

**Core self-test library compliant with IEC 60730, IEC 60335 UL 60730,
UL 1998 documentation**

Variable Memory Test

Document revision history

Date	Author	Version	Notes
11/2015	Jozef Sedlak	0.1	Initial version
11/2015	Jozef Sedlak	1.0	Version for certification
11/2015	Jozef Sedlak	1.1	Correction of source code – pattern for March
10/2016	Jozef Sedlak	1.2	NXP
11/2018	Jozef Sedlak	3.0	Release with new supported compilers

1	VARIABLE MEMORY TEST ARCHITECTURE	4
2	VARIABLE MEMORY TEST IN COMPLIANCE WITH IEC/UL STANDARDS	7
3	VARIABLE MEMORY TEST IMPLEMENTATION.....	7
3.1	IEC60730B_CM4_CM7_RAM_AFTERRESETTEST().....	8
3.2	IEC60730B_CM4_CM7_RAM_RUNTIMETEST()	10
3.3	IEC60730B_CM4_CM7_RAM_SEGMENTMARCHC()	11
3.4	IEC60730B_CM4_CM7_RAM_SEGMENTMARCHX()	12
3.5	IEC60730B_CM4_CM7_RAM_COPYTOBACKUP()	12
3.6	IEC60730B_CM4_CM7_RAM_COPYFROMBACKUP().....	13
4	VARIABLE MEMORY TEST MODULE TEST CONCEPT	14
5	VARIABLE MEMORY TEST VALIDATION.....	14

1 Variable Memory Test Architecture

The variable memory test for CM4 and CM7 Kinetis devices checks the on chip RAM for DC faults. The application stack area can also be tested. The March C and March X schemes are used as control mechanisms. The User (application developer) must choose whether the March C or March X scheme will be used. The handling functions for an after-reset test and for a runtime test are different. Both functions need to have a backup area defined in the RAM and reserved by the application developer. The size of this area must be at least the same as the size of the tested block. A RAM test is considered destructive. This is because the data from the memory area with the variables, the stack area and the functions placed in the RAM is moved away, rewritten multiple times and then moved back to the original memory area. The test procedure is very sensitive and cannot be interrupted. The block diagrams for the RAM tests are shown in following figures:

