Class B Library Help.chm file).

• The ...\Class B sub-folder under the ..\Microchip folder is where the C files related to the Class B library are located.

• The ...\Class B sub-folder under the Include folder is where the Header files related to the Class B library are located.

3.1 Class B Library Files

The following files should be included in the project:

Common	
classb.h	This file joins all definitions, macros and functions prototypes related to Class B library. To use the library API only this header can be included in the application code.
classb_config.h	Class B library configurations.
RAM Tests	
classb_marchc_.s	March C and March C Minus test routines.
classb_marchb_.s	March B test routines.
classb_checkerboard_.s	Checker Board test routines.
classb_startup_.s	Contains the code to run the entire RAM march tests before C startup code.
CRC Checksum Calculation	
classb_crc.c	CRC checksum calculation routines for Flash and EEPROM.
MCU Registers Tests	
classb_registers_.s	MCU registers test routines.
classb_pc_.s	MCU program counter test routines.
Clock Tests	
classb_clock.c, classb_clock_.s	Clock source test routines.

3.2 Class B Library Configuration

The following Class B Library settings should be defined in classb_config.h file:

CLASSB_MARCH_C_STARTUP	Startup march C test enable flag. If this option is not zero, entire RAM will be tested using the march C test before main() function entry.
CLASSB_MARCH_C_MINUS_STARTUP	Startup march C minus test enable flag. If this option is not zero, entire RAM will be tested using the march C minus test before main() function entry.
CLASSB_MARCH_B_STARTUP	Startup march B test enable flag. If this option is not zero, entire RAM will be tested using the march C test before main() function entry.
CLASSB_DEVICE_FLASH_SIZE_KB	Device flash memory size in KBytes. This option is used by the program counter test.
CLASSB_CLOCK_TEST_TIME_MS	Clock test time in mS. Can be about 20 mS if clock is between 1-32MHz and the reference clock is 50Hz-33kHz.
CLASSB_CLOCK_TEST_TIMER_ADDRESS	Clock test hardware timer SFR address. The timer is used to count the reference clock pulses.

3.3 Using API

The Class B Library doesn't require modifications of the linker script (.gld file). To use the library functions one header file "classb.h" must be included in the application source file. Also the library configuration "classb_config.h" file must exist in the application project. The interrupts are disabled during the march C, march B and clock tests. The memory regions (start address and length) can be retrieved form the map file ("Generate map file" option in the "Diagnostic" group of the linker settings must be enabled). Or the basic memory map information can be found in "Build" window if the "Display memory usage" option is selected in "Diagnostic" group of the linker settings. The following code explains the Class B library test functions usage:

```
-----------------------------------------------------------------------------------------------------------------
// Check for startup march tests error.
// These tests are run before main() function entry if CLASSB_MARCH_C_STARTUP,
// CLASSB_MARCH_C_MINUS_STARTUP or/and CLASSB_MARCH_B_STARTUP flags are set
// in classb_config.h file.
if(CLASSB_StartupMarchTestGetResult())
{
// ERROR IS DETECTED
while(1);
}
-----------------------------------------------------------------------------------------------------------------
// CPU Registers test.
```

```
if (CLASSB_CPURegistersTest())
{
// ERROR IS DETECTED
while (1);
}
-------------------------------------------------------------------------------------------------------------
// CPU Program Counter test.
// Should be used with a watchdog timer.
// Check for a watchdog timer timeout error.
if(RCONbits.WDTO == 1)
{
// ERROR IS DETECTED
while(1);
}
// Check program counter status everywhere using CLASSB_CPUPCTestGetResult()
// especially before critical code execution.
// !!! Interrupts are disabled during this test.!!!
if(CLASSB_CPUPCTestGetResult())
{
// ERROR IS DETECTED
while(1);
}
// Run PC test.
CLASSB_CPUPCTest();
// Clear watchdog.
ClrWdt();
-------------------------------------------------------------------------------------------------------------
// Checker board test.
// RAM tests memory of the .nbss section
// The linker has generated the following map information:
// Data Memory [Origin = 0x800, Length = 0x7800]
//
// section address alignment gaps total length (dec)
// ------- ------- -------------- --------------------
// .nbss 0x800 0 0xfa2 (4002)
//
// Total data memory used (bytes): 0xfa2 (4002) 13%
if (CLASSB_RAMCheckerboardTest(0x0800, // start address (must be aligned by word (2 bytes))
```

