EPC Gen2V2 Tag Test Report

Report No.: EMC94348B Rev. 2

TEST NAME: EPC Gen2 Version2 Baseline Interoperability (Tag)

Product Name: EM | echo
Trade Mark: EM4423V221
Product ID: EM4423V221
Manufacturer: EM Microelectronic Marin SA
Client: EM Microelectronic Marin SA
Standard(s): EPCglobal Class-1 Generation-2 UHF RFID Protocol, Release 2.0.1

This report shall not be reproduced except in full without the written permission of the Test Laboratory and shall not be quoted out of context.
TABLE OF CONTENTS

EPC GEN2V2 TAG TEST REPORT ................................................................. 1

TABLE OF CONTENTS ........................................................................... 2

1. IDENTIFICATION SUMMARY ................................................................. 3
   1.1. Test Laboratory ............................................................................... 3
   1.2. Client ......................................................................................... 4
   1.3. Manufacturer ............................................................................. 4
   1.4. Implementation Under Test (IUT) ................................................... 5
   1.5. Testing Environment ................................................................... 5
   1.6. Limits and reservations ................................................................. 6
   1.7. Record of agreement .................................................................... 6

2. IUT Baseline Interoperability Status ....................................................... 7

3. TEST CAMPAIGN REPORT (Test Logs) ................................................ 22

4. MEASUREMENT UNCERTAINTY ............................................................. 113

ANNEXES
   A. PICS
   B. PIXIT
   C. TEST RESULTS
   D. PHOTOGRAPHS
1. IDENTIFICATION SUMMARY

1.1. Test Laboratory

<table>
<thead>
<tr>
<th>NAME:</th>
<th>MET LABORATORIES, INC.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>914 W. PATAPSCO AVE.</td>
</tr>
<tr>
<td>City:</td>
<td>BALTIMORE</td>
</tr>
<tr>
<td>Postal code:</td>
<td>MD</td>
</tr>
<tr>
<td>Country:</td>
<td>21230</td>
</tr>
<tr>
<td>Telephone:</td>
<td>410-354-3300</td>
</tr>
<tr>
<td>Fax:</td>
<td>410-354-3313</td>
</tr>
<tr>
<td>URL:</td>
<td><a href="http://www.metlabs.com">www.metlabs.com</a></td>
</tr>
<tr>
<td>Contact person:</td>
<td></td>
</tr>
<tr>
<td>Name:</td>
<td>Benjamin Taylor</td>
</tr>
<tr>
<td>e-mail:</td>
<td><a href="mailto:Benjamin.taylor@metlabs.com">Benjamin.taylor@metlabs.com</a></td>
</tr>
</tbody>
</table>

**Competences and guarantees:**

MET Laboratories Inc. is a testing laboratory competent to carry out the tests described in this report.

In order to assure the traceability to other national and international laboratories, MET Laboratories has a calibration and maintenance programme for its measuring equipment.

MET Laboratories guarantees the reliability of the data presented in this report, which is the result of measurements and tests performed to the item under test on the date and under the conditions stated on the report and is based on the knowledge and technical facilities available at MET Laboratories at the time of execution of the test.

MET Laboratories is liable to the client for the maintenance by its personnel of the confidentiality of all information related to the item under test and the results of the test.
1.2. Client

<table>
<thead>
<tr>
<th>NAME:</th>
<th>EM Microelectronic Marin SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address:</td>
<td>Sors 3</td>
</tr>
<tr>
<td>City:</td>
<td>Marin-Epagnier</td>
</tr>
<tr>
<td>Postal code:</td>
<td>2074</td>
</tr>
<tr>
<td>Country:</td>
<td>Switzerland</td>
</tr>
<tr>
<td>Telephone:</td>
<td>+41327555195</td>
</tr>
<tr>
<td>Contact person:</td>
<td></td>
</tr>
<tr>
<td>Name:</td>
<td>Jim Springer</td>
</tr>
<tr>
<td>e-mail:</td>
<td><a href="mailto:Jim.Springer@emmicro-us.com">Jim.Springer@emmicro-us.com</a></td>
</tr>
</tbody>
</table>

1.3. Manufacturer

<table>
<thead>
<tr>
<th>NAME:</th>
<th>EM Micro</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address:</td>
<td>EM Microelectronic Marin SA</td>
</tr>
<tr>
<td>City:</td>
<td>Sors 3</td>
</tr>
<tr>
<td>Postal code:</td>
<td>Marin-Epagnier</td>
</tr>
<tr>
<td>Country:</td>
<td>2074</td>
</tr>
<tr>
<td>Telephone:</td>
<td>Switzerland</td>
</tr>
<tr>
<td>Contact person:</td>
<td></td>
</tr>
<tr>
<td>Name:</td>
<td>Contact person:</td>
</tr>
<tr>
<td>e-mail:</td>
<td>Paul Miller</td>
</tr>
</tbody>
</table>
1.4. Implementation Under Test (IUT)

<table>
<thead>
<tr>
<th>PRODUCT NAME:</th>
<th>EM</th>
<th>echo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trademark:</td>
<td>EM4423V221</td>
<td></td>
</tr>
<tr>
<td>Product ID:</td>
<td>EM4423V221</td>
<td></td>
</tr>
<tr>
<td>Hw version:</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Sw version:</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Profiles supported:</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Protocol Specification(s):</td>
<td>EPCglobal Generation 2 Version 2</td>
<td></td>
</tr>
<tr>
<td>ICS:</td>
<td>See annex A</td>
<td></td>
</tr>
<tr>
<td>Description of IUT:</td>
<td>UHF RFID Tag</td>
<td></td>
</tr>
<tr>
<td>Sample method:</td>
<td>Samples undergoing test have been selected by: The Client</td>
<td></td>
</tr>
</tbody>
</table>

1.5. Testing Environment

<table>
<thead>
<tr>
<th>IXIT:</th>
<th>SEE ANNEX B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Period of testing:</td>
<td>April, 2017</td>
</tr>
<tr>
<td>Conformance log reference:</td>
<td>See annex C</td>
</tr>
<tr>
<td>Retention date for log reference:</td>
<td>5 years</td>
</tr>
<tr>
<td>Test Requested</td>
<td>RF &amp; EPC Gen 2 v2 conformance</td>
</tr>
</tbody>
</table>

Means of testing identification:

<table>
<thead>
<tr>
<th>Hardware:</th>
<th>CISC RFID Xplorer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software:</td>
<td>CISC RFID Xplorer Tag Conformance Tester, v. 2-2722667_10</td>
</tr>
<tr>
<td>Test Setup:</td>
<td>See section 8</td>
</tr>
</tbody>
</table>

Test conditions:

NOMINAL

TEMPERATURE IN THE RANGE 18°C TO 27 °C
1.6. Limits and reservations

The test results presented in this test report apply only to the particular implementation under test (IUT) declared in section 1.4 of this report, for the functionality described in the relevant Protocol Implementation Conformance Statement (PICS), as presented for test on the date(s) declared in section 1.5 and configured as declared in the relevant Protocol Implementation Extra Information for Testing (PIXIT).

This test report does not constitute or imply, by its own, to be an approval of the product by Qualification Bodies, Certification Bodies or competent Authorities.

This document is only valid if complete; no partial reproduction can be made without written approval of the Test Laboratory.

This test report cannot be used partially or in full for publicity and/or promotional purposes without previous written approval of the Test Laboratory.

1.7. Record of agreement

The following samples were used for testing.

<table>
<thead>
<tr>
<th>METTrak No.:</th>
<th>SERIAL NO.:</th>
<th>DATE OF RECEPTION:</th>
</tr>
</thead>
<tbody>
<tr>
<td>94348</td>
<td>NA</td>
<td>02/11/2016</td>
</tr>
</tbody>
</table>
2. IUT Baseline Interoperability Status

This IUT has been shown by conformance assessment to be conforming to the referenced specification(s).

| Baseline Interoperability | None |

Number of test cases run

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PASSED</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failed</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inconclusive</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>12</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2.2. Interoperability Test Case Checklists

2.2.1. Access Memory

<table>
<thead>
<tr>
<th>Access_memory.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Lock</th>
<th>Action</th>
<th>AP State</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_1</td>
<td>-</td>
<td>zero</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_2</td>
<td>-</td>
<td>zero</td>
<td>Incorrect</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_3</td>
<td>-</td>
<td>non-zero</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_4</td>
<td>-</td>
<td>non-zero</td>
<td>Incorrect</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_5</td>
<td>-</td>
<td>non-zero</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_L_1</td>
<td>L</td>
<td>zero</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_L_2</td>
<td>L</td>
<td>zero</td>
<td>Incorrect</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_L_3</td>
<td>L</td>
<td>non-zero</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_L_4</td>
<td>L</td>
<td>non-zero</td>
<td>Incorrect</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>R_AP_L_5</td>
<td>L</td>
<td>non-zero</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_Z</td>
<td>zero</td>
<td>non-zero</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_NZ_1</td>
<td>non-zero</td>
<td>zero</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_NZ_2</td>
<td>non-zero</td>
<td>non-zero</td>
<td>Correct</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_NZ_3</td>
<td>non-zero</td>
<td>non-zero</td>
<td>Incorrect</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_NZ_4</td>
<td>non-zero</td>
<td>non-zero</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_L_NZ_1</td>
<td>L</td>
<td>non-zero</td>
<td>zero</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_L_NZ_2</td>
<td>L</td>
<td>non-zero</td>
<td>zero</td>
<td>Incorrect</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_L_NZ_3</td>
<td>L</td>
<td>non-zero</td>
<td>non-zero</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>W_AP_L_NZ_4</td>
<td>L</td>
<td>non-zero</td>
<td>non-zero</td>
<td>Incorrect</td>
</tr>
</tbody>
</table>
Table 1. Access Memory Test Cases

<table>
<thead>
<tr>
<th></th>
<th>W_AP_L_NZ_5</th>
<th>L</th>
<th>non-zero</th>
<th>non-zero</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Y</td>
<td></td>
<td>zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_2</td>
<td></td>
<td>zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_3</td>
<td></td>
<td></td>
<td>non-zero</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_4</td>
<td></td>
<td></td>
<td>non-zero</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>U_AP_L_1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>U_AP_L_2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>U_AP_L_3</td>
<td></td>
<td></td>
<td>non-zero</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>U_AP_L_4</td>
<td></td>
<td></td>
<td>non-zero</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Inventory Single Test Cases

<table>
<thead>
<tr>
<th>Inventory_single.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Lock</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>I</td>
<td></td>
<td>I</td>
<td>L</td>
<td>Non-select inventory</td>
</tr>
<tr>
<td>Yes</td>
<td>I_L</td>
<td></td>
<td>SI_E</td>
<td>L</td>
<td>Select EPC complete</td>
</tr>
<tr>
<td>Yes</td>
<td>SI_E_L</td>
<td></td>
<td>SI_T</td>
<td>L</td>
<td>Select TID complete</td>
</tr>
<tr>
<td>Yes</td>
<td>SI_T_L</td>
<td></td>
<td>SI_U</td>
<td>L</td>
<td>Select User complete</td>
</tr>
<tr>
<td>Yes</td>
<td>SI_U_L</td>
<td></td>
<td></td>
<td></td>
<td>Select User partial</td>
</tr>
</tbody>
</table>
### 2.2.3. Write Read

<table>
<thead>
<tr>
<th>Write_read.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Lock</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_KP</td>
<td></td>
<td>Kill password</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_KP_L</td>
<td>L</td>
<td>Kill password</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_KP_Z</td>
<td></td>
<td>Kill password zero</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_KP_NZ</td>
<td></td>
<td>Kill password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_KP_L_NZ</td>
<td>L</td>
<td>Kill password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>L_KP</td>
<td></td>
<td>Kill password</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>U_KP_L</td>
<td>L</td>
<td>Kill password</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>K_Z</td>
<td></td>
<td>Kill zero password</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>K_INZ</td>
<td></td>
<td>Kill incorrect non-zero password</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>K_INZ_L</td>
<td>L</td>
<td>Kill incorrect non-zero password</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_E_C</td>
<td></td>
<td>EPC complete</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_E_P</td>
<td></td>
<td>EPC partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_E_L_P</td>
<td>L</td>
<td>EPC partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_E_C</td>
<td></td>
<td>EPC complete</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_E_P</td>
<td></td>
<td>EPC partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_E_L_P</td>
<td>L</td>
<td>EPC partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>L_E</td>
<td></td>
<td>EPC memory</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>U_E_L</td>
<td>L</td>
<td>EPC memory</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_T_C</td>
<td></td>
<td>TID complete</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_T_P</td>
<td></td>
<td>TID partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_T_L_P</td>
<td>L</td>
<td>TID partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_T_C</td>
<td></td>
<td>TID complete</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_T_P</td>
<td></td>
<td>TID partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_T_L_P</td>
<td>L</td>
<td>TID partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>L_T</td>
<td></td>
<td>TID memory</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>U_T_L</td>
<td>L</td>
<td>TID memory</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_U_C</td>
<td></td>
<td>User complete</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_U_P</td>
<td></td>
<td>User partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_U_P_L</td>
<td>L</td>
<td>User partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_U_C</td>
<td></td>
<td>User complete</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_U_P</td>
<td></td>
<td>User partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_U_L_P</td>
<td>L</td>
<td>User partial</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>L_U</td>
<td></td>
<td>User memory</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>U_U_L</td>
<td>L</td>
<td>User memory</td>
</tr>
</tbody>
</table>

Table 3. Write Read Test Cases
### 2.2.4. PermaUnlocked

<table>
<thead>
<tr>
<th>PermaUnlocked.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Lock</th>
<th>Action</th>
<th>AP State</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>R_KP_PU</td>
<td>PU</td>
<td>Kill password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>W_KP_PU_NZ</td>
<td>PU</td>
<td>Kill password</td>
<td>non-zero</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>PU_KP</td>
<td>PU</td>
<td>Kill password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>U_KP_PU</td>
<td>PU</td>
<td>Kill password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>L_KP_PU</td>
<td>PU</td>
<td>Kill password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>PU_KP_PU</td>
<td>PU</td>
<td>Kill password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>K_INZ_PU</td>
<td>PU</td>
<td>Kill incorrect</td>
<td>non-zero</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>R_E_PU_P</td>
<td>PU</td>
<td>EPC partial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>W_E_PU_P</td>
<td>PU</td>
<td>EPC partial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>PU_E</td>
<td>EPC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>U_E_PU</td>
<td>EPC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>L_E_PU</td>
<td>EPC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>PL_E_PU</td>
<td>EPC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>PU_E_PU</td>
<td>EPC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>K_INZ_PU</td>
<td>PU</td>
<td>EPC memory</td>
<td>should fail</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>R_U_P_P</td>
<td>PU</td>
<td>User partial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>W_U_PU_P</td>
<td>PU</td>
<td>User partial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>PU_U</td>
<td>User</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>U_U_PU</td>
<td>User</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>L_U_PU</td>
<td>User</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>PL_U_PU</td>
<td>User</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>R_AP_PU_1</td>
<td>PU</td>
<td>-</td>
<td>zero</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>R_AP_PU_2</td>
<td>PU</td>
<td>-</td>
<td>zero</td>
<td>Incorrect</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>R_AP_PU_3</td>
<td>PU</td>
<td>-</td>
<td>non-zero</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>R_AP_PU_4</td>
<td>PU</td>
<td>-</td>
<td>non-zero</td>
<td>Incorrect</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>R_AP_PU_5</td>
<td>PU</td>
<td>-</td>
<td>non-zero</td>
<td>None</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>W_AP_PU_Z</td>
<td>PU</td>
<td>zero</td>
<td>non-zero</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>N/A</td>
<td>W_AP_PU_NZ_1</td>
<td>PU</td>
<td>non-zero</td>
<td>zero</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>W_AP_PU_NZ_2</td>
<td>PU</td>
<td>non-zero</td>
<td>non-zero</td>
<td>Correct</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>W_AP_PU_NZ_3</td>
<td>PU</td>
<td>non-zero</td>
<td>non-zero</td>
<td>Incorrect</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>W_AP_PU_NZ_4</td>
<td>PU</td>
<td>non-zero</td>
<td>non-zero</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>PU_AP</td>
<td>-</td>
<td>zero</td>
<td>Correct</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_PU_1</td>
<td>PU</td>
<td>-</td>
<td>zero</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_PU_3</td>
<td>PU</td>
<td>-</td>
<td>non-zero</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>PU_AP_PU_1</td>
<td>PU</td>
<td>-</td>
<td>zero</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>PU_AP_PU_2</td>
<td>PU</td>
<td>-</td>
<td>non-zero</td>
<td>Correct</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>PU_AP_PU_3</td>
<td>PU</td>
<td>-</td>
<td>non-zero</td>
<td>None</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SI_E_PU</td>
<td>PU</td>
<td>Select EPC</td>
<td>partial</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SI_T_PU</td>
<td>PU</td>
<td>Select TID</td>
<td>partial</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SI_U_PU</td>
<td>PU</td>
<td>Select User</td>
<td>partial</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>PU_U_PU</td>
<td>PU</td>
<td>User memory</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>PU_T</td>
<td>TID memory</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4. PermaUnlocked Test Cases

2.2.5. PermaLocked_L

<table>
<thead>
<tr>
<th>PermaLocked_L.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Lock</th>
<th>Action</th>
<th>AP State</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>PL_AP_L_1</td>
<td>L</td>
<td>-</td>
<td>non-zero</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>PL_AP_L_2</td>
<td>L</td>
<td>-</td>
<td>non-zero</td>
<td>None</td>
</tr>
</tbody>
</table>

Table 5. PermaLocked_L Test Cases
### 2.2.6. PermaLocked_APZ

<table>
<thead>
<tr>
<th>PermaLocked_APZ.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Lock</th>
<th>Action</th>
<th>AP State</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_KP_PL_1</td>
<td>PL</td>
<td>Kill password; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_KP_PL_NZ_1</td>
<td>PL</td>
<td>Kill password non-zero; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>PL_KP</td>
<td></td>
<td>Kill password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>PL_KP_L_1</td>
<td>L</td>
<td>Kill password; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>U_KP_PL_1</td>
<td>PL</td>
<td>Kill password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>L_KP_PL_1</td>
<td>PL</td>
<td>Kill password; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>PU_KP_PL_1</td>
<td>PL</td>
<td>Kill password; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>K_INZ_PL</td>
<td>PL</td>
<td>Kill incorrect non-zero password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>K_NZ</td>
<td></td>
<td>Kill non-zero password</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>R_E_PL_P_1</td>
<td>PL</td>
<td>EPC partial; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>W_E_PL_P_1</td>
<td>PL</td>
<td>EPC partial; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>PL_E_L_1</td>
<td>L</td>
<td>EPC memory; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>U_E_PL_1</td>
<td>PL</td>
<td>EPC memory; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td></td>
<td>L_E_PL_1</td>
<td>PL</td>
<td>EPC memory; Access password zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PU_E_PL_1</td>
<td>PL</td>
<td>EPC memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>----------------</td>
<td>------</td>
<td>----------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R_T_PL_P_1</td>
<td>PL</td>
<td>TID partial; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W_T_PL_P_1</td>
<td>PL</td>
<td>TID partial; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PL_T_L_1</td>
<td>L</td>
<td>TID memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U_T_PL_1</td>
<td>PL</td>
<td>TID memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L_T_PL_1</td>
<td>PL</td>
<td>TID memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PU_T_PL_1</td>
<td>PL</td>
<td>TID memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R_U_P_PL_1</td>
<td>PL</td>
<td>User partial; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W_U_PL_P_1</td>
<td>PL</td>
<td>User partial; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PL_U_L_1</td>
<td>L</td>
<td>User memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U_U_PL_1</td>
<td>PL</td>
<td>User memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L_U_PL_1</td>
<td>PL</td>
<td>User memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PU_U_PL_1</td>
<td>PL</td>
<td>User memory; Access password zero</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PL_AP_L</td>
<td>L</td>
<td>-</td>
<td>zero</td>
<td>Correct</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U_AP_PL_1</td>
<td>PL</td>
<td>-</td>
<td>zero</td>
<td>Correct</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L_AP_PL_1</td>
<td>PL</td>
<td>-</td>
<td>zero</td>
<td>Correct</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PU_AP_PL_1</td>
<td>PL</td>
<td>-</td>
<td>zero</td>
<td>Correct</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I_PU</td>
<td>PU</td>
<td>Non-select inventory</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I_PL</td>
<td>PL</td>
<td>Non-select inventory</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SI_E_PL</td>
<td>PL</td>
<td>Select EPC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SI_T_PL</td>
<td>PL</td>
<td>Select TID partial</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>---------</td>
<td>------</td>
<td>--------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SI_U_PL</td>
<td>PL</td>
<td>Select User partial</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6. PermaLocked_APZ Test Cases
### 2.2.7. PermaLocked_APNZ