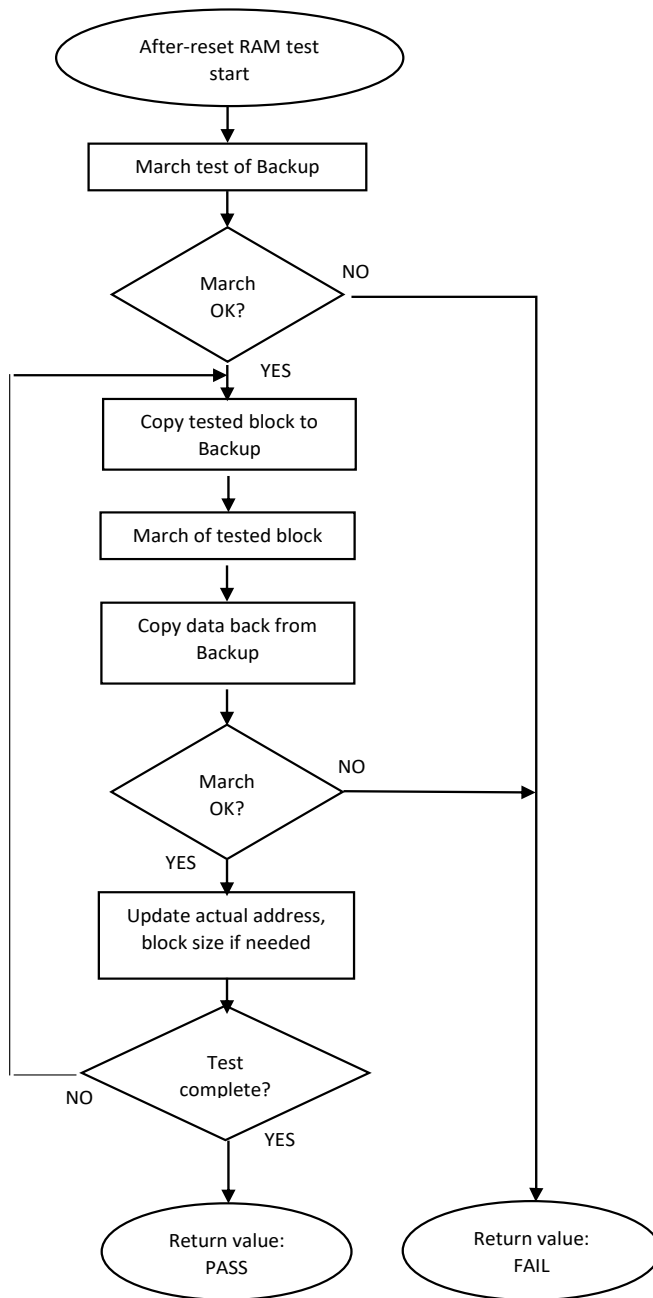
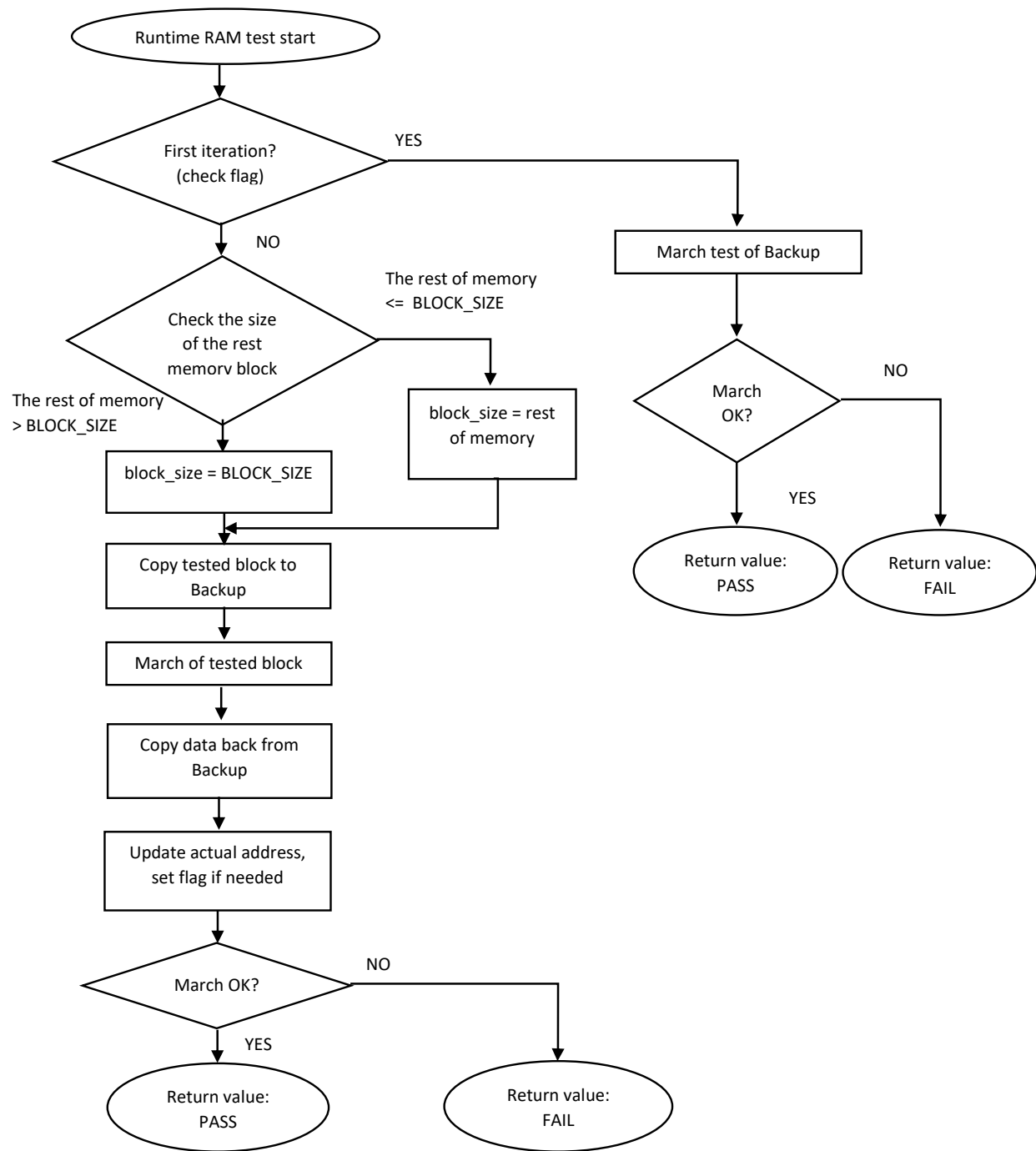