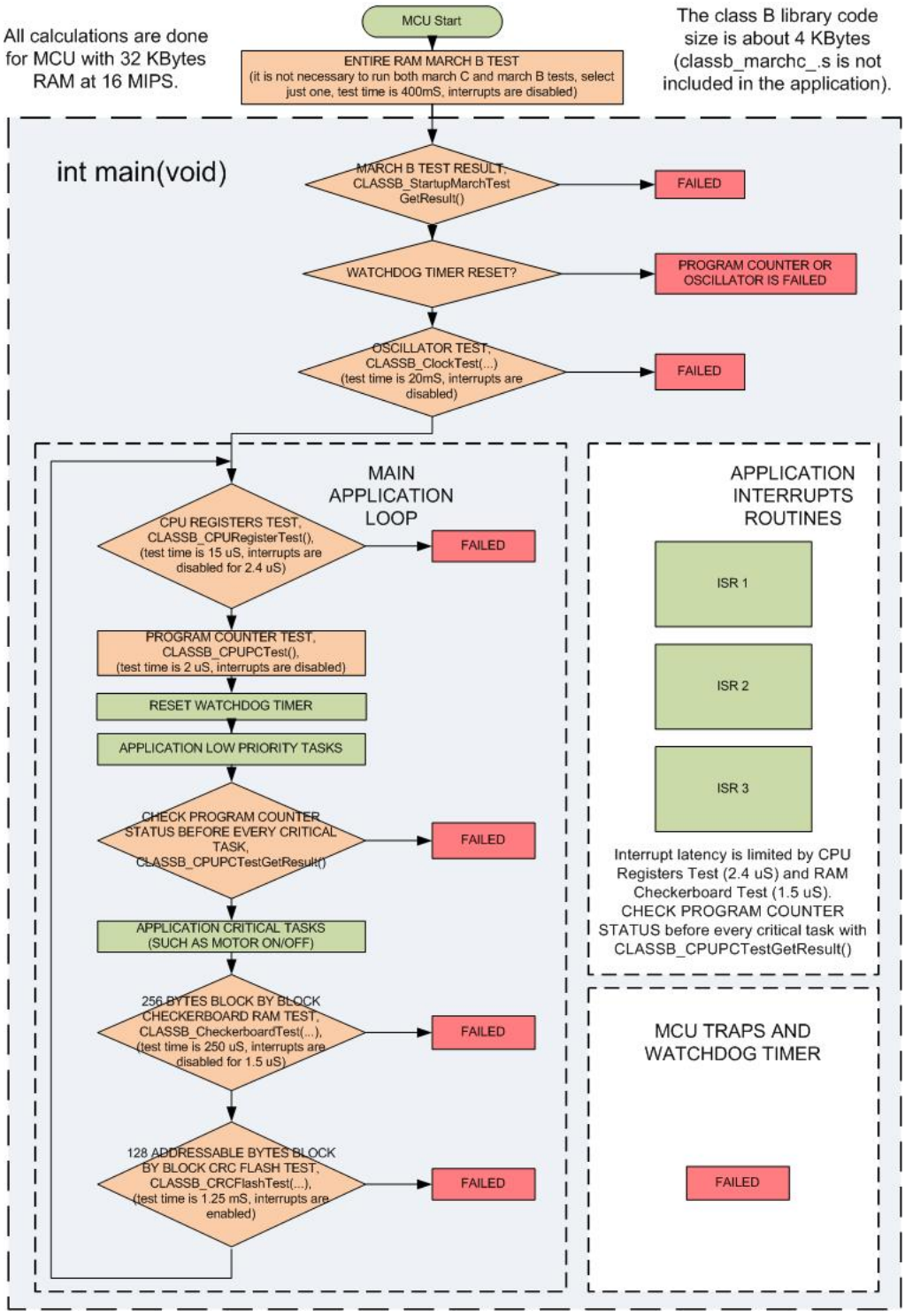
```
0x0fa4)) // length in bytes (must be aligned by 4 bytes)
{
// ERROR IS DETECTED
while (1);
}

-----------------------------------------------------------------------------------------------------------
// March C test.
// !!!The selected memory region must not cross the buffer!!!
// !!! Interrupts are disabled during this test.!!!
if (CLASSB_RAMMarchCTest(0x0850,  // start address (must be aligned by word (2 bytes))
sizeof(marchTestBuffer), // byte length (must be even number)
marchTestBuffer,  // buffer to store the tested memory content
false))
{
// ERROR IS DETECTED
while(1);
}

-----------------------------------------------------------------------------------------------------------
// March C Minus test.
// !!!The selected memory region must not cross the buffer!!!
// !!! Interrupts are disabled during this test.!!!
if (CLASSB_RAMMarchCTest(0x0850,  // start address (must be aligned by word (2 bytes))
sizeof(marchTestBuffer), // byte length (must be even number)
marchTestBuffer,  // buffer to store the tested memory content
true))
{
// ERROR IS DETECTED
while(1);
}

-----------------------------------------------------------------------------------------------------------
//  March B test.
// !!!The selected memory region must not cross the buffer!!!
// !!! Interrupts are disabled during this test.!!!
if(CLASSB_RAMMarchBTest((0x0850,  // start address (must be aligned by word (2 bytes))
sizeof(marchTestBuffer), // byte length (must be even number)
```

```
marchTestBuffer))  // buffer to store the tested memory content
{
// ERROR IS DETECTED
while(1);
}
-----------------------------------------------------------------------------------------------------------
//  Flash memory CRC.
// Calculate the check sum for interrupt vectors table and .text section.
// The linker has generated the following map information:
