CLAIMS

1. A secure element (100) for securely processing digital information, said digital information being segmented and stored in a plurality of M segments in a memory (200) external to the secure element (100), including:

- a secure processor (110) for securely processing the plurality of M segments of digital information-;

- a system (140) configured to load a segment of the plurality of M segments of digital information (Si) from the external memory (200) into the secure element (100), and pre-process said segment of digital information by executing a cryptographic algorithm before processing said segment of digital information by the secure processor (110);

wherein the system (140) is configured to determine an identification information of the segment of digital information that has been loaded, the determination of said identification information including obtaining a version number of said segment of digital information, and to execute the cryptographic algorithm by using said identification information;

characterized in that the system (140) is configured to load a version table (CtrlObj) containing the respective version numbers of the plurality of M segments of digital information, from the external memory (200) into an internal memory (122) of the secure element (100), and to read, from the version table (CtrlObj) stored in the internal memory (122), the version number (Vi) of each segment of digital information that is loaded. ~~in the version table (CtrlObj) stored in the internal memory (122).~~

2. The secure element according to claim 1, wherein the system (140) is configured to determine the identification information of the segment of digital information, the determination of said identification information further including obtaining a segment identifier of said segment of digital information, and to execute the cryptographic algorithm by using said identification information, said segment identifier being either an index assigned to the corresponding segment of digital information in the external memory or an

identifier present in a header of the corresponding segment of digital information.

3. The secure element according to any of claims 1 and 2, wherein said secure element stores at least one unique cryptographic key, that was generated uniquely for the secure element, to be used by the system in executing the cryptographic algorithm.

4. The secure element according to claim 3, wherein the system (140) comprises a key derivation function to generate at least one cryptographic key, to be used by the cryptographic algorithm, by using the at least one unique cryptographic key and the identification information of the digital information (Si) as inputs.

5. The secure element according to any of claims 1 to 4, wherein the cryptographic algorithm comprises an authentication algorithm (MAC) to authenticate the segment of digital information (Si) that has been loaded.

6. The secure element according to any of claims 1 to 5, wherein the cryptographic algorithm comprises a decryption algorithm to decrypt the segment of digital information (Si) that has been loaded.

7. The secure element according to claim 6, wherein the system (140) is configured to generate an initialization vector (IV), to be used by the decryption algorithm (144), by using the identification information of the segment of digital information (Si) as an input.

8. The secure element according to any of claims 1 to 7, wherein the system (140) is further configured to verify a freshness information (CtrlObj_Version) of said version table (CtrlObj) by comparing said freshness information (CtrlObj_Version) of the version table (CtrlObj) with a freshness counter (FC) stored in an internal non-volatile memory (130) of the secure element (100).

9. The secure element according to any of claims 1 to 8, wherein the system (140) is configured to load the version table (CtrlObj) in a startup procedure of the secure element (100).

10. The secure element according to any of claims 1 to 9, wherein the system (140) is configured to authenticate and decrypt the version table (CtrlObj) that has been loaded.

11. The secure element according to any of claims 1 to 10, further comprising an update module (150) configured, in case that a segment of digital information (Sj) is modified when it is processed by the secure processor (110), to control an operation of writing the modified segment of digital information (Sj') and an updated version table (CtrlObj') in the external memory (200), and incrementing the freshness counter in the internal non-volatile memory (130).

12. A system including the secure element (100) according to any of claims 1 to 11~~12~~ and an external memory (200) for storing the digital information.

13. A method for securely processing digital information by a secure element (100), said digital information being segmented and stored in a plurality of M segments in a memory (200) external to the secure element (100), including the following steps, performed by the secure element (100), of:

- loading (S54) a segment of the plurality of M segments of digital information ($[Si]_{k1}|MAC_{[Si]}$) from the external memory (200) into the secure element (100);
- determining an identification information of the segment of digital information that has been loaded by obtaining a version number (Vi) of said segment of digital information;
- pre-processing (S56-S58) said segment of digital information ($[Si]_{k_1}|MAC_i$) by executing a cryptographic algorithm using said identification information before processing said segment of digital information (Si) by a secure processor (110) of the secure element (100);
- securely processing (S59) the segment of digital information (Si) by the secure processor (110)~~-~~;

characterized in that the method further comprises

a step of loading a version table (CtrlObj) containing respective version numbers of the plurality of M segments of digital information from the external memory (200) into an internal memory (122) of the secure element (100) and

a step of reading, from the version table (CtrlObj) stored in the internal memory (122), the version number (Vi) of each segment of digital information that is loaded. ~~in the version table (CtrlObj) stored in the internal memory (122).~~

14. The method according to claim 13, further comprising a step of verifying a freshness information (CtrlObj_Version) of said version table (CtrlObj) by comparing said freshness information (CtrlObj_Version) of the version table (CtrlObj) with a freshness counter (FC) stored in an internal non-volatile memory (130) of the secure element (100).