Figure 1. Block diagram for after-reset test of RAM

**Figure 2. Block diagram for runtime test of RAM**

2 Variable Memory Test in compliance with IEC/UL standards

The performed overload test fulfils safety requirements according to IEC 60730-1, IEC 60335, UL 60730 and UL 1998 standards as described in the following table:

Table 1. Variable Memory Test in Compliance with the Standards

Test	Component	Fault/Error	SW / HW Class	Acceptable Measures
Variable memory	4.2 – Variable memory	DC fault	B/R.1	Periodic self-test using March test

3 Variable Memory Test Implementation

The test functions for the variable memory (RAM) test are placed in *IEC60730_B_CM4_CM7_ram.S* and are written as assembler functions. The header file with return values and function prototypes is *IEC60730_B_CM4_CM7_ram.h*. *IEC60730_B_CM4_CM7.h* and *asm_mac_common.h* are files that are included in *IEC60730_B_ram.S* and therefore need to also be placed in the application.

The RAM test consists of the following public functions:

- *IEC60730B_CM4_CM7_RAM_RuntimeTest()*
- *IEC60730B_CM4_CM7_RAM_AfterResetTest()*
- *IEC60730B_CM4_CM7_RAM_SegmentMarchC()*
- *IEC60730B_CM4_CM7_RAM_SegmentMarchX()*
- *IEC60730B_CM4_CM7_RAM_CopyToBackup()*
- *IEC60730B_CM4_CM7_RAM_CopyFromBackup()*

The first two functions provide a complex RAM test. The application developer does not need to work directly with the remaining functions.

3.1 IEC60730B_CM4_CM7_RAM_AfterResetTest()

The after-reset test is done by the *IEC60730B_CM4_CM7_RAM_AfterResetTest* function. This function is called once after reset, when the execution time is not critical. The application developer must reserve free memory space for the backup area. The block size parameter cannot be larger than the size of the backup area. The function firstly checks the backup area. Then the loop begins. Blocks of memory are copied to the backup area and their locations are checked with the respective March test. The data is copied back to the original memory area and the actual address with the block size is updated. This is repeated until the last block of memory is tested. If a DC fault is detected, the function returns a fail pattern. The block diagram is shown in Figure 1.