// Program Memory [Origin = 0x200, Length = 0x2a9fe]
//
// section address length (PC units) length (bytes) (dec)
// ------- ------- ----------------- --------------------
// .text 0x200 0xc0 0x120 (288)
// .text 0x2c0 0xa80 0xfc0 (4032)
// .dinit 0xd40 0x8 0xc (12)
// _03F32480_at_address_0000AAAA 0xaaaa 0x6 0x9 (9)
// _03F32400_at_address_00015554 0x15554 0x6 0x9 (9)
//
// Total program memory used (bytes): 0x10fe (4350) 1%

checkSum = CLASSB_CRCFlashTest(0x000000, // interrupt vectors table start address
0x000100, // 256 PC units (addressable bytes)
-1);  // initial seed is 0xffff

checkSum = CLASSB_CRCFlashTest(0x000200, // the first part of the .text
0x0000c0, // length in PC units (addressable bytes)
checkSum); // seed is the previous check sum

checkSum = CLASSB_CRCFlashTest(0x0002c0,  // the second part of the .text
0x000a80,  // length in PC units (addressable bytes)
checkSum); // seed is the previous check sum

-----------------------------------------------------------------------------------------------------------
// Clock test.
// !!! Interrupts are disabled during this test.!!!
// The reference clock should be connected to the timer.The timer must be initialized by the application code
// to count the reference clock pulses. The period register must be set to the maximum value. The address of
```

```
// the timer counter register must be specified in classb_config.h using the compile time option
// CLASSB_CLOCK_TEST_TIMER_ADDRESS. There's a second compile option CLASSB_CLOCK_TEST_TIME_MS
// in the classb_config.h. It defines the test time and can be about 20 mS if clock is between 1-32MHz and the
// reference clock is 50Hz-33kHz.
if(CLASSB_ClockTest(32000000, // clock is 32 MHz
32768, // secondary oscillator frequency as reference
5)) // 0.5% tolerance limit
{
// ERROR IS DETECTED
while(1);
```


