**RESTRICTED ACCESS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DISTRIBUTION :** | **Firm** | **To** | **Copies** | e-mail |
|  | AEE-C |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**SW MODULE/GLOBAL INTEGRATION TEST PLAN**

OBJECT: This document gathers the tests for the integration of BFS software module

SUMMARY: Those integration tests are done on BFS SW-C for PP Platform.

These tests have been designed for High Power platform project with Autosar environment.

CONCLUSION:

**EVOLUTION OF THE DOCUMENT**

|  |  |  |  |
| --- | --- | --- | --- |
| Issue | Date | Author | Motive and nature of the modifications |
| 001 | 02.10.12 | C.Sauvage | First release. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**CONTENTS**

1. scope of INTEGRation 3

2. List of applicable documents 3

3. list of reference documents 3

4. Terminology 3

5. Integration Test Strategy 4

6. LIST OF TESTS 5

6.1 Tests for BFS\_runBeltFunctionSelection 6

6.1.1 INT\_BFS\_01001: Period call and scheduling 6

6.1.2 INT\_BFS\_01002: Triggering flags management 7

6.1.3 INT\_BFS\_01003: Interrupt flags management 8

6.1.4 INT\_BFS\_01004: Belt function priorities 9

7. Annex 10

7.1 Belt functions priorities for integration tests 10

7.2 Cycles & steps parameters for integration tests 10

7.3 Adaptation of Belt Function decision algorithms 10

# scope of INTEGRation

The purpose of the tests is to check the correct integration of the BFS component (which is a generic component provided by the PP Platform) in the

# List of applicable documents

|  |  |  |  |
| --- | --- | --- | --- |
| **Nb** | **DOCUMENT** | **REFERENCE** | **Company** |
|  | SW work Product Follow-up (SPF) | Exxxxxx | **AEFC** |
|  | BFS – SW Design Document | Available on MKS | **AEFC** |
|  |  |  |  |

Note: The documents releases are specified in the SPF.

# list of reference documents

|  |  |  |  |
| --- | --- | --- | --- |
| **Nb** | **DOCUMENT** | **REFERENCE** | **Company** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Note: The documents releases are specified in the SPF.

# Terminology

BFD: Belt Function Decision

BFE: Belt Function Execution

BFS: Belt Function Selection

DTC: Diagnostic Trouble Code

NVM: Non Volatile Memory

RTE: Real Time Environment

TBC: To Be Confirmed

TBD: To Be Defined

# Integration Test Strategy

The Integration Test strategy adopted is detailed in the SW Global Integration Test Follow-up. It specifies the test environment (software components, hardware means, compilation options used, EEPROM file, performance and timing …)

* Tests definition

The integration tests are divided in 4 types: nominal, robustness, endurance and qualification tests.

Nominal Test: A nominal test consists to apply entry values sequences to the software that should cause all the behaviours expected in the design documents.

Robustness Test: A robustness test consists to inject faults and limits values to the software in order to check that the behaviour is coherent to the designs documents.

Endurance Test: An endurance test is done in order to check the repetitiveness and the repeatability of the results.

Qualification Test: A qualification test aims at precisely checking (and reporting) a behaviour or a metric (for instance to measure a timing or a memory consumption).

# LIST OF TESTS

The classification to use for the test type is:

- Nominal tests (N),

- Robustness tests (R),

- Endurance tests (E).

- Qualification tests (Q)

|  |  |  |
| --- | --- | --- |
| **Test**  **Id** | **Type**  **Classification** | **Purpose** |
| **BFS\_AC\_BeltFunctionSelection\_BFS\_runBeltFunctionSelection** | | |
| INT\_BFS\_01001 | N | Period call and scheduling |
| INT\_BFS\_01002 | N | Triggering flags management |
| INT\_BFS\_01003 | N | Interrupt flags management |
| INT\_BFS\_01004 | N | Belt function priorities |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |



## Tests for BFS\_runBeltFunctionSelection

### INT\_BFS\_01001: Period call and scheduling

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  | |  |  | |  |  |
|  | **Type of the test: N** | | | | | | | | | | |  |
|  |  |  | |  |  |  | |  |  | |  |  |
|  | **Purpose of the test:**  The goal of this test is to check the periodicity of the BFS main function call and to check that the first call is performed after the RTE initialization function | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **Environment:**  CAN environment to stay alive.  ECU or mock-up flashed with an instrumented code.  Debugger plugged with 2 breakpoints :   * At the closing brace of RTE\_Init function * At the beginning of BFS main function   A toggle pin at the entry of BFS main function shall be added and a scope probe shall be used at the debug pin to check the periodicity | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **INITIAL STATE:**  ECU flashed and not running | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **ACTION:**   1. Perform an In Target reset of the application. 2. Run the application and wait the stop at the first breakpoint point 3. Press ‘Go’ button to continue until the next breakpoint 4. Remove the 2nd breakpoint & press ‘Go’ button | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **EXPECTED RESULT:**   1. Nothing expected 2. Check that the program halts at the end of the RTE\_Init function at first 3. Check that the program halts at the entry of the BFS main function. 4. Check on the scope that the signal on the debug pin toggles every 10 ms   [COVERS: DSG\_BFS\_00004] | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  |  |  |  | |  | |  |  | |  |  |  |