<table>
<thead>
<tr>
<th>PermaLocked_APNZ.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Lock</th>
<th>Action</th>
<th>AP State</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>R_KP_PL_2</td>
<td>PL</td>
<td>Kill password; Access password non-zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>W_KP_PL_NZ_2</td>
<td>PL</td>
<td>Kill password non-zero; Access password non-zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>PL_KP_L_2</td>
<td>L</td>
<td>Kill password; Access password non-zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>U_KP_PL_2</td>
<td>PL</td>
<td>Kill password; Access password non-zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>L_KP_PL_2</td>
<td>PL</td>
<td>Kill password; Access password non-zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>PU_KP_PL_2</td>
<td>PL</td>
<td>Kill password; Access password non-zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>R_E_PL_P_2</td>
<td>PL</td>
<td>EPC partial; Access password non-zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>W_E_PL_P_2</td>
<td>PL</td>
<td>EPC partial; Access password non-zero</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>PL_E_L_2</td>
<td>L</td>
<td>EPC memory; Access password</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>non-zero</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>U_E_PL_2</td>
<td>PL</td>
<td>EPC memory; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>L_E_PL_2</td>
<td>PL</td>
<td>EPC memory; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>PU_E_PL_2</td>
<td>PL</td>
<td>EPC memory; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>R_T_PL_P_2</td>
<td>PL</td>
<td>TID partial; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>W_T_PL_P_2</td>
<td>PL</td>
<td>TID partial; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>PL_T_L_2</td>
<td>L</td>
<td>TID memory; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>U_T_PL_2</td>
<td>PL</td>
<td>TID memory; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>L_T_PL_2</td>
<td>PL</td>
<td>TID memory; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>PU_T_PL_2</td>
<td>PL</td>
<td>TID memory; Access password non-zero</td>
</tr>
<tr>
<td>Yes</td>
<td>R_U_P_PL_2</td>
<td>PL</td>
<td>User partial; Access password non-zero</td>
</tr>
<tr>
<td>User</td>
<td>User partial; Access password non-zero</td>
<td>User memory; Access password non-zero</td>
<td>User memory; Access password non-zero</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------</td>
<td>--------------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>Yes</td>
<td>W_U_PL_P_2</td>
<td>PL</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>PL_U_L_2</td>
<td>L</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>U_U_PL_2</td>
<td>PL</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>L_U_PL_2</td>
<td>PL</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>PU_U_PL_2</td>
<td>PL</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>R_AP_PL_1</td>
<td>PL</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>R_AP_PL_2</td>
<td>PL</td>
<td>None</td>
</tr>
<tr>
<td>Yes</td>
<td>W_AP_PL_NZ_1</td>
<td>PL</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>W_AP_PL_NZ_2</td>
<td>PL</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_PU_2</td>
<td>PU</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>U_AP_PL_2</td>
<td>PL</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>U_AP_PL_3</td>
<td>PL</td>
<td>None</td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_PL_2</td>
<td>PL</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>L_AP_PL_3</td>
<td>PL</td>
<td>None</td>
</tr>
<tr>
<td>Yes</td>
<td>PU_AP_PL_2</td>
<td>PL</td>
<td>Correct</td>
</tr>
<tr>
<td>Yes</td>
<td>PU_AP_PL_3</td>
<td>PL</td>
<td>None</td>
</tr>
</tbody>
</table>

Table 7. PermaLocked_APBZ Test Cases
2.2.8. Inventory Multiple

<table>
<thead>
<tr>
<th>Inventory_multiple.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>I_MH</td>
<td>Non-select inventory homogeneous</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>SI_MH_E</td>
<td>Select EPC complete homogeneous</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>SI_MH_T</td>
<td>Select TID complete homogeneous</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>SI_MH_U</td>
<td>Select User complete homogeneous</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>I_MM</td>
<td>Non-select inventory mixed</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>SI_MM_E</td>
<td>Select EPC complete mixed</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>SI_MM_T</td>
<td>Select TID complete mixed</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>SI_MM_U</td>
<td>Select User complete mixed</td>
</tr>
</tbody>
</table>

Table 8. Inventory Multiple Test Cases
### 2.2.9. Select/Query EPC Memory

<table>
<thead>
<tr>
<th>sq_epc_test.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Action</th>
<th>Pointer</th>
<th>Length</th>
<th>Truncate</th>
<th>Lock</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>SQ_E_S0_1</td>
<td>S0</td>
<td>0</td>
<td>32</td>
<td>96</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S0_2</td>
<td>S0</td>
<td>1</td>
<td>34</td>
<td>64</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S0_3</td>
<td>S0</td>
<td>10</td>
<td>37</td>
<td>48</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S0_4</td>
<td>S0</td>
<td>11</td>
<td>43</td>
<td>24</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S0_5</td>
<td>S0</td>
<td>100</td>
<td>55</td>
<td>12</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S0_6</td>
<td>S0</td>
<td>101</td>
<td>79</td>
<td>6</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S0_7</td>
<td>S0</td>
<td>110</td>
<td>95</td>
<td>3</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S0_8</td>
<td>S0</td>
<td>111</td>
<td>127</td>
<td>1</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S1_1</td>
<td>S1</td>
<td>0</td>
<td>117</td>
<td>1</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S1_2</td>
<td>S1</td>
<td>1</td>
<td>97</td>
<td>3</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S1_3</td>
<td>S1</td>
<td>10</td>
<td>83</td>
<td>6</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S1_4</td>
<td>S1</td>
<td>11</td>
<td>64</td>
<td>12</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S1_5</td>
<td>S1</td>
<td>100</td>
<td>46</td>
<td>24</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S1_6</td>
<td>S1</td>
<td>101</td>
<td>42</td>
<td>48</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S1_7</td>
<td>S1</td>
<td>110</td>
<td>38</td>
<td>64</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S1_8</td>
<td>S1</td>
<td>111</td>
<td>32</td>
<td>96</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S2_1</td>
<td>S2</td>
<td>0</td>
<td>32</td>
<td>96</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S2_2</td>
<td>S2</td>
<td>1</td>
<td>32</td>
<td>95</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S2_3</td>
<td>S2</td>
<td>10</td>
<td>44</td>
<td>48</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S2_4</td>
<td>S2</td>
<td>11</td>
<td>56</td>
<td>30</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S2_5</td>
<td>S2</td>
<td>100</td>
<td>67</td>
<td>20</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S2_6</td>
<td>S2</td>
<td>101</td>
<td>75</td>
<td>6</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S2_7</td>
<td>S2</td>
<td>110</td>
<td>127</td>
<td>0</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S2_8</td>
<td>S2</td>
<td>111</td>
<td>126</td>
<td>1</td>
<td>0</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S3_1</td>
<td>S3</td>
<td>0</td>
<td>124</td>
<td>1</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S3_2</td>
<td>S3</td>
<td>1</td>
<td>102</td>
<td>3</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S3_3</td>
<td>S3</td>
<td>10</td>
<td>87</td>
<td>6</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S3_4</td>
<td>S3</td>
<td>11</td>
<td>59</td>
<td>12</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S3_5</td>
<td>S3</td>
<td>100</td>
<td>45</td>
<td>24</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S3_6</td>
<td>S3</td>
<td>101</td>
<td>41</td>
<td>48</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S3_7</td>
<td>S3</td>
<td>110</td>
<td>36</td>
<td>64</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_S3_8</td>
<td>S3</td>
<td>111</td>
<td>32</td>
<td>96</td>
<td>0</td>
<td>L EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_1</td>
<td>SL</td>
<td>0</td>
<td>127</td>
<td>2</td>
<td>1</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_2</td>
<td>SL</td>
<td>1</td>
<td>63</td>
<td>64</td>
<td>1</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_3</td>
<td>SL</td>
<td>10</td>
<td>127</td>
<td>1</td>
<td>1</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_4</td>
<td>SL</td>
<td>11</td>
<td>63</td>
<td>0</td>
<td>1</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_5</td>
<td>SL</td>
<td>100</td>
<td>58</td>
<td>12</td>
<td>1</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_6</td>
<td>SL</td>
<td>101</td>
<td>81</td>
<td>16</td>
<td>1</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_7</td>
<td>SL</td>
<td>110</td>
<td>32</td>
<td>0</td>
<td>1</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>------------</td>
<td>----</td>
<td>-----</td>
<td>----</td>
<td>---</td>
<td>---</td>
<td>-----</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_8</td>
<td>SL</td>
<td>111</td>
<td>124</td>
<td>3</td>
<td>1</td>
<td>EPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_9</td>
<td>SL</td>
<td>0</td>
<td>32</td>
<td>96</td>
<td>1</td>
<td>L</td>
<td>EPC</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_10</td>
<td>SL</td>
<td>1</td>
<td>32</td>
<td>95</td>
<td>1</td>
<td>L</td>
<td>EPC</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_11</td>
<td>SL</td>
<td>10</td>
<td>44</td>
<td>48</td>
<td>1</td>
<td>L</td>
<td>EPC</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_12</td>
<td>SL</td>
<td>11</td>
<td>56</td>
<td>30</td>
<td>1</td>
<td>L</td>
<td>EPC</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_13</td>
<td>SL</td>
<td>100</td>
<td>67</td>
<td>20</td>
<td>1</td>
<td>L</td>
<td>EPC</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_14</td>
<td>SL</td>
<td>101</td>
<td>75</td>
<td>6</td>
<td>1</td>
<td>L</td>
<td>EPC</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_15</td>
<td>SL</td>
<td>110</td>
<td>127</td>
<td>0</td>
<td>1</td>
<td>L</td>
<td>EPC</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_E_SL_16</td>
<td>SL</td>
<td>111</td>
<td>126</td>
<td>1</td>
<td>1</td>
<td>L</td>
<td>EPC</td>
<td></td>
</tr>
</tbody>
</table>

Table 9. Select/Query EPC Test Cases
2.2.10 Select/Query TID Memory

<table>
<thead>
<tr>
<th>sq_tid_test.txt</th>
<th>Pass?</th>
<th>N/A?</th>
<th>Test case</th>
<th>Target</th>
<th>Action</th>
<th>Pointer</th>
<th>Length</th>
<th>Truncate</th>
<th>Lock</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>SQ_T_S0_1</td>
<td>S0</td>
<td>0</td>
<td>32</td>
<td>Full</td>
<td>0</td>
<td>TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_S0_2</td>
<td>S0</td>
<td>1</td>
<td>32</td>
<td>Half</td>
<td>0</td>
<td>TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_S1_1</td>
<td>S1</td>
<td>10</td>
<td>32</td>
<td>Full</td>
<td>0</td>
<td>L TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_S1_2</td>
<td>S1</td>
<td>11</td>
<td>32</td>
<td>Half</td>
<td>0</td>
<td>L TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_S2_1</td>
<td>S2</td>
<td>100</td>
<td>32</td>
<td>Full</td>
<td>0</td>
<td>TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_S2_2</td>
<td>S2</td>
<td>101</td>
<td>32</td>
<td>Half</td>
<td>0</td>
<td>TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_S3_1</td>
<td>S3</td>
<td>110</td>
<td>32</td>
<td>Full</td>
<td>0</td>
<td>L TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_S3_2</td>
<td>S3</td>
<td>111</td>
<td>32</td>
<td>Half</td>
<td>0</td>
<td>L TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_SL_1</td>
<td>SL</td>
<td>0</td>
<td>32</td>
<td>Full</td>
<td>0</td>
<td>TID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td>SQ_T_SL_2</td>
<td>SL</td>
<td>1</td>
<td>32</td>
<td>Half</td>
<td>0</td>
<td>TID</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table. 10 Select/Query TID Test Cases
3. TEST CAMPAIGN REPORT (Test Logs)
The abbreviations used in the header row of the test campaign report tables are:

Applicable: Indicates whether or not a test case have been selected for execution against the IUT identified in section XX according to the analysis of the information in the ICS and IXIT for the IUT.

Executed: Indicate whether or not the corresponding test case has been run to completion.

Verdict: Records the verdict assigned to each test case run to completion. Following verdicts are possible:
  Pass: If the test case passed
  Fail: If the test case failed
  Inc: If the test case is inconclusive.

Observations: Provides a reference to additional information relevant to the test presented in section 7.

Access Memory

# Created by Impinj and modified by MET Laboratories for use in EPCglobal sanctioned Gen2 RFID interoperability testing
# Permission granted for free use only for EPCglobal sponsored interoperability testing activities
#
# EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable for any
# injury, loss, damages, financial or otherwise, arising from, related to, or caused by the use of this
# document. Use of said document does not guarantee compliance with applicable state and/or federal laws.
# User is responsible for the interpretation of and compliance with applicable pedigree laws. The use of
# said document shall constitute your express consent to the foregoing disclaimer.

connect 169.254.10.1 FCC 201
  Connected to reader 169.254.10.1, type: FCC, mode: 201
power 20.0
  Power set to 20.0
# This section initializes the tag
#
Write pass 00000000 apass
  Write result: 1
  Write SUCCESSFUL
Write pass 00000000 kpass
  Write result: 1
  Write SUCCESSFUL
#
# End of initialize tag

# === Access Password Section ===
#
# Access password only matters if it non-zero and the memory is locked.
#
# Tag is initially set with all zero AP and KP and unlocked memory
# AP = access password; U/L is unlocked/locked memory; TC = Test Case

# Running the unlocked Access Commands here
#
Read pass 00000000 apass
  Read data: 00000000
  Read SUCCESSFUL
# AP = 00000000, U; Verifies AP is all zeros; TC = R_AP_1
# This TC reads the correct AP when the AP is zero.

Read fail 00000000 apass EEEEEEEE
  Read data:
      Read SUCCESSFUL
# AP = 00000000, U; AP read w/ incorrect AP; TC = R_AP_2
# This TC reads the correct AP when the AP is zero, using an incorrect AP.

Lock fail apass DDDDDDDD
  Lock result: 0
  Lock SUCCESSFUL
# AP = 00000000, U; Attempted lock w/ incorrect AP; TC = L_AP_2
# This TC attempts to lock the AP in the zero state using an incorrect AP.

WrRd pass 11111111 apass
  WrRd Write result: 1
  WrRd read result: 11111111
  WrRd SUCCESSFUL
# AP = 11111111, U; Set initial AP; TC = W_AP_NZ_1, R_AP_3
# This TC writes and then reads a non-zero AP, using a correct all-zero AP.

Read pass 11111111 apass
   Read data: 11111111
   Read SUCCESSFUL
# AP = 11111111, U; Read AP from open state; TC = R_AP_5
# This TC attempts to read a non-zero AP using no AP.

Lock fail apass
   Lock result: 0
   Lock SUCCESSFUL
# AP = 11111111, U; Attempted Lock w/ nonzero AP; TC = L_AP_5
# This TC attempts to lock the AP from the open state, when the AP is non-zero and no AP is given.

Write pass 22222222 apass 11111111
   Write result: 1
   Write SUCCESSFUL
Read pass 22222222 apass 22222222
   Read data: 22222222
   Read SUCCESSFUL
# AP = 22222222, U; Verifies AP change after incorrect AP operation; TC = W_AP_NZ_2
# This TC writes and then reads a non-zero AP, using another correct non-zero AP.

Write pass 33333333 apass
   Write result: 1
   Write SUCCESSFUL
# AP = 33333333, U; Verifies AP change in open state; TC = W_AP_NZ_4
# This TC writes a non-zero AP, using no AP, when the AP is non-zero.

Read fail 44444444 apass 44444444
   Read data:
   Read SUCCESSFUL
# AP = 33333333, U; Attempted Read of AP w/ incorrect AP; TC = R_AP_4
# This TC reads the AP using an incorrect non-zero AP.

Write fail 99999999 apass FFFFFFF
   Write result: 0
   Write SUCCESSFUL
Read pass 33333333 apass
   Read data: 33333333
   Read SUCCESSFUL
# AP = 33333333, U; Attempted AP change w/ incorrect AP; TC = W_AP_NZ_3
# This TC attempts to write a non-zero AP, using an incorrect non-zero AP. It reads the correct AP from the open state.

Write pass 00000000 apass 33333333
   Write result: 1
   Write SUCCESSFUL
Read pass 00000000 apass 00000000
   Read data: 00000000
   Read SUCCESSFUL
# AP = 00000000, U; AP changed to all zeros; TC = W_AP_Z
# This TC writes a zero AP, using a correct non-zero AP. It then reads that it is correct.

# Running the Locked Access Commands here

Lock pass apass
   Lock result: 1
   Lock SUCCESSFUL
# AP = 00000000, L; Verifies Lock operation; TC = L_AP_1
# This TC locks the AP from the secured state, when the AP is zero.

Read pass 00000000 apass
   Read data: 00000000
   Read SUCCESSFUL
# AP = 00000000, L; Verifying all zero AP; TC = R_AP_L_1
# This TC reads the locked AP when the AP is zero and correct.

Read fail 00000000 apass CCCCCCCC
   Read data:
      Read SUCCESSFUL
# AP = 00000000, L; Attempted Read w/ incorrect AP; TC = R_AP_L_2
# This TC attempts to read the locked AP, when the AP is zero, using an incorrect AP.

Write fail 44444444 apass BBBBBBBB
   Write result: 0
   Write SUCCESSFUL
Read pass 00000000 apass
   Read data: 00000000
   Read SUCCESSFUL
# AP = 00000000, L; Attempted Write w/ incorrect AP; TC = W_AP_L_NZ_2
# This TC attempts to write a non-zero AP, when the AP is locked and zero, using an incorrect AP.
# The AP is read as unchanged.
Write pass 11111111 apass 00000000
  Write result: 1
  Write SUCCESSFUL
Read pass 11111111 apass 11111111
  Read data: 11111111
  Read SUCCESSFUL
# AP = 11111111, L; Change AP in Locked State; TC = W_AP_L_NZ_1, R_AP_L_3
# This TC writes a non-zero AP, when the AP is locked and zero, using the correct AP.
# The AP is read as changed.

Read fail 11111111 ap
  Read data:

    Read SUCCESSFUL
# AP = 11111111, L; Attempting read w/o AP (open state read); TC = R_AP_L_5
# This TC attempts to read the correct AP, when the AP is locked and non-zero, using no AP.
# This TC fails because the read can't be done from the open state.

Read fail 33333333 apass AAAAAAAA
  Read data:

    Read SUCCESSFUL
# AP = 11111111, L; Attempted Read w/ incorrect AP; TC = R_AP_L_4
# This TC reads the incorrect AP, when the AP is locked and non-zero, using an incorrect AP.

Write pass 22222222 apass 11111111
  Write result: 1
  Write SUCCESSFUL
Read pass 22222222 apass 22222222
  Read data: 22222222
  Read SUCCESSFUL
# AP = 22222222, L; Verifies AP change in locked state; TC = W_AP_L_NZ_3
# This TC writes to the AP, when the AP is locked and non-zero, using the correct AP.

Write fail 66666666 apass 55555555
  Write result: 0
  Write SUCCESSFUL
Read pass 22222222 apass 22222222
  Read data: 22222222
  Read SUCCESSFUL
# AP = 22222222, L; Attempted Write w/ incorrect AP; TC = W_AP_L_NZ_4
# This TC attempts to write to the AP, when the AP is locked and non-zero, using an incorrect AP.

Report No.: EMC94348B Rev. 2

Date: 2017-12-08  Page 26 of 114
Write fail 66666666 apass
  Write result: 0
  Write SUCCESSFUL
Read pass 22222222 apass 22222222
  Read data: 22222222
  Read SUCCESSFUL
# AP = 22222222, L; Attempted Write to locked memory in open state; TC = W_AP_L_NZ_5
# This TC attempts to write to the AP, when the AP is locked and non-zero, using no AP.

#
# Run the Unlock Commands here
#

Unlock fail apass AAAAAAAA
  UnLock result: 0
  UnLock SUCCESSFUL
# AP = 22222222, L; Attempted Unlock w/ incorrect AP; TC = U_AP_L_4
Write fail 33333333 apass
  Write result: 0
  Write SUCCESSFUL
# AP = 22222222, L; Verifies cannot write from open state
Read fail 22222222 apass
  Read data:
  Read SUCCESSFUL
# AP = 22222222, L; Verifies cannot read from open state
Read pass 22222222 apass 22222222
  Read data: 22222222
  Read SUCCESSFUL
# AP = 22222222, L; Verifies read from secured state
# This TC attempts to unlock the AP, when the AP is locked and non-zero, using an incorrect AP.
Unlock pass apass 22222222
  UnLock result: 1
  UnLock SUCCESSFUL
# AP = 22222222, U; Successful Unlock w/ correct AP; TC = U_AP_L_3
WrRd pass 33333333 apass
  WrRd Write result: 1
  WrRd read result: 33333333
  WrRd SUCCESSFUL
# AP = 33333333, U; Verifies can write and read from open state
Write pass 44444444 apass 33333333
  Write result: 1
  Write SUCCESSFUL
Read pass 44444444 apass 44444444
  Read data: 44444444
  Read SUCCESSFUL
# AP = 44444444, U; Verifies can write and read from secured state
# This TC unlocks the AP, when the AP is formerly locked and non-zero, using the correct AP.
# It also verifies the AP is unlocked.

Lock pass apass 44444444
  Lock result: 1
  Lock SUCCESSFUL
# AP = 44444444, L; Verifies lock from secured state; TC = L_AP_3
Write fail 33333333 apass
  Write result: 0
  Write SUCCESSFUL
# AP = 44444444, L; Verifies cannot write from open state
Read fail 44444444 apass
  Read data:
  Read SUCCESSFUL
# AP = 44444444, L; Verifies read from secured state
# This TC locks the AP, when the AP is unlocked and non-zero, using the correct AP.
# Both open and secured states are verified.