Example of the function calling:

```
#include "IEC60730_B_CM4_CM7.h"
if(IEC60730B_ST_RAM_FAIL == IEC60730B_CM4_CM7_RAM_AfterResetTest(start_address, \
end_address, block_size, backup_address, IEC60730B_CM4_CM7_RAM_SegmentMarchC))
SafetyError();
```

Function prototype:

```
IEC60730B_RESULT IEC60730B_CM4_CM7_RAM_AfterResetTest ( unsigned long first_address,
unsigned long last_address, unsigned long block_size, unsigned long backup_address,
tFcn pMarch_type);
```

Function inputs:

start_address – Starting address for test.

end_address – Last address for test.

block_size – Size of block for test iterations. It is limited by the size of backup area. It should be a multiple of 0x4.

backup_address – Address of the backup area of RAM.

pMarch_type – pointer to the function to be used for the March test.
(IEC60730B_CM4_CM7_RAM_SegmentMarchC, or
IEC60730B_CM4_CM7_RAM_SegmentMarchX)

Function output:

typedef unsigned long IEC60730B_RESULT;

Can have two values:

IEC60730B_ST_RAM_FAIL (0x00000401)

IEC60730B_ST_RAM_PASS (0)

Function performance:

Function size is 98 bytes.²

The execution time depends on the memory size. It is also varies with different block sizes and which March method is used.¹

Table 2. IEC60730B_CM4_CM7_RAM_AfterResetTest duration

Memory size(Bytes)	Block size (Bytes)	Cycles – March X	Cycles – March C
0x100	0x20	3715	5085
0x100	0x40	3723	5205
0x100	0x80	4099	5854
0x200	0x20	7075	9661
0x200	0x40	6843	9509
0x200	0x80	7099	10021
0x400	0x20	13795	18813
0x400	0x40	13083	18117
0x400	0x80	13099	18357

Calling restrictions:

This function is used once after the microcontroller reset, when the execution time is not critical. It cannot be interrupted. The backup area must be at least the same size as the tested block size defined by the block_size parameter.

3.2 IEC60730B_CM4_CM7_RAM_RuntimeTest()

The runtime test is done by the *IEC60730B_CM4_CM7_RAM_RuntimeTest()* function. The application developer must reserve free memory space dedicated for the backup area. The block size parameter cannot be larger than the size of the backup area. During the first call the function checks the backup area. After the call the blocks of memory are processed in sequence. They are copied to the backup area and their locations are checked with the respective March test. Data is copied back to the original memory area and the actual address and the block size are updated. This is repeated until the last block of memory is tested. If a DC fault is detected the function returns a fail pattern. The block diagram is shown in Figure 2. The example of function calling is as follows:

```
#include "IEC60730_B_CM4_CM7.h"
if(IEC60730B_ST_RAM_FAIL == IEC60730B_CM4_CM7_RAM_RuntimeTest(start_address, end_address,
&actual_address, block_size, backup_address, IEC60730B_CM4_CM7_RAM_SegmentMarchX))
SafetyError();
```

Function prototype:

```
IEC60730B_RESULT IEC60730B_CM4_CM7_RAM_RuntimeTest ( unsigned long first_address,
unsigned long last_address, unsigned long *pActual_address, unsigned long block_size,
unsigned long backup_address, tFcn pMarch_type);
```

Function inputs:

first_address – Starting address for the test.

last_address – Last address for the test.

*pActual_address – Pointer to variable for the actual address. It is updated within the function.

block_size – Size of the block for test iterations. It is limited by the size of the backup area. It must be a multiple of 0x4.

backup_address – Address of the backup area of the RAM.

pMarch_type – pointer to the function to be used for the March test.
(IEC60730B_CM4_CM7_RAM_SegmentMarchC, or
IEC60730B_CM4_CM7_RAM_SegmentMarchX)

Function output:

```
typedef unsigned long IEC60730B_RESULT;
```

Can have two values:

```
IEC60730B_ST_RAM_FAIL (0x00000401)
IEC60730B_ST_RAM_PASS (0)
```

Function performance:

Function size is 118 bytes.²

The execution time depends on the block size and is different for March C and March X methods.¹

Table 3. IEC60730B_CM4_CM7_RAM_RuntimeTest duration

Block size(Bytes)	Cycles – March X	Cycles – March C
0x4	193	216
0x8	235	283
0x20	504	675
0x40	870	1179

Calling restrictions:

The function cannot be interrupted. The backup area must have at least the same size as the tested block size defined by the block_size parameter. The execution time depends on the block size.