### INT\_BFS\_01002: Triggering flags management

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  | |  |  | |  |  |
|  | **Type of the test: N** | | | | | | | | | | |  |
|  |  |  | |  |  |  | |  |  | |  |  |
|  | **Purpose of the test:**  The goal of this test s to check the nominal behavior of the BFS | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **Environment:**  CAN environment to stay alive.  ECU or mock-up flashed with an instrumented code.  On the Watch Window of the debugger : access to the 6 data defined in the ‘Annex’ chapters [7]  On the Watch Window : add the executed cycle data (provided by BFE component) & the Selected cycle (output of the BFS) | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **INITIAL STATE:**  ECU flashed and not running  Software instrumented like described on the Annex chapter | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **ACTION:**   1. Perform an In Target reset of the application. 2. Run the application. 3. Request the triggering of BFD 1 (u8TriggerFlagCycle0\_test = 0xAA) 4. Remove the triggering request (u8TriggerFlagCycle0\_test = 0x55) for the BFD 1 before 10 sec 5. Request the triggering of BFD 2 (u8TriggerFlagCycle1\_test = 0xAA) 6. Remove the triggering request (u8TriggerFlagCycle1\_test = 0x55) for the BFD 2 after 14 sec | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **EXPECTED RESULT:**   1. Nothing expected 2. Check that no cycle is executed (excepted the anti-patina belt function if the anti-patina is not managed by modified BFD 1/2/3) 3. Check that the First cycle starts and that au8TrigFlags[0] = 0xAA 4. Check that the cycle stops after 10 seconds, and that the execution counter for the Belt function 01 has been incremented by one. And au8TrigFlags[0] = 0x55 5. Checks that the 2nd cycle starts and au8TrigFlags[1] = 0xAA 6. Check that the cycle ends & restarts after 10 seconds then stops after 20 seconds. And that the execution counter for the belt function 02 has been incremented by 2. And au8TrigFlags[1] = 0x55   [COVERS: DSG\_BFS\_00001] | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  |  |  |  | |  | |  |  | |  |  |  |

### INT\_BFS\_01003: Interrupt flags management

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  | |  |  | |  |  |
|  | **Type of the test: N** | | | | | | | | | | |  |
|  |  |  | |  |  |  | |  |  | |  |  |
|  | **Purpose of the test:**  The goal of this test is to check that an executed cycle is well interrupted in case of interruption request. | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **Environment:**  CAN environment to stay alive.  ECU or mock-up flashed with an instrumented code.  On the Watch Window of the debugger : access to the 6 data defined in the ‘Annex’ chapters [7]  On the Watch Window : add the executed cycle data (provided by BFE component) & the Selected cycle (output of the BFS) | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **INITIAL STATE:**  ECU flashed and not running  Software instrumented like described on the Annex chapter | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **ACTION:**   1. Perform an In Target reset of the application. 2. Run the application. 3. Request the triggering of BFD 1 (u8TriggerFlagCycle0\_test = 0xAA) 4. Remove the triggering request (u8TriggerFlagCycle0\_test = 0x55) for the BFD 1 before 10 sec and set the interrupt flag (u8AbortFlagCycle0\_test = 0xAA) for the BFD 1 in the same time 5. Request the triggering of BFD 1 (u8TriggerFlagCycle0\_test = 0xAA) 6. Remove the interrupt flag (u8AbortFlagCycle0\_test = 0x55) for the BFD 1 7. Remove the triggering request (u8TriggerFlagCycle0\_test = 0x55) for the BFD 1 after few seconds (<10) | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **EXPECTED RESULT:**   1. Nothing expected 2. Check that no cycle is executed (excepted the anti-patina belt function if the anti-patina is not managed by modified BFD 1/2/3) 3. Check that the First cycle starts and that au8TrigFlags[0] = 0xAA 4. Check that the cycle stops immediately, and that the execution counter for the Belt function 01 has been incremented by one. And au8TrigFlags[0] = 0x55 & au8IntFlags[0] = 0xAA 5. Checks that the first cycle does not restart 6. Checks that the first cycle starts immediately when removing the interrupt flag 7. Checks that the 1st cycle stops after 10 seconds and au8TrigFlags[0] = 0x55 And that the execution counter for belt function 1 has been increased by one   [COVERS: DSG\_BFS\_00001, DSG\_BFS\_00002] | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  |  |  |  | |  | |  |  | |  |  |  |