Unlock pass apass 44444444
  UnLock result: 1
  UnLock SUCCESSFUL
# AP = 44444444, U; Unlocks AP for the next test.

Lock fail apass 42424242
  Lock result: 0
  Lock SUCCESSFUL
# AP = 44444444, U; Verifies lock with incorrect password; TC = L_AP_4
WrRd pass 33333333 apass
  WrRd Write result: 1
  WrRd read result: 33333333
  WrRd SUCCESSFUL
# AP = 33333333, U; Verifies can write and read from open state
Write pass 44444444 apass 33333333
  Write result: 1
  Write SUCCESSFUL
Read pass 44444444 apass 44444444
  Read data: 44444444
  Read SUCCESSFUL
# AP = 44444444, U; Verifies can write and read from secured state
# This TC attempts to lock the AP, when the AP is unlocked and non-zero, using an incorrect AP.

Write pass 00000000 apass 44444444
  Write result: 1
  Write SUCCESSFUL
# AP = 00000000, U; Reset AP to all zeros
Read pass 00000000 apass
  Read data: 00000000
  Read SUCCESSFUL
# AP = 00000000, U; Verifies AP is all zeros
Lock pass apass 00000000
  Lock result: 1
  Lock SUCCESSFUL
# AP = 00000000, L; Locks AP

Unlock pass apass 00000000
  UnLock result: 1
  UnLock SUCCESSFUL
# AP = 00000000, U; Locks AP to all zeros; TC = U_AP_L_1
WrRd pass 33333333 apass
  WrRd Write result: 1
  WrRd read result: 33333333
  WrRd SUCCESSFUL
# AP = 33333333, U; Verifies can write and read from open state
Write pass 44444444 apass 33333333
  Write result: 1
  Write SUCCESSFUL
Read pass 44444444 apass 44444444
  Read data: 44444444
  Read SUCCESSFUL
# AP = 44444444, U; Verifies can write and read from secured state
# This TC unlocks the AP, when the AP is locked and zero, using the correct zero AP.

Write pass 00000000 apass 44444444
  Write result: 1
  Write SUCCESSFUL
# AP = 00000000, U; Reset AP to all zeros

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
Read pass 00000000 apass
  Read data: 00000000
  Read SUCCESSFUL
# AP = 00000000, U; Verifies AP all zeros
Lock pass apass 00000000
  Lock result: 1
  Lock SUCCESSFUL
# AP = 00000000, L; Locks the AP

Unlock fail apass BBBBBBBB
  UnLock result: 0
  UnLock SUCCESSFUL
# AP = 00000000, L; Unsuccessful unlock w/ incorrect AP; TC = U_AP_L_2
Write pass 11111111 apass 00000000
  Write result: 1
  Write SUCCESSFUL
# AP = 11111111, L; Verifies Write from secured state (dropped into secured state because AP is all zeros)
Read pass 11111111 apass 11111111
  Read data: 11111111
  Read SUCCESSFUL
# AP = 11111111, L; Verifies read from secured state
# This TC attempts to unlock the AP, when the AP is locked and zero, using an incorrect AP.

# At the end of the script, the AP should be locked and the AP should be 0x11111111.

disconnect
  Disconnected.
Script completed without failures

**Inventory Single**
# Created by Impinj and modified by MET Laboratories for use in EPCglobal sanctioned Gen2 RFID interoperability testing
# Permission granted for free use only for EPCglobal sponsored interoperability testing activities
#
# EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable for any
# injury, loss, damages, financial or otherwise, arising from, related to, or caused by the use of this
# document. Use of said document does not guarantee compliance with applicable state and/or federal laws.

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
# User is responsible for the interpretation of and compliance with applicable pedigree laws. The use of said document shall constitute your express consent to the foregoing disclaimer.

connect 169.254.10.1 FCC 201
  Connected to reader 169.254.10.1, type: FCC, mode: 201
power 20.0
  Power set to 20.0

# This section initializes the tag
WrRd pass 340000000000000240601FB8D epc
  Write result: 1
  WrRd read result: 340000000000000240601FB8D
  WrRd SUCCESSFUL
# WrRd pass E280B0A1 tid
#
# Initialize TID if it is not Permalocked
#
Write pass 00000000 apass
  Write result: 1
  Write SUCCESSFUL
#
# WrRd pass 0000111122223333 user
# Initialize user memory if available on chip
# Requires tag vendor customization as this feature is optional in Gen2
# This is an example syntax - total memory varies by vendor
#
# End of initialize tag

# === Inventory Section ===
#
# Access password only matters if it non-zero and the memory is locked.
#
# Tag is initially set with all zero AP and KP and unlocked memory
# EPC = 340000000000000240601FB8D
# AP = access password; U/L is unlocked/locked memory; TC = Test Case
#
Read pass E280B0A1 tid
  Read data: E280B0A1
  Read SUCCESSFUL
# TID = E2801050; Verify TID for tags w/ Permalock TID

# Now do inventory cases with unlocked memory locations
Inventory pass 1
Inventoried 1 unique tags
  00000000000000240601fb8d :1
  Inventory SUCCESSFUL

# EPC = 340000000000000000240601FB8D, U; AP = 00000000; TC = I
Read pass 340000000000000000240601FB8D epc
  Read data: 340000000000000000240601FB8D
  Read SUCCESSFUL
# EPC = 340000000000000000240601FB8D, U; AP = 00000000; Verifies EPC
# Inventory pass 1 340000000000000000240601FB8D epc
# This TC does a non-select inventory of the tag and reads its EPC number.

# EPC = 340000000000000000240601FB8D, U; AP = 00000000; TC = SI_E
Read pass 340000000000000000240601FB8D epc
  Read data: 340000000000000000240601FB8D
  Read SUCCESSFUL
# EPC = 340000000000000000240601FB8D, U; AP = 00000000; Verifies EPC
# This TC does a complete selection of the single tag's EPC number.

# Inventory pass 1 E280B0A1 tid
Inventoried 1 unique tags
  00000000000000240601fb8d :1
  Inventory SUCCESSFUL
# TID = E2801050, U; AP = 00000000; TC = SI_T
Read pass E280B0A1 tid
  Read data: E280B0A1
  Read SUCCESSFUL
# TID = E2801050, U; AP = 00000000; TC = SI_T; Verifies TID
# This TC does a complete selection of the single tag's TID number.

# This section is for tags with user memory
# Currently this functionality has not been verified
# Remove the comment lines on Inventory and Read directives to test the functionality
# This is an example syntax - total memory varies by vendor
#
#XXX Inventory pass 1 0000111122223333 user
# USER = 0000111122223333, U; AP = 00000000; TC = SI_U
#XXX Read pass 0000111122223333 user
# USER = 0000111122223333, U; AP = 00000000; Verifies EPC
# This TC does a complete selection of the single tag's user data.
#
#
# Now perform Inventory cases with memory locked

Write pass 11111111 apass
   Write result: 1
   Write SUCCESSFUL
Lock pass epc 11111111
   Lock result: 1
   Lock SUCCESSFUL
   # EPC = E28033B2DDD9014035050000, L; AP = 11111111
Write fail 3400000011112222333344445555 epc
   Write result: 0
   Write SUCCESSFUL
   # Verified the EPC memory is locked and not writable
Lock pass tid 11111111
   Lock result: 1
   Lock SUCCESSFUL
   # TID = E2801050, L; AP = 11111111;
Write fail F3112161 tid
   Write result: 0
   Write SUCCESSFUL
   # Verifies the TID is locked or Permalocked
   #
   # This section is for tags with user memory
   # Currently this functionality has not been verified
   # Remove the comment lines on Inventory and Read directives to test the functionality
   # This is an example syntax - total memory varies by vendor
   #
   #XXX Lock pass user 11111111
   # USER = 0000111122223333, L; AP = 11111111;
   #XXX
   #XXX Write fail FFFFFFFFFFFFFFFFFF user

Inventory pass 1
   Inventoried 1 unique tags
   00000000000000240601fb8d :1
   Inventory SUCCESSFUL
   # EPC = 340000000000000000240601FB8D, L; AP = 11111111; TC = I_L
Read pass 3400000000000000000240601FB8D epc 11111111
   Read data: 3400000000000000000240601FB8D
   Read SUCCESSFUL
   # EPC = 340000000000000000240601FB8D, L; AP = 11111111; Verifies EPC
   # This TC inventories the single tag while the AP, epc, and TID are locked/permalocked.

Inventory pass 1 0000 epc

Report No.: EMC94348B Rev. 2
Date: 2017-12-08  Page 33 of 114
Inventoried 1 unique tags
00000000000000240601fb8d :1
Inventory SUCCESSFUL
# EPC = 340000000000000000240601FB8D, L; AP = 11111111; TC = SI_E_L
Read pass 340000000000000000240601FB8D epc 11111111
   Read data: 340000000000000000240601FB8D
   Read SUCCESSFUL
# EPC = 340000000000000000240601FB8D, L; AP = 11111111; Verifies EPC
# This TC selects a portion of the EPC for a single tag while the AP, epc, and TID are locked/permalocked.

Inventory pass 1 E280 tid
    Inventoried 1 unique tags
    00000000000000240601fb8d :1
    Inventory SUCCESSFUL
# TID = E2801050, L; AP = 00000000; TC = SI_T_L
Read pass E280B0A1 tid
   Read data: E280B0A1
   Read SUCCESSFUL
# TID = E2801050, L; AP = 00000000; Verifies TID
# This TC selects a portion of the TID for a single tag while the AP, epc, and TID are locked/permalocked.

# This section is for tags with user memory
# Currently this functionality has not been verified
# Remove the comment lines on Inventory and Read directives to test the functionality
# # Inventory pass 1 00001111 user
# # USER = 0000111122223333, L; AP = 00000000; TC = SI_U_L
# # Read pass 0000111122223333 user,0,4
# # USER = 0000111122223333, L; AP = 00000000; Verifies user memory
#

disconnect
    Disconnected.
Script completed without failures

**Write and Read**
# Created by Impinj and modified by MET Laboratories for use in EPCglobal sanctioned Gen2 RFID interoperability testing
# Permission granted for free use only for EPCglobal sponsored interoperability testing activities
# EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable for any injury, loss, damages, financial or otherwise, arising from, related to, or caused by the use of this document. Use of said document does not guarantee compliance with applicable state and/or federal laws. User is responsible for the interpretation of and compliance with applicable pedigree laws. The use of said document shall constitute your express consent to the foregoing disclaimer.

connect 169.254.10.1 FCC 201
    Connected to reader 169.254.10.1, type: FCC, mode: 201
    power 20.0
    Power set to 20.0

# This section initializes the tag
WrRd pass 3400E2800000000000240601FB8D epc
    WrRd Write result: 1
    WrRd read result: 3400E2800000000000240601FB8D
    WrRd SUCCESSFUL
#
# TTT WrRd pass E280B0A1 tid
# Initialize TID if it is not Permalocked
#
Write pass 00000000 apass
    Write result: 1
    Write SUCCESSFUL
#
# WrRd pass 0000111122223333 user
# Initialize user memory if available on chip
# Requires tag vendor customization as this feature is optional in Gen2
# This is an example syntax - total memory varies by vendor
#
# End of initialize tag

# === Write and Read Test Section ===
# ===  Section tests EPC memory  ===
#
# Access password only matters if it non-zero and the memory is locked.
#
# Tag is initially set with all zero AP and KP and unlocked memory
# EPC = 3400E2800000000000240601FB8D
# AP = access password; U/L is unlocked/locked memory; TC = Test Case
#
Read pass E280B0A1 tid
Read data: E280B0A1
Read SUCCESSFUL
# TID = E2001050; Verify TID for tags w/ Permalock TID

WrRd pass 3400E28000000000000240601FB8D epc
    WrRd Write result: 1
    WrRd read result: 3400E28000000000000240601FB8D
    WrRd SUCCESSFUL
# EPC = 3400E28000000000000240601FB8D, U; TC = W_E_C, R_E_C
# These TCs write and then read the complete EPC number.

WrRd pass E280 epc.2
    WrRd Write result: 1
    WrRd read result: E280
    WrRd SUCCESSFUL
# EPC = 3400E28033B2DDD9014035050000, U; TC = W_E_P, R_E_P
# These TCs write and then read a portion of the EPC number.

#
# Now run test cases for locked EPC

Write pass 11111111 apass
    Write result: 1
    Write SUCCESSFUL
# EPC = 3400E28033B2DDD9014035050000, U; AP = 11111111; Sets AP
Lock pass epc 11111111
    Lock result: 1
    Lock SUCCESSFUL
# EPC = 3400E28033B2DDD9014035050000, L; AP = 11111111; TC = L_E
# This TC locks the epc number.

Write fail 3400E28000000000000240601FB8D epc
    Write result: 0
    Write SUCCESSFUL
# EPC = 3400E28033B2DDD9014035050000, L; AP = 11111111; Attempted Write from open state;
Write fail 3200E28033B2DDD9014035050000 epc FFFFFFFF
    Write result: 0
    Write SUCCESSFUL
# EPC = 3400E28033B2DDD9014035050000, L; AP = 11111111; Attempted Write incorrect AP;
WrRd pass E280 epc.2 11111111
    WrRd Write result: 1
    WrRd read result: E280
    WrRd SUCCESSFUL
# EPC = 3400E28000000000000240601FB8D, L; AP = 11111111; TC = W_E_L_P and R_E_L_P;
# This TC writes and then reads a portion of the EPC from the secured state, while the EPC memory is locked.

#
# Now run test cases to unlock EPC

Unlock fail epc
  UnLock result: 0
  UnLock SUCCESSFUL
# EPC = 3400E28000000000000240601FB8D, L; AP = 11111111; Attempted Unlock w/o AP
Write fail 3400E28033B2DDD9014035050000 epc
  Write result: 0
  Write SUCCESSFUL
# EPC = 3400E28000000000000240601FB8D, L; AP = 11111111; Attempted Write from open state;
Write fail 3200E28033B2DDD9014035050000 epc FFFFFFFF
  Write result: 0
  Write SUCCESSFUL
# EPC = 3400E2
  # EPC = 3400E28000000000000240601FB8D, L; AP = 11111111; Attempted Write incorrect AP;
  Unlock fail epc FFFFFFFF
  UnLock result: 0
  UnLock SUCCESSFUL
# EPC = 3400E28000000000000240601FB8D, L; AP = 11111111; Attempted Unlock w/ incorrect AP
Write fail 3400E28033B2DDD9014035050000 epc
  Write result: 0
  Write SUCCESSFUL
# EPC = 3400E28000000000000240601FB8D, L; AP = 11111111; Attempted Write from open state;
Write fail 3400E28033B2DDD9014035050000 epc FFFFFFFF
  Write result: 0
  Write SUCCESSFUL
# EPC = 3400E28000000000000240601FB8D, L; AP = 11111111; Attempted Write incorrect AP;
Unlock pass epc 11111111
  UnLock result: 1
  UnLock SUCCESSFUL
# EPC = 3400E28000000000000240601FB8D, U; AP = 11111111; TC = U_E_L
WrRd pass E280 epc,2 11111111
 WrRd Write result: 1
 WrRd read result: E280
 WrRd SUCCESSFUL
 # EPC = 3400E28033B2DDD9014035050000, U; AP = 11111111; Write from secured state;
 WrRd pass E280 epc.2
    WrRd Write result: 1
    WrRd read result: E280
    WrRd SUCCESSFUL
 # EPC = 3400E280000000000000240601FB8D, U; AP = 11111111; Write from open state;
 # This TC unlocks the EPC memory from the locked state.

 WrRd pass 00000000 apass
    WrRd Write result: 1
    WrRd read result: 00000000
    WrRd SUCCESSFUL
 # EPC = 3400E280000000000000240601FB8D, U; AP = 00000000; Resets AP

 # === Write and Read Test Section ===
 # === Section tests TID memory ===
 #
 # Access password only matters if it non-zero and the memory is locked.
 #
 # Tag is initially set with all zero AP and KP and unlocked memory
 # Assumes a 32-bit programmable TID
 # TID = E2001050
 # AP = access password; U/L is unlocked/locked memory; TC = Test Case
 #
 # NOTE: This section needs to be modified for each tag with a different TID
 #
 Read pass E280B0A1 tid
    Read data: E280B0A1
    Read SUCCESSFUL
 # TID = E2001050; Verify TID for tags w/ Permalock TID

 #TTT WrRd pass E280B0A1 tid
 # TID = E2001050, U; TC = W_T_C, R_T_C
 # Not valid for permalocked TID
 # This TC writes and then reads the complete TID.

 #TTT WrRd pass 2222 tid,2
 # TID = E2002222, U; TC = W_T_P, R_T_P
 # Not valid for permalocked TID
# This TC writes and then reads a portion of the TID.

# Now run test cases for locked TID

Write pass 11111111 apass
   Write result: 1
   Write SUCCESSFUL
# TID = E2001050, U; AP = 11111111; Sets AP
Lock pass tid 11111111
   Lock result: 1
   Lock SUCCESSFUL
# TID = E2001050, L; AP = 11111111; TC = L_T
Write fail E280B0A1 tid
   Write result: 0
   Write SUCCESSFUL
# TID = E2001050, L; AP = 11111111; AP = 11111111; Attempted Write from open state;
Write fail E280B0A1 tid FFFFFFFF
   Write result: 0
   Write SUCCESSFUL
# TID = E2001050, L; AP = 11111111; Attempted Write incorrect AP;
# TID = E2001050, L; AP = 11111111; ATTEMPTED Write incorrect AP;
# This TC locks the TID memory.

#TTT WrRd pass 3333 tid,2 11111111
# TID = E2003333, L; AP = 11111111; TC = W_T_L_P and R_T_L_P;
# Not valid for permalocked TID
# This TC writes and then reads a portion of the TID while the TID is locked.

# Now run test cases to unlock TID

Unlock fail tid
   UnLock result: 0
   UnLock SUCCESSFUL
# TID = E2001050, L; AP = 11111111; Attempted Unlock w/o AP
Write fail E2004444 tid
   Write result: 0
   Write SUCCESSFUL
# TID = E2001050, L; AP = 11111111; Attempted Write from open state;
Write fail E2004444 tid FFFFFFFF
   Write result: 0
   Write SUCCESSFUL
# TID = E2001050, L; AP = 11111111; Attempted Write incorrect AP;
Unlock fail tid FFFFFFFF
  UnLock result: 0
  UnLock SUCCESSFUL
# TID = E2001050, L; AP = 11111111; Attempted Unlock w/ incorrect AP
Write fail E20044444 tid
  Write result: 0
  Write SUCCESSFUL
# TID = E2001050, L; AP = 11111111; Attempted Write from open state;
Write fail E20044444 tid FFFFFFFF
  Write result: 0
  Write SUCCESSFUL
# TID = E2001050, L; AP = 11111111; Attempted Write incorrect AP;

## TID is Permalocked
Unlock fail tid 11111111
  UnLock result: 0
  UnLock SUCCESSFUL
# TID = E2001050, U; AP = 11111111; TC = U_T_L
# WrRd pass 1111 tid,2 11111111
# TID = E2001111, U; AP = 11111111; Write from secured state;
# WrRd pass 0000 tid,2
# TID = E2000000, U; AP = 11111111; Write from open state;
# This TC attempts to unlock the TID from the locked/permalocked state.
# It fails because the TID is permalocked, and therefore the tag doesn't allow the reader
to deselect its Lock/Action bits.