3.3 IEC60730B_CM4_CM7_RAM_SegmentMarchC()

This function performs a March C test of the memory block that is given by the start address and the block size. The content of the tested memory remains changed after the execution of this function.

Function prototype:

IEC60730B_RESULT IEC60730B_CM4_CM7_RAM_SegmentMarchC(unsigned long start_address, unsigned long block_size);

Function inputs:

start_address – Starting address for the test.
 block_size – Size of the block to be tested.

Function output:

typedef unsigned long IEC60730B_RESULT;

This function can have two values:

IEC60730B_ST_RAM_FAIL (0x00000401)

IEC60730B_ST_RAM_PASS (0)

Function performance:

Function size is 118 bytes.²

3.4 IEC60730B_CM4_CM7_RAM_SegmentMarchX()

This function performs a March X test of the memory block that is given by the start address and the block size. The content of the tested memory remains changed after the execution of this function.

Function prototype:

IEC60730B_RESULT IEC60730B_CM4_CM7_RAM_SegmentMarchX(unsigned long start_address, unsigned long block_size);

Function inputs:

start_address – Starting address for the test.
block_size – Size of the block to be tested.

Function output:

typedef unsigned long IEC60730B_RESULT;

This can have two values:

IEC60730B_ST_RAM_FAIL (0x00000401)
IEC60730B_ST_RAM_PASS (0)

Function performance:

Function size is 86 bytes.²

3.5 IEC60730B_CM4_CM7_RAM_CopytoBackup()

This function performs a copy of data to the backup area.

Function prototype:

void IEC60730B_CM4_CM7_RAM_CopyToBackup(unsigned long start_address, unsigned long block_size, unsigned long backup_address);

Function inputs:

start_address - starting address of the memory with data to be copied.
block_size - the size of the block to be copied.
backup_address - starting address of the backup memory.

Function output:

Void. The function does not return any value.

Function performance:

The function size is 16 bytes.²

3.6 IEC60730B_CM4_CM7_RAM_CopyfromBackup()

This function performs a copy of the data from the backup area to the original area of memory defined by its starting address.

Function prototype:

```
void IEC60730B_CM4_CM7_RAM_CopyFromBackup(unsigned long start_address,  
unsigned long block_size, unsigned long backup_address);
```

Function inputs:

start_address - starting address of the memory
block_size - size of the block to be copied
backup_address - starting address of the backup memory.

Function output:

Void. The function does not return any value.

Function performance:

The function size is 16 bytes.²

-
- 1- The number of cycles and execution time were measured with the use of a MKV31 / 80 MHz CPU clock / 20 MHz flash clock.
 - 2- Function compiled by IAR v8.22.2

4 Variable Memory Test Module test concept

Content moved to TestReport_IEC60730B_RAM_rev3_0 document

5 Variable Memory Test Validation

Content moved to TestReport_IEC60730B_RAM_rev3_0 document

Table V. Validation

Date	Validated by	Validated Revision of Document	Validated Version of Source Code	Validation Result
11/2015	Jaroslav Lepka	1.0	1.0	P
11/2015	Jaroslav Lepka	1.1	1.1	P
11/2016	Pavel Sustek	1.2	1.1	P
11/2018	Jaroslav Lepka	3.0	3.0	P

Validation result options:

P – Passed

F – Failed

N/A – Not applicable

How to Reach Us:

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address:
nxp.com/SalesTermsandConditions.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, Freescale, and the Freescale logo are trademarks of NXP B.V. All other product or service names are the property of their respective owners.

ARM, the ARM logo, and Cortex are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

© 2016 NXP B.V.