### INT\_BFS\_01004: Belt function priorities

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  | |  |  | |  |  |
|  | **Type of the test: N** | | | | | | | | | | |  |
|  |  |  | |  |  |  | |  |  | |  |  |
|  | **Purpose of the test:**  The goal of this test is to check that an executed cycle is well interrupted in case of interruption request. | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **Environment:**  CAN environment to stay alive.  ECU or mock-up flashed with an instrumented code.  On the Watch Window of the debugger : access to the 6 data defined in the ‘Annex’ chapters [7]  On the Watch Window : add the executed cycle data (provided by BFE component) & the Selected cycle (output of the BFS) | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **INITIAL STATE:**  ECU flashed and not running  Software instrumented like described on the Annex chapter | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **ACTION:**   1. Perform an In Target reset of the application. 2. Run the application. 3. Request the triggering of BFD 1 (u8TriggerFlagCycle0\_test = 0xAA) 4. Request the triggering of BFD 2 (u8TriggerFlagCycle1\_test = 0xAA) 5. Remove the triggering of BFD 2 (u8TriggerFlagCycle1\_test = 0x55) | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  | **EXPECTED RESULT:**   1. Nothing expected 2. Check that no cycle is executed (excepted the anti-patina belt function if the anti-patina is not managed by modified BFD 1/2/3) 3. Check that the First cycle starts and that au8TrigFlags[0] = 0xAA 4. Check that the 2nd cycle is started. And au8TrigFlags[0]= au8TrigFlags[1]= 0xAA. the execution counter for belt function 1 & 2 shall have been increased by 1 5. Checks that the 2nd cycle stops after 10 seconds, then the 1st cycle starts after 10 ms. The execution counter for first cycle shall be increased by 1 again.   [COVERS: DSG\_BFS\_00001, DSG\_BFS\_00003] | | | | | | | | | | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |  |  | |  | |  |  | |  |  |  |
|  |  |  |  | |  | |  |  | |  |  |  |

# Annex

## Belt functions priorities for integration tests

For BFS integration tests, only the belt functions [1;3] will be used. The related priorities shall be follow the following rule :

Priority( Cycle 1 ) < Priority( Cycle 2 ) < Priority( Cycle 3 )

## Cycles & steps parameters for integration tests

For integration tests, it would be better to change the belt function parameters to avoid DTC qualification, or HW/Mechanical issues. It’s better to put the 3 first cycles in releasing direction.

|  |  |  |
| --- | --- | --- |
| **Byte :** | **Raw** | **Unit/Comment** |
| 1 | 0x04 | Calendar Week |
| 2 | 0x11 | Year |
| 3 | 0x07 |  |
| 4 | 0xFF | No following cycle |
| 5 | 0x00 | Step Id #0 |
| 6 | 0x01 | Step Id #1 |
| 7 | 0xFF | No Step |
| 8 | 0xFF | No Step |
| 9 | 0xFF | No Step |
| 10 | 0xFF | No Step |
| 11 | 0xFF | No Step |
| 12 | 0xFF | No Step |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Steps parameters : | | | | | | |
|  | **Byte :** | **Raw** | **Value** | **Unit/Comment** | | |
| **Step 1** | 1 | 03 |  |  |  |  |
| 2 | E8 | 10000 | ms |  |  |
| 3 | F6 | -10 | % PWM |  |  |
| 4 | 00 |  | S - Straight - Motor Power in % PWM | | |
| 5 | 00 | 0 |  |  |  |
| 6 | 00 |  |  |  |  |
| **Step 2** | 1 | 00 |  |  |  |  |
| 2 | 0A | 100 | ms |  |  |
| 3 | 00 | 0 | % PWM |  |  |
| 4 | 80 |  | A - Ramp-step - Motor Power in % PWM | | |
| 5 | 00 | 0 |  |  |  |
| 6 | 00 |  |  |  |  |

With this configuration the tests can be done with a complete retractor or a ‘free’ motor without DTCs.

This cycle is like a ‘long’ anti patina cycle.

## Adaptation of Belt Function decision algorithms

The belt functions decisions algorithms in charge of the update of the triggering/abortions flags for belt functions 1/2/3 shall be modified. 6 global data shall be defined and will be used (in association with RTE macros) during the tests to request the triggering and/or the abortion of these cycles.

Snippet :

|  |
| --- |
| /\* Declaration of local data for BFS tests execution : \*/  uint8 u8TriggerFlagCycle0\_test = 0x55;  uint8 u8TriggerFlagCycle1\_test = 0x55;  uint8 u8TriggerFlagCycle2\_test = 0x55;  uint8 u8AbortFlagCycle0\_test = 0x55;  uint8 u8AbortFlagCycle1\_test = 0x55;  uint8 u8AbortFlagCycle2\_test = 0x55;  /\* Modification of RTE services call in BFD algo \*/  Rte\_Write\_psrAlgo01\_Flags\_b8InterruptFlag(u8AbortFlagCycle0\_test);  Rte\_Write\_psrAlgo02\_Flags\_b8InterruptFlag(u8AbortFlagCycle1\_test);  Rte\_Write\_psrAlgo03\_Flags\_b8InterruptFlag(u8AbortFlagCycle2\_test);  Rte\_Write\_psrAlgo01\_Flags\_b8TriggerFlag(u8TriggerFlagCycle0\_test);  Rte\_Write\_psrAlgo02\_Flags\_b8TriggerFlag(u8TriggerFlagCycle1\_test);  Rte\_Write\_psrAlgo03\_Flags\_b8TriggerFlag(u8TriggerFlagCycle2\_test); |