WrRd pass 00000000 apass
  WrRd Write result: 1
  WrRd read result: 00000000
  WrRd SUCCESSFUL
# TID = E2001050, U; AP = 00000000; Resets AP

# === Write and Read Test Section ===
# === Section tests USER memory ===
#
# Access password only matters if it non-zero and the memory is locked.
#
# Tag is initially set with all zero AP and KP and unlocked memory
# Assumes a 64-bit programmable USER
# USER = 0000111122223333
# AP = access password; U/L is unlocked/locked memory; TC = Test Case
#
# Note: This entire section requires tag vendor customization as this feature is optional
in Gen2

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
# This is an example syntax - total memory varies by vendor

#UuU WrRd pass 000111122223333 user
# USER = 0000111122223333, U; TC = W_U_C, R_U_C

#UuU WrRd pass 4444 user,4
# USER = 0000111122224444, U; TC = W_U_P, R_U_P

#
# Now run test cases for locked USER

Write pass 11111111 apass
    Write result: 1
    Write SUCCESSFUL
# USER = 0000111122224444, U; AP = 11111111; Sets AP
#UuU Lock pass user 11111111
# USER = 0000111122224444, L; AP = 11111111; TC = L_U
#Write fail 0000111122223333 user
# USER = 0000111122223333, L; AP = 11111111; AP = 11111111; Attempted Write from open state;
#Write fail 0000111122223333 user FFFFFFFF
# USER = 0000111122224444, L; AP = 11111111; Attempted Write incorrect AP;
#UuU WrRd pass 3333 user,4 11111111
# USER = 0000111122223333, L; AP = 11111111; TC = W_U_L_P and R_U_L_P;

#
# Now run test cases to unlock USER

#UuU Unlock fail user
# USER = 0000111122223333, L; AP = 11111111; Attempted Unlock w/o AP
#Write fail 0000111122224444 user
# USER = 0000111122223333, L; AP = 11111111; Attempted Write from open state;
#Write fail 0000111122224444 user FFFFFFFF
# USER = 0000111122224444, L; AP = 11111111; Attempted Write incorrect AP;

#UuU Unlock fail user FFFFFFFF
# USER = 0000111122223333, L; AP = 11111111; Attempted Unlock w/ incorrect AP
#Write fail 0000111122224444 user
# USER = 0000111122223333, L; AP = 11111111; Attempted Write from open state;
#Write fail 0000111122224444 user FFFFFFFF
# USER = 0000111122223333, L; AP = 11111111; Attempted Write incorrect AP;

#UuU Unlock pass user 11111111
# USER = 0000111122223333, L; AP = 11111111; TC = U_U_L
#UuU WrRd pass 4444 user,4 11111111

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
# USER = 0000111122224444, U; AP = 11111111; Write from secured state;
# UUU WrRd pass 3333 user,4
# USER = 0000111122223333, U; AP = 11111111; Write from open state;

# 
# Now run test cases for Kill

Write pass 00000000 kpass
Write result: 1
Write SUCCESSFUL
# KP = 00000000, U; AP = 11111111; TC = W_KP_Z
# This TC writes to the KP when the KP is all zeros.

Read pass 00000000 kpass
Read data: 00000000
Read SUCCESSFUL
# KP = 00000000, U; AP = 11111111; TC = R_KP
# This TC reads the KP when the KP is unlocked.

Write pass 01234567 kpass
Write result: 1
Write SUCCESSFUL
# KP = 01234567, U; AP = 11111111; TC = W_KP_NZ
Read pass 01234567 kpass
Read data: 01234567
Read SUCCESSFUL
# KP = 01234567, U; AP = 11111111;
# This TC writes a non-zero KP to the KP memory.

Lock pass kpass 11111111
Lock result: 1
Lock SUCCESSFUL
# KP = 01234567, L; AP = 11111111; Locks the kill password; TC = L_KP
Read fail 01234567 kpass
Read data:

Read SUCCESSFUL
# KP = 01234567, L; AP = 11111111; Attempt to read KP from open state
Write fail 33333333 kpass
Write result: 0
Write SUCCESSFUL
# KP = 01234567, L; AP = 11111111; Verifies cannot write from open state
Read fail 01234567 kpass 44444444
Read data:
Read SUCCESSFUL
# AP = 44444444, L; AP = 11111111; Verifies cannot read w/ incorrect AP
# This TC locks the KP.

Read pass 01234567 kpass 11111111
   Read data: 01234567
   Read SUCCESSFUL
# KP = 01234567, L; AP = 11111111; Verifies read from secured state; TC = R_KP_L
# This TC reads the KP when the KP is locked.

Write pass 76543210 kpass 11111111
   Write result: 1
   Write SUCCESSFUL
# KP = 76543210, L; AP = 11111111; Verifies write from secured state; TC = W_KP_L_NZ
Read pass 76543210 kpass 11111111
   Read data: 76543210
   Read SUCCESSFUL
# KP = 76543210, L; AP = 11111111; Verifies read from secured state;
# This TC writes to the KP when the KP is locked and non-zero.

Kill fail 01234567
   Kill result: 0
   Kill SUCCESSFUL
# KP = 76543210, L; AP = 11111111; TC = K_INZ_L
# This TC attempts to kill the tag with an incorrect, non-zero KP, while the KP is locked.

Unlock pass kpass 11111111
   UnLock result: 1
   UnLock SUCCESSFUL
# KP = 76543210 , U; AP = 11111111; TC = U_KP_L
# This TC unlocks the KP when it was locked.

Kill fail FFFFFFFF
   Kill result: 0
   Kill SUCCESSFUL
# KP = 76543210 , U; AP = 11111111; TC = K_INZ
Read pass 76543210 kpass
   Read data: 76543210
   Read SUCCESSFUL
# KP = 76543210 , U; AP = 11111111; Verifies tag is still alive
# This TC attempts to kill the tag using an incorrect, non-zero KP, while the KP is unlocked.
Write pass 00000000 kpass 11111111
Write result: 1
Write SUCCESSFUL
# KP = 00000000, U; AP = 11111111; Sets KP to all zeros
Kill fail 00000000
Kill result: 0
Kill SUCCESSFUL
# KP = 00000000 , U; AP = 11111111; TC = K_Z
# This TC attempts to kill the tag using an all-zero KP.

# At the end of this script, the tag is unlocked with an AP of 0x11111111 and a KP of 0x00000000

disconnect
Disconnected.
Script completed without failures

**PermaUnlocked**

# Created by Impinj and modified by MET Laboratories for use in EPCglobal sanctioned Gen2 RFID interoperability testing
# Permission granted for free use only for EPCglobal sponsored interoperability testing activities
# # EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable for any
# injury, loss, damages, financial or otherwise, arising from, related to, or caused by the use of this
# document. Use of said document does not guarantee compliance with applicable state and/or federal laws.
# User is responsible for the interpretation of and compliance with applicable pedigree laws. The use of
# said document shall constitute your express consent to the foregoing disclaimer.

connect 169.254.10.1 FCC 201
Connected to reader 169.254.10.1, type: FCC, mode: 201
power 20
Power set to 20

# # This script verifies PermaUnlock behavior with non-zero AP # # # "unlocked" means that the bit pattern to the associated memory bank is "00"
# "locked" that the bit pattern to the associated memory bank is "10"
# "permaunlocked" means that the bit pattern to the associated memory bank is "01"
# "permalocked" means that the bit pattern to the associated memory bank is "11"

Write pass 00000000 apass
  Write result: 1
  Write SUCCESSFUL
Read pass 00000000 apass
  Read data: 00000000
  Read SUCCESSFUL
Write pass AAAAAAAA kpass
  Write result: 1
  Write SUCCESSFUL
Read pass AAAAAAAA kpass
  Read data: AAAAAAAA
  Read SUCCESSFUL
Unlock pass kpass
  UnLock result: 1
  UnLock SUCCESSFUL
Unlock pass epc
  UnLock result: 1
  UnLock SUCCESSFUL
# The following line is for rewritable TID
# TTT Unlock pass tid
# The following line is for Permalocked TID
Unlock fail tid
  UnLock result: 0
  UnLock SUCCESSFUL
# UUU Unlock pass user
WrRd pass 3400111122223333444455556666 epc
  WrRd Write result: 1
  WrRd read result: 3400111122223333444455556666
  WrRd SUCCESSFUL
# EPC = 3400111122223333444455556666
# TTT WrRd pass E280110C tid
# Assumes a 48-bit programmable TID memory
# TID = E20010501111
# UUU WrRd pass 0000111122223333 user
# Assumes a 64-bit programmable USER memory
# USER = 0000111122223333

### ==== PermaUnlock Access Password Section ==== ###
Unlock pass apass
  UnLock result: 1
  UnLock SUCCESSFUL
PermLock pass apass
PermLock result: 1
PermLock SUCCESSFUL
# AP = 00000000, PU; Permanently unlocks the access password; TC = PU_AP
# Sets the PermaLock bit in the AP field to a one
# This TC permaunlocks the tag’s AP with the correct, zero AP.

Write pass 22222222 apass
Write result: 1
Write SUCCESSFUL
# AP = 22222222, PU; Verifies can write from open state; TC = W_AP PU_NZ_1
# This TC writes a non-zero AP when the AP is permaunlocked and zero, using the correct AP.

Read pass 22222222 apass 22222222
Read data: 22222222
Read SUCCESSFUL
# AP = 22222222, PU; Verifies can read from secured state; TC = R_AP PU_3
# This TC reads a non-zero AP, when the AP is permaunlocked and non-zero, using the correct AP.

Write pass 11111111 apass 22222222
Write result: 1
Write SUCCESSFUL
# AP = 11111111, PU; Verifies can write from secured state; TC = W_AP PU_NZ_2
# This TC writes a non-zero AP, when the AP is permaunlocked and non-zero, using the correct AP.

Read pass 11111111 apass
Read data: 11111111
Read SUCCESSFUL
# AP = 11111111, PU; Verifies can read from open state; TC = R_AP PU_5
# This TC reads a non-zero AP, when the AP is permaunlocked and non-zero, using no AP.

Read fail 11111111 apass FFFFFFFF
Read data:

Read SUCCESSFUL
# AP = 11111111, PU; Attempted read w incorrect AP; TC = R_AP PU_4
# This TC attempts to read a non-zero AP, when the AP is permaunlocked and non-zero, using an incorrect AP.

Read pass 11111111 apass 11111111
Read data: 11111111
Read SUCCESSFUL  
# AP = 11111111, PU; Verifies AP is all ones  
Write pass 22222222 apass  
   Write result: 1  
   Write SUCCESSFUL  
# AP = 22222222, PU; Verifies can write from open state; TC = W_AP_PU_NZ_4  
# This TC writes a non-zero AP, when the AP is permaunlocked and non-zero, using no AP.

Read pass 22222222 apass  
   Read data: 22222222  
   Read SUCCESSFUL  
# AP = 22222222, PU; Verifies can read from open state  
Write fail 11111111 apass FFFFFFFF  
   Write result: 0  
   Write SUCCESSFUL  
# AP = 22222222, PU; Attempted write with incorrect AP; TC = W_AP_PU_NZ_3  
# This TC attempts to write a non-zero AP, when the AP is permaunlocked and non-zero, using an incorrect AP.

Write pass 00000000 apass 22222222  
   Write result: 1  
   Write SUCCESSFUL  
# AP = 00000000, PU; Resets AP to all zeros; TC = W_AP_PU_Z  
# This TC writes a zero AP, when the AP is permaunlocked and non-zero, using the correct AP.

Read pass 00000000 apass 00000000  
   Read data: 00000000  
   Read SUCCESSFUL  
# AP = 00000000, PU; Verifies AP all zeros; TC = R_AP_PU_1  
# This TC reads a zero AP, when the AP is permaunlocked and zero, using the correct AP.

Read fail FFFFFFFF apass  
   Read data: 00000000  
   Read SUCCESSFUL  
# AP = 00000000, PU; Verifies failure to read despite with incorrect AP; TC = R_AP_PU_2  
# This TC attempts to read a zero AP, when the AP is permaunlocked and zero, using no AP.

PermUnLock fail apass  
   PermUnLock result: 0  
   PermUnLock SUCCESSFUL

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
AP = 00000000, PU; Tries to deselect the permalock bit to the access password; TC = PU_AP_PU_1
# This TC attempts to change the permalock bit, when the AP is permaunlocked and zero, using no AP.

Write pass 22222222 apass
  Write result: 1
  Write SUCCESSFUL
AP = 22222222, PU; Verifies can write from open state
Read pass 22222222 apass
  Read data: 22222222
  Read SUCCESSFUL
AP = 22222222, PU; Verifies can read from open state
PermUnLock fail apass
  PermUnLock result: 0
  PermUnLock SUCCESSFUL
AP = 22222222, PU; Tries to deselect the permalock bit for the access password from open state; TC = PU_AP_PU_3
# This TC attempts to change the permalock bit, when the AP is permaunlocked and non-zero, using no AP.

PermUnLock fail apass 22222222
  PermUnLock result: 0
  PermUnLock SUCCESSFUL
AP = 22222222, PU; Tries to deselect the permalock bit for the access password from secured state; TC = PU_AP_PU_2
# This TC attempts to change the permalock bit, when the AP is permaunlocked and non-zero, using the correct AP.

Lock fail apass
  Lock result: 0
  Lock SUCCESSFUL
AP = 22222222, PU; Tries to lock the access password from open state; TC = L_AP_PU_3
# This TC attempts to set the lock bit, when the AP is permaunlocked and non-zero, using no AP.

Lock fail apass 22222222
  Lock result: 0
  Lock SUCCESSFUL
AP = 22222222, PU; Tries to lock the access password using the correct access password, when the access password is non-zero; TC = L_AP_PU_2
# This TC attempts to set the lock bit, when the AP is permaunlocked and non-zero, using the correct AP.

Write pass 00000000 apass 22222222
  Write result: 1
  Write SUCCESSFUL
# AP = 00000000, PU; Sets tag AP to all zeros
Read pass 00000000 apass
  Read data: 00000000
  Read SUCCESSFUL
# Tag AP is in PermaUnlocked state with all zeros.

# Continue with the PermaUnlocked EPC, TID, and USER memory test cases
# and Single tag Inventory cases after EPC

### Now working on EPC memory ###
PermLock pass epc
  PermLock result: 1
  PermLock SUCCESSFUL
# EPC = 3400111122223333444455556666, PU; AP = 00000000, PU; Permanently unlocks the EPC bank; TC = PU_E
# Sets the PermaLock bit in the AP field to a one
# This TC puts the EPC in permaunlock, when the AP is permaunlocked and zero, using no AP.

Lock fail epc
  Lock result: 0
  Lock SUCCESSFUL
# EPC = 3400111122223333444455556666, PU; AP = 00000000, PU; TC = L_E_PU
# Verifies the epc bank is PermaUnlocked and that it can't be locked from the open state
# This TC attempts to set the EPC in permaLock, when the AP is permaunlocked and zero and the EPC is permaunlocked, using no AP.

Unlock pass epc
  UnLock result: 1
  UnLock SUCCESSFUL
# EPC = 3400111122223333444455556666, PU; AP = 00000000, PU; TC = U_E_PU
# This TC unsets the EPC lock bit (or confirms that it is unset in this case), when the AP is permaunlocked and zero and the EPC is permaunlocked.

PermUnLock fail epc
  PermUnLock result: 0
  PermUnLock SUCCESSFUL
# EPC = 3400111122223333444455556666, PU; AP = 00000000, PU; TC = PU_E_PU
# This TC attempts to unset the EPC permalock bit, when the AP is permaunlocked and
zero and the EPC is permaunlocked, using no AP.

Lock fail epc
  Lock result: 0
  Lock SUCCESSFUL
# EPC = 3400111122223333444455556666, PU; AP = 00000000, PU
# Verifies the EPC bank is still PermaUnlocked
Write pass 7777 epc,5
  Write result: 1
  Write SUCCESSFUL
# EPC = 3400111122223333777755556666, PU; AP = 00000000, PU; TC = W_E_PU_P
# This TC writes to a portion of the EPC, when the AP is permaunlocked and zero and
the EPC is permaunlocked.

Read pass 7777 epc,5,1
  Read data: 7777
  Read SUCCESSFUL
# EPC = 3400111122223333777755556666, PU; AP = 00000000, PU; TC = R_E_PU_P
# This TC reads a portion of the EPC, when the AP is permaunlocked and zero and the
EPC is permaunlocked, using no AP.

#Lock pass epc 00000000
# EPC = 3400111122223333777755556666, PL; AP = 00000000, PU; TC = L_E_PU, PL_E_PU
# Verifies the epc bank is PermaLocked and that it can be permalocked from the secure
state
# This test case is taken care of by PL_E_L_2

### Now working on TID memory ###
# This section is for generic TID
# Assumes a 48-bit programmable TID memory
# TID = E20010501111
#TTT PermLock pass tid
# TID = E20010501111, PU; AP = 00000000, PU; Permanently unlocks the EPC bank;
TC = PU_T
# Sets the PermaLock bit in the AP field to a one
#TTT Lock fail tid
# TID = E20010501111, PU; AP = 00000000, PU; TC = L_T_PU
# Verifies the epc bank is PermaUnlocked
#TTT Unlock pass tid
# TID = E20010501111, PU; AP = 00000000, PU; TC = U_T_PU
# TTT PermUnLock fail tid
# TID = E20010501111, PU; AP = 00000000, PU; TC = PU_T_PU
# TTT Lock fail tid
# TID = E20010501111, PU; AP = 00000000, PU
# Verifies the TID bank is still PermaUnlocked
# TTT Write pass 2222 tid,2
# TID = E20010502222, PU; AP = 00000000, PU; TC = W_T_PU_P
# TTT Read pass 2222 tid,2,1
# TID = E20010502222, PU; AP = 00000000, PU; TC = R_T_PU_P

# This section is for tags with PermaUnlocked TID memory
# Note: This section is not applicable for tags with Permalocked TID memory
#
# PermLock pass tid
# TID = E2001050, PL; AP = 00000000, PU; Permanently unlocks the TID bank; TC = PU_T
# Sets the PermaLock bit in the AP field to a one
# Lock pass tid
# TID = E2001050, PL; AP = 00000000, PU; TC = L_T_PU
# Verifies the tid bank is PermaUnlocked
# Unlock fail tid
# TID = E2001050, PL; AP = 00000000, PU; TC = U_T_PU
# PermUnLock fail tid
# TID = E2001050, PL; AP = 00000000, PU; TC = PU_T_PU
# Lock pass tid
# TID = E2001050, PL; AP = 00000000, PU
# Verifies the TID bank is still PermaUnlocked
# Write fail 1111 tid,1
# TID = E2001050, PL; AP = 00000000, PU; TC = W_T_PU_P
# Read pass 110C tid,1,1
# TID = E2001050, PL; AP = 00000000, PU; TC = R_T_PU_P

### Now working on USER memory ###
# Assumes a 64-bit programmable USER
# USER = 0000111222233
# Note: This section is not applicable for tags with no user memory
# UUU Unlock pass user
# UUU PermLock pass user
# USER = 0000111122223333, PU; AP = 00000000, PU; Permanently unlocks the USER bank; TC = PU_U
# Sets the PermaLock bit in the AP field to a one
# UUU Lock fail user 00000000
# USER = 0000111122223333, PU; AP = 00000000, PU; Attempt to lock the USER bank; TC = L_U_PU
#UUU Unlock pass user
# USER = 0000111122223333, PU; AP = 00000000, PU; TC = U_U_PU
#PermUnLock fail user
# USER = 0000111122223333, PU; AP = 00000000, PU; TC = PU_U_PU
# Verifies the User memory is still Permaunlocked
#UUU Write pass 4444 user,3
# USER = 0000111122224444, PU; AP = 00000000, PU; TC = W_U_PU_P
#UUU Read pass 4444 user,3,1
# USER = 0000111122224444, PU; AP = 00000000, PU; TC = R_U_PU_P

### Now working on Kill password ###
PermLock pass kpass
   PermLock result: 1
   PermLock SUCCESSFUL
# KP = AAAAAAAA, PU; AP = 00000000, PU; Permanently unlocks the EPC bank;
   TC = PU_KP
# This TC puts the KP in permaunlock, when the AP is zero, using no AP.

# Sets the PermaLock bit in the AP field to a one
Lock fail kpass
   Lock result: 0
   Lock SUCCESSFUL
# KP = AAAAAAAA, PU; AP = 00000000, PU; TC = L_KP_PU
# Verifies the kill password is PermaUnlocked and that it can’t lock it from the open
   state.
# This TC attempts to set the lock bit for the KP, using no AP, when the KP is
   permaunlocked.

Unlock pass kpass
   UnLock result: 1
   UnLock SUCCESSFUL
# KP = AAAAAAAA, PU; AP = 00000000, PU; TC = U_KP_PU
# This TC unsets the lock bit to the KP (or confirms that it is already unset) when the
   KP is permaunlocked.

PermUnLock fail kpass
   PermUnLock result: 0
   PermUnLock SUCCESSFUL
# KP = AAAAAAAA, PU; AP = 00000000, PU; TC = PU_KP_PU
# This TC attempts to unset the permalock bit, while the KP is permaunlocked.

Lock fail kpass
   Lock result: 0
   Lock SUCCESSFUL
# KP = AAAAAAAA, PU; AP = 00000000, PU
# Verifies the Kill password is still PermaUnlocked
Write pass BBBBBBBB kpass
   Write result: 1
   Write SUCCESSFUL
# KP = BBBBBBBB, PU; AP = 00000000, PU; TC = W_KP_PU_NZ
# This TC writes a non-zero KP, when the AP is zero and the KP is permaunlocked and non-zero.

Read pass BBBBBBBB kpass
   Read data: BBBBBBBB
   Read SUCCESSFUL
# KP = BBBBBBBB, PU; AP = 00000000, PU; TC = R_KP_PU
# This TC reads a non-zero KP, when the AP is zero and the KP is permaunlocked and non-zero.

Kill fail ABBAABBA
   Kill result: 0
   Kill SUCCESSFUL
# KP = AAAAAAAA, PU; AP = 00000000, PU; TC = K_INZ_PU
# This TC attempts to kill the tag using an incorrect, non-zero KP, while the KP is permaunlocked.

### Now run the Inventory cases ###

Inventory pass 1
   Inventoried 1 unique tags
      111122223333777755556666 :1
   Inventory SUCCESSFUL
# TC = I_PU
# This TC does an inventory of a single tag.