3.3.1 Typical Application using Class B Library Flowchart

The following flowchart shows a possible application structure using the class B library:



3.4 Demo Projects

The demo projects are located in the "CLASS B Demos" folder. The demos can be run using the Explorer 16 Development Board with PIC24F or dsPIC33 plug-in modules (DM240001), 16-bit XLP Development Board (DM240311) and the Motor Control Starter Kit with mTouch Sensing Board (DM330015). The device specific code for the projects (board hardware initialization, MCU traps, configuration bits and so on) is placed in system.h and system.c files in "CLASS B

Demos\Configurations" folder. Each demo project includes classb_config.h file for the library configuration.

4 Tests Execution Time and Interrupts Timing

The time was measured for "S" code optimization option.

Test Name	Test Function Name	Time	Interrupts Timing
March C	CLASSB_RAMMarchCTest	115200 Instruction Cycles per KByte	Disabled during the test.
March C Minus	CLASSB_RAMMarchCTest	121600 Instruction Cycles per KByte	Disabled during the test.
March B	CLASSB_RAMMarchBTest	192000 Instruction Cycles per KByte	Disabled during the test.
Checker Board	CLASSB_RAMCheckerboardTest	8000 Instruction Cycles per KByte	Disabled for 24 Instruction Cycles.
Flash CRC	CLASSB_CRCFlashTest	160000 Instruction Cycles per KByte	Enabled.
EEPROM CRC	CLASSB_CRCEEPROMTest	128000 Instruction Cycles per KByte	Enabled.
MCU Registers	CLASSB_CPURegistersTest	240 Instruction Cycles	Disabled for 39 Instruction Cycles.
MCU Program Counter	CLASSB_CPUPCTest	32 Instruction Cycles	Disabled for 32 Instruction Cycles.
Clock Source	CLASSB_ClockTest	160000-640000 Instruction Cycles	Disabled during the test.

5 Memory Requirements

The library requires just a few bytes of RAM. The program memory size for each library module is listed in the following table:

Test Name	Files	Flash
March C, March C Minus	classb_marchc_.s	1364 Bytes
March B	classb_marchb_.s	2003 Bytes
Checker Board	classb_checkerboard_.s	190 Bytes
Flash CRC, EEPROM CRC	classb_crc.c	383 Bytes
MCU Registers	classb_registers_.s	1054 Bytes
MCU Program Counter	classb_pc_.s	41 Bytes
Clock Source	classb_clock.c, classb_clock_.s	633 Bytes
Total		5668 Bytes

6 API Reference

This section describes Class B library API.

6.1 Structures and Enumerations

In this section the library structures and enumerations are described.

6.1.1 CLASSBRESULT

Description: this enumeration is used by the class B test functions to return the result.

Values:

- **CLASSB_TEST_PASS** - the test finished successfully (**zero**).
- **CLASSB_TEST_FAIL** - the test is failed.
- **CLASSB_TEST_TIMEOUT** - the test is failed because a timeout was detected.

6.2 CLASSBRESULT CLASSB_RAMMarchBTest(startAddress, length, bufferAddress)

Description: This function tests the RAM region using march B algorithm. The interrupts are disabled during the test.

Parameters:

- **startAddress** - the first address of the tested memory (must be even number),
- **length** - the byte length of the tested memory (must be even number),
- **bufferAddress** - the first address of the buffer to store the tested memory content (must be even number), if this parameter is NULL then tested memory will be cleared.

Returns:

- The test result (zero if successful).

6.3 CLASSBRESULT CLASSB_RAMMarchCTest(startAddress, length, bufferAddress, minus)

Description: This macro tests the RAM region using march C or march C minus algorithms. The interrupts are disabled during the test.

Parameters:

- **startAddress** - the first address of the tested memory (must be even number),
- **length** - the byte length of the tested memory (must be even number),
- **bufferAddress** - the first address of the buffer to store the tested memory content (must be even number), if this parameter is NULL then tested memory will be cleared,
- **minus** - if the parameter is TRUE the "minus" algorithm is used.

Returns:

- The test result (zero if successful).

6.4 CLASSBRESULT CLASSB_StartupMarchTestGetResult()

Description: This macro returns the result of the startup march tests. These tests are run before main() function entry if CLASSB_MARCH_C_STARTUP, CLASSB_MARCH_C_MINUS_STARTUP or/and CLASSB_MARCH_B_STARTUP flags are set in classb_config.h file. The entire memory march tests can take a while. The watchdog timer should provide enough time to complete the tests. It is not possible to debug the startup march test code portion because these tests erase the debugger data in the memory.