Inventory pass 1 11112222 epc
   Inventoried 1 unique tags
      111122223333777755556666 :1
   Inventory SUCCESSFUL
# EPC = 34001112222333777755556666, PU; AP = 00000000; TC = SI_E_PU
Read pass 3400111122223333777755556666 epc
   Read data: 3400111122223333777755556666
   Read SUCCESSFUL
# EPC = 3400111122223333777755556666, PU; AP = 00000000; Verifies EPC
# This TC inventories a single tag, based on a portion of the EPC being present.

Inventory pass 1 E280 tid
   Inventoried 1 unique tags
      111122223333777755556666 :1
Inventory SUCCESSFUL
# TID = E2001050, PU/PL; AP = 00000000; TC = SI_T_PU
Read pass E280B0A1 tid
  Read data: E280B0A1
  Read SUCCESSFUL
# TID = E2001050, PU/PL; AP = 00000000; Verifies TID
# This TC inventories a single tag, based on a portion of the TID being present.

#UUU Inventory pass 1 00001111 user
# USER = 0000111122224444, PU; AP = 00000000; TC = SI_U_PU
#UUU Read pass 0000111122224444 user,0,4
# USER = 0000111122224444, PU; AP = 00000000; Verifies user memory
# This TC inventories a single tag, based on a portion of the User data being present.

disconnect
  Disconnected.

Script completed without failures

**PermaLocked_L**

# Created by Impinj and modified by MET Laboratories for use in EPCglobal sanctioned Gen2 RFID interoperability testing
# Permission granted for free use only for EPCglobal sponsored interoperability testing activities
#
# EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable for any
# injury, loss, damages, financial or otherwise, arising from, related to, or caused by the use of this
# document. Use of said document does not guarantee compliance with applicable state and/or federal laws.
# User is responsible for the interpretation of and compliance with applicable pedigree laws. The use of
# said document shall constitute your express consent to the foregoing disclaimer.

connect 169.254.10.1 FCC 201
  Connected to reader 169.254.10.1, type: FCC, mode: 201
power 20
  Power set to 20

# "unlocked" means that the bit pattern to the associated memory bank is "00"
# "locked" that the bit pattern to the associated memory bank is "10"
# "permaunlocked" means that the bit pattern to the associated memory bank is "01"
# "permalocked" means that the bit pattern to the associated memory bank is "11"

Write pass 00000000 apass
  Write result: 1
  Write SUCCESSFUL
Read pass 00000000 apass
  Read data: 00000000
  Read SUCCESSFUL
Write pass AAAAAAAA kpass
  Write result: 1
  Write SUCCESSFUL
Read pass AAAAAAAA kpass
  Read data: AAAAAAAA
  Read SUCCESSFUL
Unlock pass kpass
  UnLock result: 1
  UnLock SUCCESSFUL
Unlock pass epc
  UnLock result: 1
  UnLock SUCCESSFUL

# The following line is for rewritable TID
# TTT Unlock pass tid
# The following line is for Permalocked TID
Unlock fail tid
  UnLock result: 0
  UnLock SUCCESSFUL

#UUU Unlock pass user
WrRd pass 3400111122223333444455556666 epc
  WrRd Write result: 1
  WrRd read result: 3400111122223333444455556666
  WrRd SUCCESSFUL
# EPC = 3400111122223333444455556666
# TTT WrRd pass E280110C tid
# Assumes a 48-bit programmable TID memory
# TID = E20010501111
#UUU WrRd pass 0000111122223333 user
# Assumes a 64-bit programmable USER memory
# USER = 0000111122223333

### Now working on the rest of the AP memory ###
Write pass 12345678 apass
  Write result: 1
  Write SUCCESSFUL
Read pass 12345678 apass 12345678

Report No.: EMC94348B Rev. 2

Date: 2017-12-08

Page 55 of 114
Read data: 12345678
Read SUCCESSFUL
# AP=12345678, U, KP=AAAAAAAA
lock pass apass 12345678
  Lock result: 1
  Lock SUCCESSFUL
# Putting tag in locked state for PL test
Permlock fail apass
  PermLock result: 0
  PermLock SUCCESSFUL
# Attempting to put in Permalock while in open state. TC=PL_AP_L_2
Write fail 33333333 apass
  Write result: 0
  Write SUCCESSFUL
# AP=12345678, L; Verifies can't write from open state
Read fail 33333333 apass
  Read data:

  Read SUCCESSFUL
# AP = 12345678, L; Verifies can't read 33333333 from open state
Read fail 12345678 apass
  Read data:

  Read SUCCESSFUL
# AP = 12345678, L; Verifies can't read 12345678 from open state
Read pass 12345678 apass 12345678
  Read data: 12345678
  Read SUCCESSFUL
# AP = 12345678, L; Verifies can write from secured state;
Write pass 11111111 apass 12345678
  Write result: 1
  Write SUCCESSFUL
# AP = 12345678, L; Verifies can write from secured state;
Read pass 11111111 apass 11111111
  Read data: 11111111
  Read SUCCESSFUL
# AP = 11111111, L; Verifies AP is 12345678 from the secured state;
# This TC attempts to set the permalock bit for the AP, while the tag's AP is locked and non-zero, using no AP
PermLock pass apass 11111111
  PermLock result: 1
  PermLock SUCCESSFUL
# Tag now in PL state with non-zero password. TC=PL_AP_L_1

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
Write fail 33333333 apass 11111111
  Write result: 0
  Write SUCCESSFUL
# AP=11111111, PL; Verifies can't write from secured state
Read fail 33333333 apass 33333333
  Read data:

    Read SUCCESSFUL
# AP = 11111111, PL; Verifies AP is not 33333333
Read fail 11111111 apass
  Read data:

    Read SUCCESSFUL
# AP = 11111111, PL; Verifies can't read from open state.
Read fail 11111111 apass 11111111
  Read data:

    Read SUCCESSFUL
# AP = 11111111, PL; Verifies can read from secured state
Lock pass apass 11111111
  Lock result: 1
  Lock SUCCESSFUL
# AP = 11111111, PL; Verifies still can lock the tag
Write fail 00000000 apass 11111111
  Write result: 0
  Write SUCCESSFUL
Read fail 00000000 apass 00000000
  Read data:

    Read SUCCESSFUL
# AP = 11111111, PL; Verifies still can't write to the AP
# This TC sets the permalock bit for the AP and puts the tag in permalock, while the
tag's AP is locked and non-zero, using the correct AP.

disconnect
  Disconnected.

Script completed without failures

**Permalocked_APZ**

# Created by Impinj and modified by MET Laboratories for use in EPCglobal sanctioned Gen2 RFID interoperability testing
# This script verifies PermaLock behavior with zero AP
#
# "unlocked" means that the bit pattern to the associated memory bank is "00"
# "locked" that the bit pattern to the associated memory bank is "10"
# "permaunlocked" means that the bit pattern to the associated memory bank is "01"
# "permalocked" means that the bit pattern to the associated memory bank is "11"

Write pass 00000000 apass
  Write result: 1
  Write SUCCESSFUL
Read pass 00000000 apass
  Read data: 00000000
  Read SUCCESSFUL
Write pass AAAAAAAAA kpass
  Write result: 1
  Write SUCCESSFUL
Read pass AAAAAAAAA kpass
  Read data: AAAAAAAAA
  Read SUCCESSFUL
Unlock pass kpass
  UnLock result: 1
  UnLock SUCCESSFUL
Unlock pass epc
  UnLock result: 1
  UnLock SUCCESSFUL
# The following line is for rewritable TID
#TTT Unlock pass tid
# The following line is for Permalocked TID
Unlock fail tid
    UnLock result: 0
    UnLock SUCCESSFUL
#UUU Unlock pass user
WrRd pass 3400111122223333444455556666 epc
    WrRd Write result: 1
    WrRd read result: 3400111122223333444455556666
    WrRd SUCCESSFUL
# EPC = 3200111122223333444455556666
#TTT WrRd pass E280110C tid
# Assumes a 48-bit programmable TID memory
# TID = E2801050111
#UUU WrRd pass 0000111122223333 user
# Assumes a 64-bit programmable USER memory
# USER = 0000111122223333

### ==== PermaLock Access Password Section ==== ###
Lock pass apass
    Lock result: 1
    Lock SUCCESSFUL
# AP = 00000000, L; Locks the access password
# Sets the Lock bit in the AP field to a one
# This action is required first in order to Permalock using PermLock directive
PermLock pass apass 00000000
    PermLock result: 1
    PermLock SUCCESSFUL
# AP = 00000000, PL; Permanently locks the access password; TC = PL_AP_L_3
# Sets the PermaLock bit in the AP field to a one
Write fail FFFFFFFF apass
    Write result: 0
    Write SUCCESSFUL
# AP = 00000000, PL; Verifies memory cannot write from open state
Read fail FFFFFFFF apass
    Read data: Read SUCCESSFUL
# AP = 00000000, PL; Verifies memory cannot read from open state
Write fail 00000000 apass 00000000
    Write result: 0
    Write SUCCESSFUL
# AP = 00000000, PL; Verifies memory cannot write from secured state
Read fail 00000000 apass 00000000
    Read data:
Read SUCCESSFUL
# AP = 00000000, PL; Verifies memory cannot read from secured state
# This TC permalocks the AP when the AP is zero and correct. It's also the TC where it permalocks the AP with it already being locked.

Unlock fail apass 00000000
   UnLock result: 0
   UnLock SUCCESSFUL
# AP = 00000000, PL; Verifies memory cannot be unlocked from secured state; TC = U_AP_PL_1
# This TC attempts to unlock the AP after it has been permalocked using the correct, zero AP.

PermUnLock fail apass 00000000
   PermUnLock result: 0
   PermUnLock SUCCESSFUL
# AP = 00000000, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_AP_PL_1
# This TC attempts to deset the permalock bit to the AP after it has been permalocked using the correct, zero AP.

Lock pass apass 00000000
   Lock result: 1
   Lock SUCCESSFUL
# AP = 00000000, PL; Verifies memory can be Permaunlocked and locked from secured state; TC = L_AP_PL_1
# This TC confirms that the lock bit is set to the AP after it has been permalocked using the correct, zero AP.

### === Now working on EPC memory === ###
Lock pass epc
   Lock result: 1
   Lock SUCCESSFUL
# AP = 00000000, L; Locks the EPC memory
# Sets the Lock bit in the EPC field to a one
# This action is required first in order to Permalock using PermLock directive
PermLock pass epc 00000000
   PermLock result: 1
   PermLock SUCCESSFUL
# EPC = 3400111122223333444455556666, PL; AP = 00000000, PL; Permanently locks the EPC bank from secured state; TC = PL_E_L_1
# This TC permalocks the EPC, after it's locked, when the AP is correct and zero.

# Sets the PermaLock bit in the AP field to a one
Write fail 7777 epc,5
Write result: 0
Write SUCCESSFUL
# EPC = 34001112222333444455556666, PL; AP = 00000000, PL; Verifies memory cannot write from open state; TC = W_E_PL_P_1
# This TC attempts to write a portion of the EPC, when the AP is zero and permalocked.

Read pass 4444 epc,5,1
Read data: 4444
Read SUCCESSFUL
# EPC = 34001112222333444455556666, PL; AP = 00000000, PL; Verifies memory cannot read from open state; TC = R_E_PL_P_1
# This TC attempts to read a portion of the EPC, when the AP is zero and permalocked.

Write fail 7777 epc,5 00000000
Write result: 0
Write SUCCESSFUL
# EPC = 34001112222333444455556666, PL; AP = 00000000, PL; Verifies memory cannot write from secured state
Read pass 4444 epc,5,1 00000000
Read data: 4444
Read SUCCESSFUL
# EPC = 34001112222333444455556666, PL; AP = 00000000, PL; Verifies memory cannot read from secured state
Unlock fail epc 00000000
Unlock result: 0
Unlock SUCCESSFUL
# EPC = 34001112222333444455556666, PL; AP = 00000000, PL; Verifies memory cannot be unlocked from secured state; TC = U_E_PL_1
# This TC attempts to unlock the EPC, when the AP is zero and permalocked.

PermUnLock fail epc 00000000
PermUnLock result: 0
PermUnLock SUCCESSFUL
# EPC = 34001112222333444455556666, PL; AP = 00000000, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_E_PL_1
# This TC attempts to unset the permalock bit to the EPC, when the AP is zero and permalocked.

Lock pass epc 00000000
Lock result: 1
Lock SUCCESSFUL
# EPC = 34001112222333444455556666, PL; AP = 00000000, PL; Verifies memory can be locked from secured state; TC = L_E_PL_1
# This TC attempts to set the lock bit to the EPC, (or confirm that it's set) when the AP is zero and permalocked.

### Now working on TID memory ###

# This section is for generic TID
# Assumes a 48-bit programmable TID memory
# TID = E20010501111
# TTT Lock pass tid
# TID = E20010501111, L; AP = 00000000, PL; Locks the TID memory
# Sets the Lock bit in the TID field to a one
# This action is required first in order to Permalock using PermLock directive
# TTT PermLock pass tid 00000000
# TID = E20010501111, PL; AP = 00000000, PL; Permanently locks the TID bank from secured state; TC = PL_T_L_1
# Sets the PermaLock bit in the AP field to a one
# TTT Write fail 2222 tid,2
# TID = E20010501111, PL; AP = 00000000, PL; Verifies memory cannot write from open state; TC = W_T_PL_P_1
# TTT Read pass 1111 tid,2,1
# TID = E20010501111, PL; AP = 00000000, PL; Verifies memory cannot read from open state; TC = R_T_PL_P_1
# TTT Write fail 2222 tid,2 00000000
# TID = E20010501111, PL; AP = 00000000, PL; Verifies memory cannot write from secured state
# TTT Read pass 1111 tid,2,1 00000000
# TID = E20010501111, PL; AP = 00000000, PL; Verifies memory cannot read from secured state
# TTT Unlock fail tid 00000000
# TID = E20010501111, PL; AP = 00000000, PL; Verifies memory cannot be unlocked from secured state; TC = U_T_PL_1
# TTT PermUnLock fail tid 00000000
# TID = E20010501111, PL; AP = 00000000, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_T_PL_1
# TTT Lock pass tid 00000000
# TID = E20010501111, PL; AP = 00000000, PL; Verifies memory can be locked from secured state; TC = L_T_PL_1

# This section is for PermaLocked TID
# TID = E2001050
Lock pass tid
  Lock result: 1
  Lock SUCCESSFUL
# TID = E2001050, L; AP = 00000000, PL; Locks the TID memory
# Sets the Lock bit in the TID field to a one
# This action is required first in order to Permalock using PermLock directive
PermLock pass tid 00000000
PermLock result: 1
PermLock SUCCESSFUL
# TID = E2001050, PL; AP = 00000000, PL; Permanently locks the TID bank from
secured state; TC = PL_T_L_1
# This TC permlocks the TID (or confirms it's permlocked) when the tag's TID is
locked and the AP is zero.

# Sets the PermaLock bit in the TID field to a one
Write fail 2222 tid,1
Write result: 0
Write SUCCESSFUL
# TID = E2001050, PL; AP = 00000000, PL; Verifies memory cannot write from open
state; TC = W_T_PL_P_1
# This TC attempts to write to the TID when the TID is permalocked, using an all-zero
AP.

Read pass B0A1 tid,1,1
Read data: B0A1
Read SUCCESSFUL
# TID = E2001050, PL; AP = 00000000, PL; Verifies memory can read from open
state; TC = R_T_PL_P_1
# This TC reads a known value from the TID when the TID is permalocked, using an
all-zero AP.

Write fail 2222 tid,1 00000000
Write result: 0
Write SUCCESSFUL
# TID = E2001050, PL; AP = 00000000, PL; Verifies memory cannot write from
secured state
Read pass B0A1 tid,1,1 00000000
Read data: B0A1
Read SUCCESSFUL
# TID = E2001050, PL; AP = 00000000, PL; Verifies memory cannot read from
secured state
Unlock fail tid 00000000
Unlock result: 0
Unlock SUCCESSFUL
# TID = E2001050, PL; AP = 00000000, PL; Verifies memory cannot be unlocked from
secured state; TC = U_T_PL_1
# This TC attempts to unlock the TID from the permalocked state when the AP is zero.

PermUnLock fail tid 00000000
PermUnLock result: 0
PermUnLock SUCCESSFUL
Lock pass tid 00000000
Lock result: 1
Lock SUCCESSFUL

# This TC attempts to unset the TID’s permalock bit when the AP is zero.

### Now working on USER memory ###
# Assumes a 64-bit programmable USER
# USER = 0000111122223333
#UUU Lock pass user
# USER = 0000111122223333, L; Locks the USER memory
# Sets the Lock bit in the USER field to a one
# This action is required first in order to Permalock using PermLock directive
#UUU PermLock pass user 00000000
# USER = 0000111122223333, PL; AP = 00000000, PL; Permanently locks the USER bank from secured state; TC = PL_U_L_1
# Sets the PermaLock bit in the USER field to a one
#UUU Write fail 4444 user,3
# USER = 0000111122223333, PL; AP = 00000000, PL; Verifies memory cannot write from open state; TC = W_U_PL_P_1
#UUU Read pass 3333 user,3,1
# USER = 0000111122223333, PL; AP = 00000000, PL; Verifies memory can read from open state; TC = R_U_PL_P_1
#UUU Write fail 4444 user,3 00000000
# USER = 0000111122223333, PL; AP = 00000000, PL; Verifies memory cannot write from secured state
#UUU Read pass 3333 user,3,1 00000000
# USER = 0000111122223333, PL; AP = 00000000, PL; Verifies memory cannot read from secured state
#UUU Unlock fail user 00000000
# USER = 0000111122223333, PL; AP = 00000000, PL; Verifies memory cannot be unlocked from secured state; TC = U_U_PL_1
#UUU PermUnLock fail user 00000000
# USER = 0000111122223333, PL; AP = 00000000, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_U_PL_1
#UUU Lock pass user 00000000
# USER = 0000111122223333, PL; AP = 00000000, PL; Verifies memory can be locked from secured state; TC = L_U_PL_1
### Now working on Kill password ###
Lock pass kpass
   Lock result: 1
   Lock SUCCESSFUL
# KP = AAAAAAAA, L; Locks the Kill password
# Sets the Lock bit in the KP field to a one
# This action is required first in order to Permalock using PermLock directive
PermLock pass kpass 00000000
   PermLock result: 1
   PermLock SUCCESSFUL
# KP = AAAAAAAA, PL; Permanently locks the kill password from secured state; TC = PL_KP_L_1, PL_KP
# Sets the PermaLock bit in the KP field to a one
# This TC permalocks the KP, after it was locked, using a zero AP.

Write fail FFFFFFFF kpass
   Write result: 0
   Write SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot write from open state; TC = W_KP_PL_NZ_1
# This TC writes a non-zero KP, when the KP is permalocked, using a zero AP.

Read fail AAAAAAAA kpass
   Read data:
   Read SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot read from open state; TC = R_KP_PL_1
# This TC attempts to read the KP, when the KP is permalocked, using a zero AP.

Write fail FFFFFFFF kpass 00000000
   Write result: 0
   Write SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot write from secured state
Read fail AAAAAAAA kpass 00000000
   Read data:
   Read SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot read from secured state

Unlock fail kpass 00000000
   UnLock result: 0
   UnLock SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot be unlocked from secured state; TC = U_KP_PL_1
PermUnLock fail kpass 00000000
PermUnLock result: 0
PermUnLock SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_KP_PL_1
# This TC attempts to deset the permaunlock bit, when the KP is permalocked, using a zero AP.

Lock pass kpass 00000000
Lock result: 1
Lock SUCCESSFUL
# KP = AAAAAAAA, PL; AP = 00000000, PL; Verifies memory can be locked from secured state; TC = L_KP_PL_1
# This TC sets the lock bit to the KP, (or confirms that it's set) when the KP is permalocked and the AP is zero.

Kill fail ABBAABBA
Kill result: 0
Kill SUCCESSFUL
# KP = AAAAAAAA, PL; AP = 00000000, PL; TC = K_INZ_PL
# This TC attempts to kill the tag, while the KP is permalocked, using an incorrect, non-zero KP.

### Now run the Inventory cases ###

Inventory pass 1
Inventoried 1 unique tags
111122223333444455556666 :1
Inventory SUCCESSFUL
# TC = I_PL
# This TC inventories the single tag while all memory is permalocked.

Inventory pass 1 11112222 epc
Inventoried 1 unique tags
111122223333444455556666 :1
Inventory SUCCESSFUL
# EPC = 3200111122223333444455556666, PL; AP = 00000000; TC = SI_E_PL
Read pass 3400111122223333444455556666 epc
Read data: 3400111122223333444455556666
Read SUCCESSFUL
# EPC = 3200111122223333444455556666, PL; AP = 00000000; Verifies EPC
# This TC inventories the tag based on a portion of the EPC while the tag's memory is permalocked.

Inventory pass 1 E280 tid
Inventoried 1 unique tags
111122223333444455556666 :1
Inventory SUCCESSFUL
# TID = E2001050, PU/PL; AP = 00000000; TC = SI_T_PL
Read pass E280B0A1 tid
  Read data: E280B0A1
  Read SUCCESSFUL
# TID = E2001050, PU/PL; AP = 00000000; Verifies TID
# This TC inventories the tag based on a portion of the TID while the tag's memory is permalocked.

#UUU Inventory pass 1 00001111 user
# USER = 0000111122223333, PL; AP = 00000000; TC = SI_U_PL
#UUU Read pass 0000111122223333 user
# USER = 0000111122223333, PL; AP = 00000000; Verifies user memory

Kill pass AAAAAAAA
  Kill result: 1
  Kill SUCCESSFUL
# KP = AAAAAAAA, PL; AP = 00000000, PL; TC = K_NZ
# Dead tag
Inventory fail 1 11112222 epc
  Inventoried 0 unique tags
  Inventory SUCCESSFUL
# EPC = 3400111122223333444455556666, PL; AP = 00000000
Inventory fail 1 E280 tid
  Inventoried 0 unique tags
  Inventory SUCCESSFUL
# TID = E2001050, PU/PL; AP = 00000000
Inventory fail 1 00001111 user
  Inventoried 0 unique tags
  Inventory SUCCESSFUL
# USER = 0000111122223333, PL; AP = 00000000
# This TC kills the tag using a non-zero KP.

disconnect
  Disconnected.
Script completed without failures

**PermaLocked_APNZ**

# Created by Impinj, with comments by MET Laboratories, for use in EPCglobal sanctioned Gen2 RFID interoperability testing
# Permission granted for free use only for EPCglobal sponsored interoperability testing activities
# This script verifies PermaLock behavior with nonzero AP
#
# "unlocked" means that the bit pattern to the associated memory bank is "00"
# "locked" that the bit pattern to the associated memory bank is "10"
# "permaunlocked" means that the bit pattern to the associated memory bank is "01"
# "permalocked" means that the bit pattern to the associated memory bank is "11"
#
Write pass ABCDEF01 apass
   Write result: 1
   Write SUCCESSFUL
Read pass ABCDEF01 apass
   Read data: ABCDEF01
   Read SUCCESSFUL
Write pass AAAAAAAA kpass ABCDEF01
   Write result: 1
   Write SUCCESSFUL
Read pass AAAAAAAA kpass ABCDEF01
   Read data: AAAAAAAA
   Read SUCCESSFUL
Unlock pass kpass ABCDEF01
   UnLock result: 1
   UnLock SUCCESSFUL
Unlock pass epc ABCDEF01
   UnLock result: 1
   UnLock SUCCESSFUL
# The following line is for rewritable TID
#TTT Unlock pass tid ABCDEF01
# The following line is for Permalocked TID
Unlock fail tid ABCDEF01
    UnLock result: 0
    UnLock SUCCESSFUL
# The following line is for User memory
    #UUU Unlock pass user
    WrRd pass 3400AAAAABBBCCDDEEEEFFFF epc ABCDEF01
        WrRd Write result: 1
        WrRd read result: 3400AAAAABBBCCDDEEEEFFFF
        WrRd SUCCESSFUL
# EPC = 3400AAAAABBBCCDDEEEEFFFF
# TTT WrRd pass E2C0B0A1 tid ABCDEF01
# Assumes a 48-bit programmable TID memory
# TID = E20010501111
# UUU WrRd pass FFFFEFDDDDCC user ABCDEF01
# Assumes a 64-bit programmable USER memory
# USER = FFFFEFDDDDCC
### ==== PermaLock Access Password Section ==== ###
Lock pass apass ABCDEF01
    Lock result: 1
    Lock SUCCESSFUL
# AP = ABCDEF01, L; Locks the access password
# Sets the Lock bit in the AP field to a one
# This action is required first in order to Permalock using PermLock directive
PermLock pass apass ABCDEF01
    PermLock result: 1
    PermLock SUCCESSFUL
# AP = ABCDEF01, PL; Permanently locks the access password; same as TC = L_AP_PU_2
# Sets the PermaLock bit in the AP field to a one
# This TC puts the AP in permalocked state, which is what happens when the AP is
# locked from the PU state, using a non-zero and correct AP.
Write fail FFFFFFFF apass
    Write result: 0
    Write SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot write from open state
Read fail ABCDEF01 apass
    Read data:
    Read SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot read from open state
Write fail FFFFEFDDDDCC apass ABCDEF01
    Write result: 0
Write SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot write from secured state; TC = W_AP_PL_NZ_1
# This TC writes a non-zero AP, when the AP is permalocked and non-zero, using the correct AP.

Read fail ABCDEF01 apass ABCDEF01
Read data:

    Read SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot read from secured state; TC = R_AP_PL_1
# This TC attempts to read the AP, when the AP is permalocked and non-zero, using the correct AP.

Write fail FFFFFFFF apass
    Write result: 0
Write SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot write from open state; TC = W_AP_PL_NZ_2
# This TC attempts to write a non-zero AP, when the AP is permalocked and non-zero, using no AP.

Read fail ABCDEF01 apass
Read data:

    Read SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot read from open state; TC = R_AP_PL_2
# This TC attempts to read the AP, when the AP is permalocked and non-zero, using no AP.

Unlock fail apass ABCDEF01
    UnLock result: 0
Unlock SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot be unlocked from secured state; TC = U_AP_PL_2
# This TC attempts to deset the lock bit to the AP, when the AP is permalocked and non-zero, using the correct AP.

Unlock fail apass
    UnLock result: 0
Unlock SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot be unlocked from open state; TC = U_AP_PL_3
# This TC attempts to unlock the AP, when the AP is permalocked and non-zero, using no AP.

PermUnLock fail apass ABCDEF01
   PermUnLock result: 0
   PermUnLock SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_AP_PL_2
# This TC attempts to deset the peramlock bit to the AP, when the AP is permalocked and non-zero, using the correct AP.

PermUnLock fail apass
   PermUnLock result: 0
   PermUnLock SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot be Permaunlocked from open state; TC = PU_AP_PL_3
# This TC attempts to deset the peramlock bit to the AP, when the AP is permalocked and non-zero, using no AP.

Lock pass apass ABCDEF01
   Lock result: 1
   Lock SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot be Permaunlocked from secured state; TC = L_AP_PL_2
# This TC locks the AP (or confirms that the lock bit is set), when the AP is permalocked and non-zero, using the correct AP.

Lock fail apass
   Lock result: 0
   Lock SUCCESSFUL
# AP = ABCDEF01, PL; Verifies memory cannot be Permaunlocked from open state; TC = L_AP_PL_3
# This TC attempts to lock the AP (or confirm that the lock bit is set), when the AP is permalocked and non-zero, using no AP.

### === Now working on EPC memory === ###

Lock pass epc ABCDEF01
   Lock result: 1
   Lock SUCCESSFUL
# AP = ABCDEF01, L; Locks the EPC memory
# Sets the Lock bit in the EPC field to a one
# This action is required first in order to Permalock using PermLock directive
PermLock pass epc ABCDEF01
   PermLock result: 1
PermLock SUCCESSFUL
# EPC = 3400AAAABBBBCCCCDDDDDEEEEEFFFF, PL; AP = ABCDEF01, PL;
Permanently locks the EPC bank from secured state; TC = PL_E_L_2
# Sets the PermaLock bit in the AP field to a one
# This TC permalocks the EPC, when the EPC is locked and the AP is non-zero.

Write fail 7777 epc,5
  Write result: 0
  Write SUCCESSFUL
# EPC = 3400AAAABBBBCCCCDDDDDEEEEEFFFF, PL; AP = ABCDEF01, PL;
Verifies memory cannot write from open state; TC = W_E_PL_P_2
# This TC attempts to write to a portion of the EPC, when the AP is non-zero and the EPC is permalocked. See following command for write from secured state.

Read pass DDDD epc,5,1
  Read data: DDDD
  Read SUCCESSFUL
# EPC = 3400AAAABBBBCCCCDDDDDEEEEEFFFF, PL; AP = ABCDEF01, PL;
Verifies memory can be read from open state; TC = R_E_PL_P_2
# This TC reads a portion of the EPC, when the AP is non-zero and the EPC is permalocked. See following command for read from secured state.

Write fail 7777 epc,5 ABCDEF01
  Write result: 0
  Write SUCCESSFUL
# EPC = 3400AAAABBBBCCCCDDDDDEEEEEFFFF, PL; AP = ABCDEF01, PL;
Verifies memory cannot write from secured state

Read pass DDDD epc,5,1 ABCDEF01
  Read data: DDDD
  Read SUCCESSFUL
# EPC = 3400AAAABBBBCCCCDDDDDEEEEEFFFF, PL; AP = ABCDEF01, PL;
Verifies memory can be read from secured state

Unlock fail epc ABCDEF01
  UnLock result: 0
  UnLock SUCCESSFUL
# EPC = 3400AAAABBBBCCCCDDDDDEEEEEFFFF, PL; AP = ABCDEF01, PL;
Verifies memory cannot be unlocked from secured state; TC = U_E_PL_2
# This TC attempts to deset the lock bit, when the AP is non-zero and the EPC is permalocked.

PermUnLock fail epc ABCDEF01
  PermUnLock result: 0
  PermUnLock SUCCESSFUL
# EPC = 3400AAAABBBBCCCDDDEEEFF, PL; AP = ABCDEF01, PL;
Verifies memory cannot be Permaunlocked from secured state; TC = PU_E_PL_2
# This TC attempts to deset the permalock bit, when the AP is non-zero and the EPC is
permalocked.

Lock fail epc 12345678
    Lock result: 0
    Lock SUCCESSFUL
# EPC = 3400AAAABBBBCCCDDDEEEFF, PL; AP = ABCDEF01, PL;
Verifies memory cannot be locked from secured state with incorrect AP

Lock fail epc
    Lock result: 0
    Lock SUCCESSFUL
# EPC = 3400AAAABBBBCCCDDDEEEFF, PL; AP = ABCDEF01, PL;
Verifies memory cannot be locked from open state
Lock pass epc ABCDEF01
    Lock result: 1
    Lock SUCCESSFUL
# EPC = 3400AAAABBBBCCCDDDEEEFF, PL; AP = ABCDEF01, PL;
Verifies memory can be locked from secured state; TC = L_E_PL_2
# This TC attempts to set the lock bit, when the AP is non-zero and the EPC is
permalocked, using an incorrect AP, no AP, and the correct AP.

### Now working on TID memory ###
# This section is for generic TID
# Assumes a 48-bit programmable TID memory
# TID = E20010501111
#TTT Lock pass tid ABCDEF01
# TID = E20010501111, L; AP = ABCDEF01, PL; Locks the TID memory
# Sets the Lock bit in the TID field to a one
# This action is required first in order to Permalock using PermLock directive
#TTT PermLock pass tid ABCDEF01
# TID = E20010501111, PL; AP = ABCDEF01, PL; Permanently locks the TID bank
from secured state; TC = PL_T_L_2
# Sets the PermaLock bit in the AP field to a one
#TTT Write fail 2222 tid,2
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory cannot write
from open state; TC = W_T_PL_P_2
#TTT Read pass 1111 tid,2
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory can be read from
open state; TC = R_T_PL_P_2
#TTT Write fail 2222 tid,2 ABCDEF01
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory cannot write from secured state
# TTT Read pass 1111 tid,2,1 ABCDEF01
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory can be read from secured state
# TTT Unlock fail tid ABCDEF01
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory cannot be unlocked from secured state; TC = U_T_PL_2
# TTT PermUnLock fail tid ABCDEF01
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory cannot be Permanently locked from secured state; TC = PU_T_PL_2
# TTT Lock fail tid 12345678
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory cannot be locked from secured state with incorrect AP
# TTT Lock fail tid
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory cannot be locked from open state
# TTT Lock pass tid ABCDEF01
# TID = E20010501111, PL; AP = ABCDEF01, PL; Verifies memory can be locked from secured state; TC = L_T_PL_2

# This section is for PermaLocked TID
# TID = E2001050
Lock pass tid ABCDEF01
  Lock result: 1
  Lock SUCCESSFUL
# TID = E2001050, L; AP = ABCDEF01, PL; Locks the TID memory
# Sets the Lock bit in the TID field to a one
# This action is required first in order to Permalock using PermLock directive
PermLock pass tid ABCDEF01
  PermLock result: 1
  PermLock SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Permanently locks the TID bank from secured state; TC = PL_T_L_2
# This TC makes the TID permalocked (or confirms that it's permalocked) when the AP is non-zero and the TID is locked.

# Sets the PermaLock bit in the TID field to a one
Write fail 2222 tid,1
  Write result: 0
  Write SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory cannot write from open state; TC = W_T_PL_P_2
# This TC (and the subsequent command for the secured state) attempt to write to a portion of the TID, when the TID is permalocked and the AP is non-zero.
Read pass B0A1 tid,1,1
   Read data: B0A1
   Read SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory can be read from open state; TC = R_T_PL_P_2
# This TC (and the subsequent command for the secured state) attempt to read from a portion of the TID, when the TID is permalocked and the AP is non-zero.

Write fail 2222 tid,1 ABCDEF01
   Write result: 0
   Write SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory cannot write from secured state
Read pass B0A1 tid,1,1 ABCDEF01
   Read data: B0A1
   Read SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory can be read from secured state

Unlock fail tid ABCDEF01
   UnLock result: 0
   UnLock SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory cannot be unlocked from secured state; TC = U_T_PL_2
# This TC attempts to deset the lock bit to the TID, when the TID is permalocked and the AP is non-zero.

PermUnLock fail tid ABCDEF01
   PermUnLock result: 0
   PermUnLock SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_T_PL_2
# This TC attempts to deset the permalock bit to the TID, when the TID is permalocked and the AP is non-zero.

Lock fail tid 12345678
   Lock result: 0
   Lock SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory cannot be locked from secured state with incorrect AP
Lock fail tid
   Lock result: 0
   Lock SUCCESSFUL

Report No.: EMC94348B Rev. 2
Date: 2017-12-08
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory cannot be locked from open state
Lock pass tid ABCDEF01
  Lock result: 1
  Lock SUCCESSFUL
# TID = E2001050, PL; AP = ABCDEF01, PL; Verifies memory can be locked; TC = L_T_PL_2
# This TC attempts to set the lock bit, when the AP is non-zero and the TID is permalocked, using an incorrect AP, no AP, and the correct AP.

### Now working on USER memory ###
# Assumes a 64-bit programmable USER
# USER = FFFFFFFEEEDDDDCCCC
#UUU Lock pass user ABCDEF01
# USER = FFFFFFFEEEDDDDCCCC, L; Locks the USER memory
# Sets the Lock bit in the USER field to a one
# This action is required first in order to Permalock using PermLock directive
#UUU PermLock pass user ABCDEF01
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Permanently locks the USER bank from secured state; TC = PL_U_L_2
# Sets the PermaLock bit in the USER field to a one
#UUU Write fail 4444 user,3
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory cannot write from open state; TC = W_U_PL_P_2
#UUU Read pass 3333 user,3,1
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory can be read from open state; TC = R_U_PL_P_2
#UUU Write fail 4444 user,3 ABCDEF01
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory cannot write from secured state
#UUU Read pass 3333 user,3,1 ABCDEF01
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory can be read from secured state
#UUU Unlock fail user ABCDEF01
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory cannot be unlocked from secured state; TC = U_U_PL_2
#UUU PermUnLock fail user ABCDEF01
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_U_PL_2
#UUU Lock fail user 12345678
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory cannot be locked from secured state with incorrect AP
#UUU Lock fail user
# USER = FFFFFFFEEEDDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory cannot be locked from open state

Report No.: EMC94348B Rev. 2
# Lock pass user ABCDEF01
# USER = FFFFFFFEEDDDCCCC, PL; AP = ABCDEF01, PL; Verifies memory can be locked while Permaunlocked from secured state; TC = L_U_PL_2

### Now working on Kill password ###
Lock pass kpass ABCDEF01
  Lock result: 1
  Lock SUCCESSFUL
# KP = AAAAAAAA, L; Locks the Kill password
# Sets the Lock bit in the KP field to a one
# This action is required first in order to Permalock using PermLock directive
PermLock pass kpass ABCDEF01
  PermLock result: 1
  PermLock SUCCESSFUL
# KP = AAAAAAAA, PL; Permanently locks the kill password from secured state; TC = PL_KP_L_2
# Sets the PermaLock bit in the KP field to a one
# This TC makes the KP permalocked, when the KP was locked and the AP is non-zero.

Write fail FFFFFFFF kpass
  Write result: 0
  Write SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot write from open state; TC = W_KP_PL_NZ_2
# This TC (and the subsequent directive from the secured state) attempt to write to the KP, when the KP is permalocked and non-zero and the AP is non-zero.

Read fail AAAAAAAA kpass
  Read data:
    Read SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot read from open state; TC = R_KP_PL_2
# This TC (and the subsequent directive from the secured state) attempt to read the KP, when the KP is permalocked and non-zero and the AP is non-zero.

Write fail FFFFFFFF kpass ABCDEF01
  Write result: 0
  Write SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot write from secured state
Read fail AAAAAAAA kpass ABCDEF01
  Read data:
    Read SUCCESSFUL
# KP = AAAAAAAA, PL; Verifies memory cannot read from secured state
Unlock fail kpass ABCDEF01
  UnLock result: 0
  UnLock SUCCESSFUL
  # KP = AAAAAAAA, PL; Verifies memory cannot be unlocked from secured state; TC = U_KP_PL_2
  # This TC attempts to deset the lock bit to the KP, when the KP is permalocked and the AP is non-zero.

PermUnLock fail kpass ABCDEF01
  PermUnLock result: 0
  PermUnLock SUCCESSFUL
  # KP = AAAAAAAA, PL; Verifies memory cannot be Permaunlocked from secured state; TC = PU_KP_PL_2
  # This TC attempts to deset the permalock bit to the KP, when the KP is permalocked and the AP is non-zero.

Lock fail kpass 12345678
  Lock result: 0
  Lock SUCCESSFUL
  # KP = AAAAAAAA, PL; AP = ABCDEF01, PL; Verifies memory cannot be locked from secured state with incorrect AP
Lock fail kpass
  Lock result: 0
  Lock SUCCESSFUL
  # KP = AAAAAAAA, PL; AP = ABCDEF01, PL; Verifies memory cannot be locked from open state
Lock pass kpass ABCDEF01
  Lock result: 1
  Lock SUCCESSFUL
  # KP = AAAAAAAA, PL; AP = ABCDEF01, PL; Verifies lock bit still set from secured state; TC = L_KP_PL_2
  # This TC attempts to set the lock bit, when the AP is non-zero and the KP is permalocked, using an incorrect AP, no AP, and the correct AP.

disconnect
  Disconnected.
Script completed without failures

**SQ_EPC_TEST**

# Created by Impinj and modified by MET Laboratories for use in EPCglobal sanctioned Gen2 RFID interoperability testing
# Permission granted for free use only for EPCglobal sponsored interoperability testing activities
# EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable for any
# injury, loss, damages, financial or otherwise, arising from, related to, or caused by the
# use of this
# document. Use of said document does not guarantee compliance with applicable state
# and/or federal laws.
# User is responsible for the interpretation of and compliance with applicable pedigree
# laws. The use of
# said document shall constitute your express consent to the foregoing disclaimer.