Returns:

- The test result (zero if successful).

6.5 CLASSBRESULT CLASSB_RAMCheckerboardTest(startAddress, length)

Description: This function tests the RAM region using the checker board pattern.

Parameters:

- **startAddress** - the first address of the tested memory (must be even number),
- **length** - the byte length of the tested memory (must be aligned by 4 bytes).

Returns:

- Returns zero if successful. Non zero means - failed.

6.6 CLASSBRESULT CLASSB_CPURegistersTest()

Description: This function tests CPU registers.

Returns:

- The test result (zero if successful).

6.7 CLASSBRESULT CLASSB_CPUPCTest()

Description: This macro tests the CPU program counter. The device flash memory size must be set using parameter CLASSB_DEVICE_FLASH_SIZE_KB in classb_conig.h. CLASSB_CPUPCTestGetResult() should be used to get the result of the test. Check program counter status everywhere using CLASSB_CPUPCTestGetResult() especially before critical code execution (such as motor on/off, power enable and so on). The interrupts are disabled during the test.

Returns:

- None.

6.8 CLASSBRESULT CLASSB_CPUPCTestGetResult()

Description: This macro returns the CPU program counter test result done with CLASSB_CPUPCTest() macro.

Returns:

- The test result (zero if successful).

6.9 uint16_t CLASSB_CRCFlashTest(startAddress, length, crcSeed)

Description: The function calculates the CRC check sum for the flash memory region. The start address and length must be even numbers (aligned by 2). The "length" parameter must be specified in PC units (addressable bytes, the PC units are used in map files).

Parameters:

- **startAddress** - the first address of the tested flash memory in bytes (must be even number),
- **length** - the length of the tested flash memory in PC units (must be even number),
- **crcSeed** - initial value of the CRC check sum.

Returns:

- The function returns the standard 16-bit CRC.

6.10 uint16_t CLASSB_CRCEEPROMTest(startAddress, length, crcSeed)

Description: The function calculates the CRC check sum for the EEPROM memory region. The start address and length must be even numbers (aligned by 2).

Parameters:

- **startAddress** - the first address of the tested EEPROM memory in bytes (must be even number),
- **length** - the byte length of the tested EEPROM memory (must be even number),
- **crcSeed** - initial value of the CRC check sum.

Returns:

- The function returns the standard 16-bit CRC.

6.11 CLASSBRESULT CLASSB_ClockTest(uint32_t clockFrequency, uint32_t referenceFrequency, uint16_t tolerance)

Description: The function tests the CPU clock source. The interrupts are disabled during the test. The reference clock should be connected to the timer. The timer must be initialized by the application code to count the reference clock pulses. The period register must be set to the maximum value. The address of the timer counter register must be specified in classb_config.h using the compile time option CLASSB_CLOCK_TEST_TIMER_ADDRESS. There's another compile option CLASSB_CLOCK_TEST_TIME_MS in the classb_config.h. It defines the test time and can be about 20 mS if clock is between 1-32MHz and the reference clock is 50Hz-33kHz.

Parameters:

- **clockFrequency** - frequency of the clock source,
- **referenceFrequency** - frequency of the reference clock (such as power line or secondary oscillator),
- **tolerance** - maximum valid frequency tolerance, can be from 1(0.1%) to 100(10%).

Returns:

- Returns zero if successful. Non zero means - failed. See CLASSBRESULT enumeration for details.

7 Known Limitations

The limitations of Class B software library are listed below:

- The address and length of the tested memory are required by some test procedures. These parameters MUST be even numbers (aligned by 2). For the CLASSB_RAMCheckerboardTest(...) the length parameter must be aligned by 4 bytes.
- The interrupts are disabled during the march C, march B, program counter and clock tests.

8 Resources

To get more information about Class B Library visit http://www.microchip.com/classb and read the following articles:

- AN1229 - Class B Safety Software Library for PIC® MCUs and dsPIC® DSCs

Index