# Connect to reader in FCC mode and set Tx power to 30 dBm
connect 169.254.10.1 FCC 201
   Connected to reader 169.254.10.1, type: FCC, mode: 201
   power 20
      Power set to 20

# Program tag with 96-bit EPC "111122223333444455556666"
WrRd pass 34001112222333444455556666 epc
   WrRd Write result: 1
   WrRd read result: 34001112222333444455556666
   WrRd SUCCESSFUL

# Write non-zero access password
Write pass 11111111 apass
   Write result: 1
   Write SUCCESSFUL

# Matching EPC, matching Query target (A->A)
SQ pass epc 96 s0 000 32 96 111122223333444455556666 0 s0 00 A
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL

# Matching EPC, non-matching Query target (A->A)
SQ fail epc 96 s0 000 32 96 111122223333444455556666 0 s0 00 B
   SQ result: 0
   SQ SUCCESSFUL

# Non-matching EPC, matching Query target (A->B)
SQ pass epc 96 s0 000 32 96 111122223333444455556667 0 s0 00 B
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (A->B)
SQ fail epc 96 s0 000 32 96 1111222233334444555566667 0 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S0_2
#
# Matching EPC, matching Query target (A->A)
SQ pass epc 96 s0 001 34 64 444488888cccd1111 0 s0 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (A->A)
SQ fail epc 96 s0 001 34 64 444488888cccd1111 0 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A)
SQ pass epc 96 s0 001 34 64 444488888cccd11FF 0 s0 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (A)
SQ fail epc 96 s0 001 34 64 444488888cccd1111 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S0_3
#
# Matching EPC, matching Query target (A)
SQ pass epc 96 s0 010 37 48 222444466668 0 s0 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (A)
SQ fail epc 96 s0 010 37 48 222444466668 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A->B)
SQ pass epc 96 s0 010 37 48 22244446666FF 0 s0 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Session (A->B)
SQ fail epc 96 s0 010 37 48 2224444666FF 0 s0 00 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S0_4
#
# Matching EPC, matching Query target (A->B)
SQ pass epc 96 s0 011 43 24 891111 0 s0 00 B
  SQ result: 1
  EPC result: 1112222333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Session (A->B)
SQ fail epc 96 s0 011 43 24 891111 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A)
SQ pass epc 96 s0 011 43 24 89111F 0 s0 00 A
  SQ result: 1
  EPC result: 11112222333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (A)
SQ fail epc 96 s0 011 43 24 89111F 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S0_5
#
# Matching EPC, matching Query target (A->B)
SQ pass epc 96 s0 100 55 12 111 0 s0 00 B
  SQ result: 1
  EPC result: 1112222333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (A->B)
SQ fail epc 96 s0 100 55 12 111 0 s0 00 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A->A)
SQ pass epc 96 s0 100 55 12 11F 0 s0 00 A
  SQ result: 1
  EPC result: 11112222333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (A->A)
SQ fail epc 96 s0 100 55 12 11F 0 s0 11 A
SQ result: 0
SQ SUCCESSFUL

# SQ_E_S0_6
#
# Matching EPC, matching Query target (A->B)
SQ pass epc 96 s0 101 79 6 A0 0 s0 00 B
  SQ result: 1
  EPC result: 111122223334444555556666
  SQ SUCCESSFUL

# Matching EPC, non-matching Query Session (A->B)
SQ fail epc 96 s0 101 79 6 A0 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL

# Non-matching EPC, matching Query target (A)
SQ pass epc 96 s0 101 79 6 AF 0 s0 00 A
  SQ result: 1
  EPC result: 111122223334444555556666
  SQ SUCCESSFUL

# Non-matching EPC, non-matching Query target (A)
SQ fail epc 96 s0 101 79 6 AF 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S0_7
#
# Matching EPC, matching Query target (A)
SQ pass epc 96 s0 110 95 3 2 0 s0 00 A
  SQ result: 1
  EPC result: 111122223334444555556666
  SQ SUCCESSFUL

# Matching EPC, non-matching Query target (A)
SQ fail epc 96 s0 110 95 3 2 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL

# Non-matching EPC, matching Query target (A->A)
SQ pass epc 96 s0 110 95 3 F 0 s0 00 A
  SQ result: 1
  EPC result: 111122223334444555556666
  SQ SUCCESSFUL

# Non-matching EPC, non-matching Query Sel (A->A)
SQ fail epc 96 s0 110 95 3 F 0 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL
# SQ_E_S0_8

# Matching EPC, matching Query target (A)
SQ pass epc 96 s0 111 127 1 0 0 s0 00 A
  SQ result: 1
  EPC result: 11112223333444455556666
  SQ SUCCESSFUL

# Matching EPC, non-matching Query Sel (A)
SQ fail epc 96 s0 111 127 1 0 0 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL

# Non-matching EPC, matching Query target (A>B)
SQ pass epc 96 s0 111 127 1 F 0 s0 00 B
  SQ result: 1
  EPC result: 11112223333444455556666
  SQ SUCCESSFUL

# Non-matching EPC, non-matching Query target (A>B)
SQ fail epc 96 s0 111 127 1 F 0 s0 00 A
  SQ result: 0
  SQ SUCCESSFUL

# Lock epc
Lock pass epc 11111111
  Lock result: 1
  Lock SUCCESSFUL

# SQ_E_S1_1

# Matching EPC, matching Query target (A>A)
SQ pass epc 96 s1 000 117 1 8 0 s1 00 A
  SQ result: 1
  EPC result: 11112223333444455556666
  SQ SUCCESSFUL

# Matching EPC, non-matching Query target (A>A)
SQ fail epc 96 s1 000 117 1 8 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL

# Non-matching EPC, matching Query target (A>B)
SQ pass epc 96 s1 000 117 1 0 0 s1 00 B
  SQ result: 1
  EPC result: 11112223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Session (A->B)
SQ fail epc 96 s1 000 117 1 0 0 s1 00 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S1_2
#
# Matching EPC, matching Query target (B->A)
SQ pass epc 96 s1 001 97 3 A 0 s1 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (B->A)
SQ fail epc 96 s1 001 97 3 A 0 s1 11 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A)
SQ pass epc 96 s1 001 97 3 F 0 s1 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (A)
SQ fail epc 96 s1 001 97 3 F 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S1_3
#
# Matching EPC, matching Query target (A)
SQ pass epc 96 s1 010 83 6 20 0 s1 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (A)
SQ fail epc 96 s1 010 83 6 20 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A->B)
SQ pass epc 96 s1 010 83 6 2F 0 s1 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Session (A->B)
SQ fail epc 96 s1 010 83 6 2F 0 s0 00 B
  SQ result: 0
SQ SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send Sx INV to A
# Do not send correct query to avoid flipping flag back to B by ACKing it
SQ fail epc 96 s0 000 32 0 0 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s1 000 32 0 0 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s2 000 32 0 0 0 s2 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s3 000 32 0 0 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S1_4
#
# Matching EPC, matching Query target (A->B)
SQ pass epc 96 s1 011 64 12 333 0 s1 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Give an extra command so that the flag will toggle back to A
# Read pass - epc
# Matching EPC, non-matching Query Sel (B->A)
SQ fail epc 96 s1 011 64 12 333 0 s1 11 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A)
SQ pass epc 96 s1 011 64 12 33F 0 s1 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Give flag time to toggle, due to matching Query
# Read pass - epc 11111111
# Non-matching EPC, non-matching Query target (A)
SQ fail epc 96 s1 011 64 12 33F 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send Sx INV to A
# Do not send correct query to avoid flipping flag back to B by ACKing it
SQ fail epc 96 s0 000 32 0 0 0 s0 00 B
SQ result: 0
SQ SUCCESSFUL
SQ fail epc 96 s1 000 32 0 0 0 s1 00 B
SQ result: 0
SQ SUCCESSFUL
SQ fail epc 96 s2 000 32 0 0 0 s2 00 B
SQ result: 0
SQ SUCCESSFUL
SQ fail epc 96 s3 000 32 0 0 0 s3 00 B
SQ result: 0
SQ SUCCESSFUL

# SQ_E_S1_5
#
# Matching EPC, matching Query target (A->B)
SQ pass epc 96 s1 100 46 24 48888C 0 s1 00 B
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL
# Matching EPC, non-matching Query target (A->B)
SQ fail epc 96 s1 100 46 24 48888C 0 s1 00 A
SQ result: 0
SQ SUCCESSFUL
# Non-matching EPC, matching Query target (B->A)
SQ pass epc 96 s1 100 46 24 48888F 0 s1 00 A
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (A->A)
SQ fail epc 96 s1 100 46 24 48888F 0 s1 11 A
SQ result: 0
SQ SUCCESSFUL

# SQ_E_S1_6
#
# Matching EPC, matching Query target (A->B)
SQ pass epc 96 s1 101 42 48 448888ccccd11 0 s1 00 B
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL
# Matching EPC, non-matching Query target (B->B)
SQ fail epc 96 s1 101 42 48 448888ccccd11 0 s1 00 A
SQ result: 0
SQ SUCCESSFUL
# Non-matching EPC, matching Query target (B)
SQ pass epc 96 s1 101 42 48 448888ccccd1F 0 s1 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL

# Non-matching EPC, non-matching Query Session (B)
SQ fail epc 96 s1 101 42 48 448888ccccd1F 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send Sx INV to A
# Do not send correct query to avoid flipping flag back to B by ACKing it
SQ fail epc 96 s0 000 32 0 0 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s1 000 32 0 0 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s2 000 32 0 0 0 s2 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s3 000 32 0 0 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S1_7
#
# Matching EPC, matching Query target (A)
SQ pass epc 96 s1 110 38 64 4448888ccccd11115 0 s1 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL

# Non-matching EPC, matching Query target (B->A)
SQ fail epc 96 s1 110 38 64 4448888ccccd1111F 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL

# Non-matching EPC, matching Query target (A->A)
SQ fail epc 96 s1 110 38 64 4448888ccccd1111F 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL
# SQ_E_S1_8

# Matching EPC, matching Query target (A)
SQ pass epc 96 s1 111 32 96 111122223333444455556666 0 s1 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL

# Matching EPC, non-matching Query target (A)
SQ fail epc 96 s1 111 32 96 111122223333444455556666 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL

# Non-matching EPC, non-matching Query target (A->B)
SQ pass epc 96 s1 111 32 96 111122223333444455556666F 0 s1 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL

# Non-matching EPC, matching Query Sel (B->A)
SQ fail epc 96 s1 111 32 96 111122223333444455556666F 0 s1 11 A
  SQ result: 0
  SQ SUCCESSFUL

# Unlock epc
UnLock pass epc 11111111
  UnLock result: 1
  UnLock SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send Sx INV to A
# Do not send correct query to avoid flipping flag back to B by ACKing it
SQ fail epc 96 s0 000 32 0 0 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s1 000 32 0 0 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s2 000 32 0 0 0 s2 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s3 000 32 0 0 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Matching EPC, matching Query target (A->A)
SQ pass epc 96 s2 000 32 96 111122223333444455556666 0 s2 00 A
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (A->A)
SQ fail epc 96 s2 000 32 96 111122223333444455556666 0 s2 11 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A->B)
SQ pass epc 96 s2 000 32 96 111122223333444455556666F 0 s2 00 B
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (B->B)
SQ fail epc 96 s2 000 32 96 111122223333444455556666F 0 s2 00 A
  SQ result: 0
  SQ SUCCESSFUL
# SQ_E_S2_2
#
# Tag is in B from previous test ( persistent session )
# Matching EPC, matching Query target (B->A)
SQ pass epc 96 s2 001 32 95 111122223333444455556666 0 s2 00 A
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (A->A)
SQ fail epc 96 s2 001 32 95 111122223333444455556666 0 s2 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A)
SQ pass epc 96 s2 001 32 95 111122223333444455556666F 0 s2 00 A
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (A)
SQ fail epc 96 s2 001 32 95 111122223333444455556666F 0 s2 11 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S2_3
#
# Matching EPC, matching Query target (A)
SQ pass epc 96 s2 010 44 48 122223333444 0 s2 00 A  
  SQ result: 1  
  EPC result: 1111222233344455556666  
  SQ SUCCESSFUL  
# Matching EPC, non-matching Query target (A)  
SQ fail epc 96 s2 010 44 48 122223333444 0 s2 00 B  
  SQ result: 0  
  SQ SUCCESSFUL  
# Non-matching EPC, matching Query target (A->B)  
SQ pass epc 96 s2 010 44 48 12222333344F 0 s2 00 B  
  SQ result: 1  
  EPC result: 1111222233344455556666  
  SQ SUCCESSFUL  
# Non-matching EPC, non-matching Query session (B->B)  
SQ fail epc 96 s2 010 44 48 12222333344F 0 s1 00 B  
  SQ result: 0  
  SQ SUCCESSFUL  

# SQ_E_S2_4  
#  
# Tag is in B from previous test ( persistent session )  
# Matching EPC, matching Query target (B->A)  
SQ pass epc 96 s2 011 56 30 22333344 0 s2 00 A  
  SQ result: 1  
  EPC result: 1111222233344455556666  
  SQ SUCCESSFUL  
# Matching EPC, non-matching Query session (A->B)  
SQ fail epc 96 s2 011 56 30 22333344 0 s0 00 B  
  SQ result: 0  
  SQ SUCCESSFUL  
# Non-matching EPC, matching Query target (B)  
SQ pass epc 96 s2 011 56 30 2233334F 0 s2 00 B  
  SQ result: 1  
  EPC result: 1111222233344455556666  
  SQ SUCCESSFUL  
# Non-matching EPC, non-matching Query target (B)  
SQ fail epc 96 s2 011 56 30 2233334F 0 s2 00 A  
  SQ result: 0  
  SQ SUCCESSFUL  

# SQ_E_S2_5  
#  
# Tag is in B from previous test ( persistent session )  
# Matching EPC, matching Query target (B->B)  
SQ pass epc 96 s2 100 67 20 999A2 0 s2 00 B
SQ result: 1  
EPC result: 111122223333444455556666  
SQ SUCCESSFUL

# Matching EPC, non-matching Query target (B->B)  
SQ fail epc 96 s2 100 67 20 999A2 0 s2 00 A  
SQ result: 0  
SQ SUCCESSFUL

# Non-matching EPC, matching Query target (B->A)  
SQ pass epc 96 s2 100 67 20 999AF 0 s2 00 A  
SQ result: 1  
EPC result: 111122223333444455556666  
SQ SUCCESSFUL

# Non-matching EPC, non-matching Query Sel (A->A)  
SQ fail epc 96 s2 100 67 20 999AF 0 s2 11 A  
SQ result: 0  
SQ SUCCESSFUL

# SQ_E_S2_6  
#  
# Matching EPC, matching Query target (A->B)  
SQ pass epc 96 s2 101 75 6 98 0 s2 00 B  
SQ result: 1  
EPC result: 111122223333444455556666  
SQ SUCCESSFUL

# Matching EPC, non-matching Query target (B->B)  
SQ fail epc 96 s2 101 75 6 98 0 s2 00 A  
SQ result: 0  
SQ SUCCESSFUL

# Non-matching EPC, matching Query target (B)  
SQ pass epc 96 s2 101 75 6 9F 0 s2 00 B  
SQ result: 1  
EPC result: 111122223333444455556666  
SQ SUCCESSFUL

# Non-matching EPC, non-matching Query session (B)  
SQ fail epc 96 s2 101 75 6 9F 0 s3 00 B  
SQ result: 0  
SQ SUCCESSFUL

# SQ_E_S2_7  
#  
# Tag is in B from previous test ( persistent session )  
# Matching EPC, matching Query target (B)  
SQ pass epc 96 s2 110 83 3 2 0 s2 00 B  
SQ result: 1  
EPC result: 111122223333444455556666  

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
SQ SUCCESSFUL
# Matching EPC, non-matching Query Session (B)
SQ fail epc 96 s2 110 83 3 2 0 s1 00 B
    SQ result: 0
    SQ SUCCESSFUL
# Non-matching EPC, matching Query target (B->A)
SQ pass epc 96 s2 110 83 3 F 0 s2 00 A
    SQ result: 1
    EPC result: 111122223333444455556666
    SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (A->A)
SQ fail epc 96 s2 110 83 3 F 0 s2 00 B
    SQ result: 0
    SQ SUCCESSFUL

# SQ_E_S2_8
#
# Matching EPC, matching Query target (A)
SQ pass epc 96 s2 111 126 1 8 0 s2 00 A
    SQ result: 1
    EPC result: 111122223333444455556666
    SQ SUCCESSFUL
# Matching EPC, non-matching Query target (A)
SQ fail epc 96 s2 111 126 1 8 0 s2 00 B
    SQ result: 0
    SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A->B)
SQ pass epc 96 s2 111 126 1 0 0 s2 00 B
    SQ result: 1
    EPC result: 111122223333444455556666
    SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (B->A)
SQ fail epc 96 s2 111 126 1 0 0 s2 11 A
    SQ result: 0
    SQ SUCCESSFUL

# Lock epc
Lock pass epc 11111111
    Lock result: 1
    Lock SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send Sx INV to A
# Do not send correct query to avoid flipping flag back to B by ACKing it
SQ fail epc 96 s0 000 32 0 0 0 s0 00 B
    SQ result: 0
SQ SUCCESSFUL
SQ fail epc 96 s1 000 32 0 0 0 s1 00 B
SQ result: 0
SQ SUCCESSFUL
SQ fail epc 96 s2 000 32 0 0 0 s2 00 B
SQ result: 0
SQ SUCCESSFUL
SQ fail epc 96 s3 000 32 0 0 0 s3 00 B
SQ result: 0
SQ SUCCESSFUL

###

# SQ_E_S3_1
#
# Matching EPC, matching Query target (A->A)
SQ pass epc 96 s3 000 124 1 0 0 s3 00 A
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL

# Matching EPC, non-matching Query Sel (A->A)
SQ fail epc 96 s3 000 124 1 0 0 s3 11 B
SQ result: 0
SQ SUCCESSFUL

# Non-matching EPC, matching Query target (A->B)
SQ pass epc 96 s3 000 124 1 F 0 s3 00 B
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL

# Non-matching EPC, non-matching Query target (B->B)
SQ fail epc 96 s3 000 124 1 F 0 s3 00 A
SQ result: 0
SQ SUCCESSFUL

# SQ_E_S3_2
#
# Tag is in B from previous test (persistent session)
# Matching EPC, matching Query target (B->A)
SQ pass epc 96 s3 001 102 3 4 0 s3 00 A
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL

# Matching EPC, non-matching Query target (A->A)
SQ fail epc 96 s3 001 102 3 4 0 s3 00 B  

---

Report No.: EMC94348B Rev. 2

Date: 2017-12-08

Page 93 of 114
SQ result: 0
SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A)
SQ pass epc 96 s3 001 102 3 4 0 s3 00 A
  SQ result: 1
  EPC result: 11112222333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (A)
SQ fail epc 96 s3 001 102 3 4 0 s3 11 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S3_3
#
# Matching EPC, matching Query target (A)
SQ pass epc 96 s3 010 87 6 20 0 s3 00 A
  SQ result: 1
  EPC result: 11112222333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (A)
SQ fail epc 96 s3 010 87 6 20 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query session (A>B)
SQ pass epc 96 s3 010 87 6 2F 0 s3 00 B
  SQ result: 1
  EPC result: 11112222333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query session (B>B)
SQ fail epc 96 s3 010 87 6 2F 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_S3_4
#
# Tag is in B from previous test ( persistent session )
# Matching EPC, matching Query target (B->A)
SQ pass epc 96 s3 011 59 12 119 0 s3 00 A
  SQ result: 1
  EPC result: 11112222333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query session (A->B)
SQ fail epc 96 s3 011 59 12 119 0 s2 00 B
  SQ result: 0
  SQ SUCCESSFUL

Report No.: EMC94348B Rev. 2

Date: 2017-12-08

Page 94 of 114
# Non-matching EPC, matching Query target (B)
SQ pass epc 96 s3 011 59 12 11F 0 s3 00 B
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL

# Non-matching EPC, non-matching Query target (B)
SQ fail epc 96 s3 011 59 12 11F 0 s3 00 A
   SQ result: 0
   SQ SUCCESSFUL

# SQ_E_S3_5
#
# Tag is in B from previous test (persistent session)
# Matching EPC, matching Query target (B->B)
SQ pass epc 96 s3 100 45 24 244446 0 s3 00 B
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL

# Matching EPC, non-matching Query target (B->B)
SQ fail epc 96 s3 100 45 24 244446 0 s3 00 A
   SQ result: 0
   SQ SUCCESSFUL

# Non-matching EPC, matching Query target (B->A)
SQ pass epc 96 s3 100 45 24 24444F 0 s3 00 A
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL

# Non-matching EPC, non-matching Query Sel (A->A)
SQ fail epc 96 s3 100 45 24 24444F 0 s3 11 A
   SQ result: 0
   SQ SUCCESSFUL

# SQ_E_S3_6
#
# Matching EPC, matching Query target (A->B)
SQ pass epc 96 s3 101 41 48 2244446666688 0 s3 00 B
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL

# Matching EPC, non-matching Query session (B->B)
SQ fail epc 96 s3 101 41 48 2244446666688 0 s0 00 B
   SQ result: 0
   SQ SUCCESSFUL

# Non-matching EPC, matching Query target (B)
SQ pass epc 96 s3 101 41 48 2244446666688F 0 s3 00 B

Report No.: EMC94348B Rev. 2

Date: 2017-12-08  Page 95 of 114
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (B)
SQ fail epc 96 s3 101 41 48 224444666668F 0 s3 00 A
    SQ result: 0
    SQ SUCCESSFUL

# SQ_E_S3_7
#
# Tag is in B from previous test ( persistent session )
# Matching EPC, matching Query target (B)
SQ pass epc 96 s3 110 36 64 1112222333344445 0 s3 00 B
    SQ result: 1
    EPC result: 111122223333444455556666
    SQ SUCCESSFUL
# Matching EPC, non-matching Query session (B)
SQ fail epc 96 s3 110 36 64 1112222333344445 0 s1 00 B
    SQ result: 0
    SQ SUCCESSFUL
# Non-matching EPC, matching Query target (B->A)
SQ pass epc 96 s3 110 36 64 1112222333344445F 0 s3 00 A
    SQ result: 1
    EPC result: 111122223333444455556666
    SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A)
SQ pass epc 96 s3 111 32 96 111122223333444455556666 0 s3 00 A
    SQ result: 1
    EPC result: 111122223333444455556666
    SQ SUCCESSFUL
# Matching EPC, matching Query target (A)
SQ fail epc 96 s3 111 32 96 111122223333444455556666 0 s3 00 B
    SQ result: 0
    SQ SUCCESSFUL
# Non-matching EPC, matching Query target (A->B)
SQ pass epc 96 s3 111 32 96 111122223333444455556666F 0 s3 00 B
    SQ result: 1
    EPC result: 111122223333444455556666

Report No.: EMC94348B Rev. 2

Date: 2017-12-08
SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (B->A)
SQ fail epc 96 s3 111 32 96 111122223333444455556666F 0 s3 11 B
  SQ result: 0
  SQ SUCCESSFUL

# Unlock epc
UnLock pass epc 11111111
  UnLock result: 1
  UnLock SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send Sx INV to A
# Do not send correct query to avoid flipping flag back to B by ACKing it
SQ fail epc 96 s0 000 32 0 0 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s1 000 32 0 0 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s2 000 32 0 0 0 s2 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail epc 96 s3 000 32 0 0 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_1
# Matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 000 126 2 8 1 s0 11 A
  SQ result: 1
  EPC result:
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (SL)
SQ fail epc 96 sl 000 126 2 8 1 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (SL->~SL)
SQ pass epc 96 sl 000 126 2 F 1 s0 10 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (~SL->~SL)
SQ fail epc 96 sl 000 126 2 F 1 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_2
#
# Matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 001 63 64 1999a2222aaab333 1 s0 11 A
  SQ result: 1
  EPC result: 63
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (SL)
SQ fail epc 96 sl 001 63 64 1999a2222aaab333 1 s0 10 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (SL)
SQ pass epc 96 sl 001 63 64 1999a2222aaab33F 1 s0 11 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (SL)
SQ fail epc 96 sl 001 63 64 1999a2222aaab33F 1 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_3
#
# Tag has SL from previous test ( SL flag has persistence )
# Matching EPC, matching Query Sel (SL)
SQ pass epc 96 sl 010 127 1 0 1 s0 11 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Target (SL)
SQ fail epc 96 sl 010 127 1 0 1 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (SL->~SL)
SQ pass epc 96 sl 010 127 1 F 1 s0 10 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (~SL->~SL)
SQ fail epc 96 sl 010 127 1 F 1 s0 11 A
SQ result: 0
SQ SUCCESSFUL

# SQ_E_SL_4
#
# Matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 011 63 8 19 1 s0 11 A
SQ result: 1
EPC result: 99a2222aaab3334c
SQ SUCCESSFUL
# Matching EPC, non-matching Query target (SL->~SL)
SQ fail epc 96 sl 011 63 8 19 1 s0 10 B
SQ result: 0
SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (~SL)
SQ pass epc 96 sl 011 63 8 1F 1 s0 10 A
SQ result: 1
EPC result: 111122223334445556666
SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (~SL)
SQ fail epc 96 sl 011 63 8 1F 1 s0 11 A
SQ result: 0
SQ SUCCESSFUL

# SQ_E_SL_5
#
# Matching EPC, matching Query Sel (~SL->~SL)
SQ pass epc 96 sl 100 58 12 88C 1 s0 10 A
SQ result: 1
EPC result: ccd11115555999a2
SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (~SL->~SL)
SQ fail epc 96 sl 100 58 12 88C 1 s0 11 A
SQ result: 0
SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 100 58 12 88F 1 s0 11 A
SQ result: 1
EPC result: 111122223334445556666
SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (SL->SL)
SQ fail epc 96 sl 100 58 12 88F 1 s0 11 B
SQ result: 0
SQ SUCCESSFUL
# SQ_E_SL_6
#
# Tag has SL from previous test (SL flag has persistence)
# Matching EPC, matching Query SL (SL->~SL)
SQ pass epc 96 sl 101 81 16 8888 1 s0 10 A
  SQ result: 1
  EPC result: aaaaccccd
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (~SL->~SL)
SQ fail epc 96 sl 101 81 16 8888 1 s0 10 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (~SL)
SQ pass epc 96 sl 101 81 16 888F 1 s0 10 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (~SL)
SQ fail epc 96 sl 101 81 16 8888 1 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_7
#
# Matching EPC, matching Query Sel (~SL)
SQ pass epc 96 sl 110 32 5 10 1 s0 10 A
  SQ result: 1
  EPC result: 222444466668888aaaaccccb
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (~SL)
SQ fail epc 96 sl 110 32 5 10 1 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 110 32 5 1F 1 s0 11 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (SL->SL)
SQ fail epc 96 sl 110 32 5 1F 1 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_8
#
# Tag has SL from previous test (SL flag has persistence)
# Matching EPC, matching Query Sel (SL)
SQ pass epc 96 sl 111 124 3 6 1 s0 11 A
  SQ result: 1
  EPC result: 63
  SQ SUCCESSFUL

# Matching EPC, non-matching Query target (SL)
SQ fail epc 96 sl 111 124 3 6 1 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL

# Non-matching EPC, matching Query Sel (SL->~SL)
SQ pass epc 96 sl 111 124 3 F 1 s0 10 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL

# Non-matching EPC, non-matching Query Sel (~SL->SL)
SQ fail epc 96 sl 111 124 3 F 1 s0 10 A
  SQ result: 0
  SQ SUCCESSFUL

# Send SELECT with 0 length mask and Action 011 to send SL to ~SL
# Do not send correct query to avoid flipping S0 INV flag to B by ACKing it
SQ fail epc 96 sl 011 32 0 0 0 s0 10 B
  SQ result: 0
  SQ SUCCESSFUL

###############################################
# Lock epc
Lock pass epc 11111111
  Lock result: 1
  Lock SUCCESSFUL

# SQ_E_SL_9
# # Matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 000 32 96 111122223333444455556666 1 s0 11 A
  SQ result: 1
  EPC result:
  SQ SUCCESSFUL

# Matching EPC, non-matching Query target (SL)
SQ fail epc 96 sl 000 32 96 111122223333444455556666 1 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (SL->~SL)
SQ pass epc 96 sl 000 32 96 111122223333444455556666F 1 s0 10 A
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL

# Non-matching EPC, non-matching Query Sel (~SL->~SL)
SQ fail epc 96 sl 000 32 96 111122223333444455556666F 1 s0 11 A
   SQ result: 0
   SQ SUCCESSFUL

# SQ_E_SL_10

# Matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 001 32 95 111122223333444455556666 1 s0 11 A
   SQ result: 1
   EPC result: 63
   SQ SUCCESSFUL

# Matching EPC, non-matching Query Sel (SL)
SQ fail epc 96 sl 001 32 95 111122223333444455556666 1 s0 10 A
   SQ result: 0
   SQ SUCCESSFUL

# Non-matching EPC, matching Query Sel (SL)
SQ pass epc 96 sl 001 32 95 111122223333444455556666 1 s0 11 A
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL

# Non-matching EPC, non-matching Query target (SL)
SQ fail epc 96 sl 001 32 95 111122223333444455556666F 1 s0 11 B
   SQ result: 0
   SQ SUCCESSFUL

# SQ_E_SL_11

# Tag has SL from previous test ( SL flag has persistence )
# Matching EPC, matching Query Sel (SL)
SQ pass epc 96 sl 010 44 48 122223333444 1 s0 11 A
   SQ result: 1
   EPC result: 4555566668
   SQ SUCCESSFUL

# Matching EPC, non-matching Query target (SL)
SQ fail epc 96 sl 010 44 48 122223333444 1 s0 11 B
   SQ result: 0
   SQ SUCCESSFUL

# Non-matching EPC, matching Query Sel (SL->~SL)
SQ pass epc 96 sl 010 44 48 122223333444F 1 s0 10 A
SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL

# Non-matching EPC, non-matching Query Sel (~SL->~SL)
SQ fail epc 96 sl 010 44 48 12222333344F 1 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_12
#
# Matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 011 56 30 22333344 1 s0 11 A
  SQ result: 1
  EPC result: 1115555999ac
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (SL->~SL)
SQ fail epc 96 sl 011 56 30 22333344 1 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (~SL)
SQ pass epc 96 sl 011 56 30 2233334F 1 s0 10 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (~SL)
SQ fail epc 96 sl 011 56 30 2233334F 1 s0 10 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_13
#
# Matching EPC, matching Query Sel (~SL->~SL)
SQ pass epc 96 sl 100 67 20 999A2 1 s0 10 A
  SQ result: 1
  EPC result: 222aaab3335a
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (~SL->~SL)
SQ fail epc 96 sl 100 67 20 999AF 1 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 100 67 20 999A2 1 s0 11 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (SL->SL)
SQ fail epc 96 sl 100 67 20 999AF 1 s0 10 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_14
#
# Tag has SL from previous test (SL flag has persistence)
# Matching EPC, matching Query Sel (SL->~SL)
SQ pass epc 96 sl 101 75 6 9A 1 s0 10 A
  SQ result: 1
  EPC result: 8888aaaacccc
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (~SL->~SL)
SQ fail epc 96 sl 101 75 6 9A 1 s0 11 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (~SL)
SQ pass epc 96 sl 101 75 6 9F 1 s0 10 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (~SL)
SQ fail epc 96 sl 101 75 6 9F 1 s0 10 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_E_SL_15
#
# Matching EPC, matching Query Sel (~SL)
SQ pass epc 96 sl 110 95 13 2AAA 1 s0 10 A
  SQ result: 1
  EPC result: 566663
  SQ SUCCESSFUL
# Matching EPC, non-matching Query target (~SL)
SQ fail epc 96 sl 110 95 13 2AAA 1 s0 10 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (~SL->SL)
SQ pass epc 96 sl 110 95 13 2AA0 1 s0 11 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query Sel (SL->SL)
SQ fail epc 96 sl 110 95 13 2AA0 1 s0 10 A
SQ result: 0
SQ SUCCESSFUL

# SQ_E_SL_16
#
# Tag has SL from previous test (SL flag has persistence)
# Matching EPC, matching Query Sel (SL)
SQ pass epc 96 sl 111 126 1 8 1 s0 11 A
  SQ result: 1
  EPC result: 63
  SQ SUCCESSFUL
# Matching EPC, non-matching Query Sel (SL)
SQ fail epc 96 sl 111 126 1 8 1 s0 10 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching EPC, matching Query Sel (SL->~SL)
SQ pass epc 96 sl 111 126 1 0 1 s0 10 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching EPC, non-matching Query target (~SL->SL)
SQ fail epc 96 sl 111 126 1 0 1 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL

# Unlock epc
UnLock pass epc 11111111
  UnLock result: 1
  UnLock SUCCESSFUL

# Send SELECT with 0 length mask and Action 011 to send SL to ~SL
# Do not send correct query to avoid flipping S0 INV flag to B by ACKing it
SQ fail epc 96 sl 011 32 0 0 0 s0 10 B
  SQ result: 0
  SQ SUCCESSFUL

###############################################################
######### Write zero access password
Write pass 00000000 apass
  Write result: 1
  Write SUCCESSFUL
disconnect  
    Disconnected.
Script completed without failures

**SQ_TID_TEST**

# Created by Impinj, and modified by MET Laboratories, for use in EPCglobal sanctioned Gen2 RFID interoperability testing  
# Permission granted for free use only for EPCglobal sponsored interoperability testing activities  
#  
# EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable for any  
# injury, loss, damages, financial or otherwise, arising from, related to, or caused by the use of this  
# document. Use of said document does not guarantee compliance with applicable state and/or federal laws.  
# User is responsible for the interpretation of and compliance with applicable pedigree laws. The use of  
# said document shall constitute your express consent to the foregoing disclaimer.

######################################################################
##########
# Connect to reader in FCC mode and set Tx power 29.0 30 dBm  
connect 169.254.10.1 FCC 201  
    Connected to reader 169.254.10.1, type: FCC, mode: 201  
power 20  
    Power set to 20

# Confirm that the TID is E280B0A1  
Read pass E280B0A1 tid  
    Read data: E280B0A1  
    Read SUCCESSFUL  
# Set the AP to a non-zero value  
Write pass 1111111111 apass  
    Write result: 1  
    Write SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send Sx INV to A  
# Do not send correct query to avoid flipping flag back to B by ACKing it  
SQ fail tid 96 s0 000 0 0 0 0 s0 00 B  
    SQ result: 0  
    SQ SUCCESSFUL
SQ fail tid 96 s1 000 0 0 0 0 s1 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail tid 96 s2 000 0 0 0 0 s2 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail tid 96 s3 000 0 0 0 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL
SQ fail tid 96 s1 100 0 0 0 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_T_S0_1
#
# Matching TID, matching Query target (A->A)
SQ pass tid 96 s0 000 0 16 E280 0 s0 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching TID, non-matching Query Sel (A->A)
SQ fail tid 96 s0 000 0 16 E280 0 s0 11 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching TID, matching Query target (A->B)
SQ pass tid 96 s0 000 0 16 E20F 0 s0 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching TID, non-matching Query target (A->B)
SQ fail tid 96 s0 000 0 16 E20F 0 s0 00 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_T_S0_2
#
# Matching TID, matching Query target (A->A)
SQ pass tid 96 s0 001 0 32 E280B0A1 0 s0 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching TID, non-matching Query target (A->A)
SQ fail tid 96 s0 001 0 32 E280B0A1 0 s0 00 B
SQ result: 0
SQ SUCCESSFUL
# Non-matching TID, matching Query target (A)
SQ pass tid 96 s0 001 0 32 E280104F 0 s0 00 A
SQ result: 1
EPC result: 111122233344455556666
SQ SUCCESSFUL
# Non-matching TID, non-matching Query Sel (A)
SQ fail tid 96 s0 001 0 32 E280104F 0 s0 11 B
SQ result: 0
SQ SUCCESSFUL

# Lock the TID
Lock pass tid 11111111
Lock result: 1
Lock SUCCESSFUL

# SQ_T_S1_1
#
# Matching TID, matching Query target (A)
SQ pass tid 96 s1 010 0 16 E280 0 s1 00 A
SQ result: 1
EPC result: 111122233344455556666
SQ SUCCESSFUL
# Matching TID, non-matching Query target (A)
SQ fail tid 96 s1 010 0 16 E280 0 s1 00 B
SQ result: 0
SQ SUCCESSFUL
# Non-matching TID, matching Query target (A->B)
SQ pass tid 96 s1 010 0 16 E20F 0 s1 00 B
SQ result: 1
EPC result: 111122233344455556666
SQ SUCCESSFUL
# Non-matching TID, non-matching Query session (B->B)
SQ fail tid 96 s1 010 0 16 E20F 0 s1 00 A
SQ result: 0
SQ SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send S1 INV to A
# Do not send correct query to avoid flipping S0 INV flag to B by ACKing it
SQ fail tid 96 s1 000 0 0 0 0 s1 00 B

Report No.: EMC94348B Rev. 2
Date: 2017-12-08
SQ result: 0
SQ SUCCESSFUL

# SQ_T_S1_2
#
# Matching TID, matching Query target (A->B)
SQ pass tid 96 s1 011 0 32 E280B0A1 0 s1 00 B
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL
# Give the flag time to toggle with the next command
#Read pass - tid
# Matching TID, non-matching Query session (B->A)
SQ fail tid 96 s1 011 0 32 E280B0A1 0 s1 00 B
   SQ result: 0
   SQ SUCCESSFUL
# Non-matching TID, matching Query target (A)
SQ pass tid 96 s1 011 0 32 E280104F 0 s1 00 A
   SQ result: 1
   EPC result: 111122223333444455556666
   SQ SUCCESSFUL
# Give the flag time to toggle with the next command
#Read pass - tid
# Non-matching TID, non-matching Query target (A)
SQ fail tid 96 s1 011 0 32 E280104F 0 s1 00 B
   SQ result: 0
   SQ SUCCESSFUL

# If TID capable of being unlocked:
#Unlock pass tid 11111111

# Send SELECT with 0 length mask and Action 000 to send S1 INV to A
# Do not send correct query to avoid flipping S0 INV flag to B by ACKing it
SQ fail tid 96 s1 000 0 0 0 s1 00 B
   SQ result: 0
   SQ SUCCESSFUL

# SQ_T_S2_1
#
# Matching TID, matching Query target (A->B)
SQ pass tid 96 s2 100 0 16 E280 0 s2 00 B
   SQ result: 1
EPC result: 111122223333444455556666
SQ SUCCESSFUL
# Matching TID, non-matching Query target (B->B)
SQ fail tid 96 s2 100 0 16 E280 0 s2 00 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching TID, matching Query target (B->A)
SQ pass tid 96 s2 100 0 16 E20F 0 s2 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching TID, non-matching Query target (A->A)
SQ fail tid 96 s2 100 0 16 E20F 0 s2 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_T_S2_2
#
# Matching TID, matching Query target (A->B)
SQ pass tid 96 s2 101 0 32 E280B0A1 0 s2 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching TID, non-matching Query Session (B->B)
SQ fail tid 96 s2 101 0 32 E280B0A1 0 s0 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching TID, matching Query target (B)
SQ pass tid 96 s2 101 0 32 E280104F 0 s2 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching TID, non-matching Query target (B)
SQ fail tid 96 s2 101 0 32 E280104F 0 s2 00 A
  SQ result: 0
  SQ SUCCESSFUL

# Lock the TID
Lock pass tid 11111111
  Lock result: 1
  Lock SUCCESSFUL

# Send SELECT with 0 length mask and Action 000 to send S2 INV to A
# Do not send correct query to avoid flipping S0 INV flag to B by ACKing it
SQ fail tid 96 s2 000 0 0 0 0 s2 00 B
# SQ_T_S3_1
#
# Matching TID, matching Query target (A)
SQ pass tid 96 s3 110 0 16 E280 0 s3 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching TID, non-matching Query target (A)
SQ fail tid 96 s3 110 0 16 E280 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching TID, matching Query target (B->A)
SQ pass tid 96 s3 110 0 16 E20F 0 s3 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching TID, non-matching Query Sel (A->A)
SQ fail tid 96 s3 110 0 16 E20F 0 s3 11 A
  SQ result: 0
  SQ SUCCESSFUL

# SQ_T_S3_2
#
# Matching TID, matching Query target (A)
SQ pass tid 96 s3 111 0 32 E280B0A1 0 s3 00 A
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Matching TID, non-matching Query target (A)
SQ fail tid 96 s3 111 0 32 E280B0A1 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching TID, matching Query target (A->B)
SQ pass tid 96 s3 111 0 32 E280104F 0 s3 00 B
  SQ result: 1
  EPC result: 111122223333444455556666
  SQ SUCCESSFUL
# Non-matching TID, non-matching Query Sel (B->A)
SQ fail tid 96 s3 111 0 32 E280104F 0 s3 11 A
  SQ result: 0
  SQ SUCCESSFUL
SQ result: 0
SQ SUCCESSFUL

# If TID capable of being unlocked:
#Unlock pass tid 11111111

# Send SELECT with 0 length mask and Action 000 to send S3 INV to A
# Do not send correct query to avoid flipping S0 INV flag to B by ACKing it
SQ fail tid 96 s3 000 0 0 0 s3 00 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_T_SL_1
#
# Matching TID, matching Query Sel (~SL->SL)
SQ pass tid 96 sl 000 0 16 E280 0 s0 11 A
  SQ result: 1
  EPC result: 1111222333444455556666
  SQ SUCCESSFUL
# Matching TID, non-matching Query Sel (SL->SL)
SQ fail tid 96 sl 000 0 16 E280 0 s0 10 A
  SQ result: 0
  SQ SUCCESSFUL
# Non-matching TID, matching Query Sel (~SL->~SL)
SQ pass tid 96 sl 000 0 16 E20F 0 s0 10 A
  SQ result: 1
  EPC result: 1111222333444455556666
  SQ SUCCESSFUL
# Non-matching TID, non-matching Query target (~SL->~SL)
SQ fail tid 96 sl 000 0 16 E20F 0 s0 10 B
  SQ result: 0
  SQ SUCCESSFUL

# SQ_T_SL_2
#
# Matching TID, matching Query Sel (~SL->SL)
SQ pass tid 96 sl 001 0 32 E280B0A1 0 s0 11 A
  SQ result: 1
  EPC result: 1111222333444455556666
  SQ SUCCESSFUL
# Matching TID, non-matching Query target (SL)
SQ fail tid 96 sl 001 0 32 E280B0A1 0 s0 11 B
SQ result: 0
SQ SUCCESSFUL
# Non-matching TID, matching Query Sel (SL)
SQ pass tid 96 sl 001 0 32 E280104F 0 s0 11 A
    SQ result: 1
    EPC result: 111122223333444455556666
    SQ SUCCESSFUL
# Non-matching TID, non-matching Query Sel (SL)
SQ fail tid 96 sl 001 0 32 E280104F 0 s0 10 A
    SQ result: 0
    SQ SUCCESSFUL

# Send SELECT with 0 length mask and Action 011 to send SL to ~SL
# Do not send correct query to avoid flipping S0 INV flag to B by ACKing it
SQ fail tid 96 sl 011 0 0 0 0 s0 10 B
    SQ result: 0
    SQ SUCCESSFUL

# disconnect
Disconnected.
Script completed without failures

4. MEASUREMENT UNCERTAINTY.

The following uncertainty values have been calculated and compared to the specified limits as in the document “Measurement Uncertainties RFID RF Tester. Version 1.0”.

The measurement uncertainty of the test equipment is defined by the sampling frequency, and using the formula:

\[ s = \frac{1}{\text{Sampling Frequency [Hz]}} \]

The default sampling frequency being 4 MHz, the measurement uncertainty is +/- 0.25µS.
5. Photographs

Photograph 1 – EM Microelectronic EM | Echo RFID Tag