Using the ECC Controller on AT91SAM9260/9263 and AT91SAM7SE Microcontrollers

1. Scope
The purpose of this document is to explain how to use the Error Corrected Code (ECC) Controller embedded in the AT91SAM9260/9263 and AT91SAM7SE family of ARM® Thumb®-based microcontrollers. The ECC controller performs 2-bit data error identification and single-bit correction to maintain integrity of data stored in NAND Flash and SmartMedia® devices.

2. NAND Flash Device Overview

2.1 Internal Array Architecture
The NAND Flash array is organized in a series of blocks which are divided in several pages. Data is stored either in byte (8 bits) or half-word (16 bits) format depending on the device type. Each page consists of a main area for storing data and a spare area (physically similar) typically used for data error identification and correction, wear levelling, etc...

One particularity of NAND Flash devices is that they may contain a percentage of invalid blocks in the memory array. Before delivering the chip, these blocks are identified and marked as “Invalid Blocks” in the first or second page of each block. The existence of bad blocks does not affect the good ones because each block is independent and individually isolated from the bit lines by block select transistors.

Because NAND Flash devices have a finite lifetime (approximately 100 000 write/erase cycles), additional invalid blocks may develop while being used. Storing data requires bad-block management and data error identification and correction. Refer to Section 3. "Invalid Block Management".

2.2 Basic Operation Principle
NAND Flash operations are fully controlled through a multiplexed I/O interface and additional control signals. Commands, addresses and data are transferred through the external input/output bus (8-bit or 16-bit) to the dedicated internal registers. In 16-bit devices, commands, addresses and data use the lower 8 bits (7 - 0), the upper 8 bits are only used during data-transfer cycles.

Read and program operations are performed on a per page basis whereas erase operations are performed on a block basis. To read or write from NAND Flash, a command sequence is issued to select a block and a page. After this selection, the entire page can be read or written.

The command sequence normally consists of a Command Latch Cycle, an Address Latch Cycle and a Data Cycle — either read or write.
The waveforms shown in Figure 2-1 depict the successive accesses: Command Latch, Address Latch and Data Output. Notice that no command can be sent to the NAND Flash during t_R due to its busy-state period.

**Figure 2-1.** Page READ Operation

Please refer to the NAND Flash manufacturer’s datasheet for command sets and full operation description.

3. **Invalid Block Management**

3.1 **Invalid Block Definition**

As mentioned in Section 2.1 "Internal Array Architecture", NAND flash devices contain a certain percentage of invalid blocks at the end of the production process. Invalid blocks are defined as blocks that contain one or more invalid bits.

3.2 **Spare Area Assignment**

The invalid block status byte location and the ECC locations within the spare area depend on the device type (small/large-page devices and 8/16-bit devices).

The widely used spare area assignment defined by SAMSUNG is illustrated in Figure 3-1, Figure 3-2, Figure 3-3 and Figure 3-4.
Figure 3-1.  Small Page 8-bit Device Organization

<table>
<thead>
<tr>
<th>LSN0</th>
<th>LSN1</th>
<th>LSN2</th>
<th>Reserved</th>
<th>Reserved</th>
<th>BI</th>
<th>ECC0</th>
<th>ECC1</th>
<th>ECC2</th>
<th>S-ECC0</th>
<th>S-ECC1</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
</tr>
</thead>
</table>

Figure 3-2.  Small Page 16-bit Device Organization

<table>
<thead>
<tr>
<th>LSN0</th>
<th>LSN1</th>
<th>LSN2</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
<th>ECCa</th>
<th>ECCb</th>
<th>ECCc</th>
<th>S-ECCa</th>
<th>S-ECCb</th>
<th>B1</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB</td>
<td>MSB</td>
<td>LSB</td>
<td>MSB</td>
<td>LSB</td>
<td>MSB</td>
<td>LSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>LSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
</tr>
</tbody>
</table>

Figure 3-3.  Large Page 8-bit Device Organization

<table>
<thead>
<tr>
<th>BI</th>
<th>BI</th>
<th>LSN0</th>
<th>LSN1</th>
<th>LSN2</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
<th>ECC0</th>
<th>ECC1</th>
<th>ECC2</th>
<th>S-ECC0</th>
<th>S-ECC1</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
</tr>
</thead>
</table>

Figure 3-4.  Large Page 16-bit Device Organization

<table>
<thead>
<tr>
<th>BI</th>
<th>BI</th>
<th>LSN0</th>
<th>LSN1</th>
<th>LSN2</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
<th>ECC0</th>
<th>ECC1</th>
<th>ECC2</th>
<th>S-ECC0</th>
<th>S-ECC1</th>
<th>Reserved</th>
<th>Reserved</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB</td>
<td>MSB</td>
<td>LSB</td>
<td>MSB</td>
<td>LSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
<td>MSB</td>
</tr>
</tbody>
</table>

Abbreviations as used in Figure 3-1 through Figure 3-4 above

- **BI**: Invalid block information
- **ECC**: ECC code for Cell Array data
- **S-ECC**: ECC code for LSN data
- **HW**: Half Word
- **LSN**: Logical sector number
3.3 Invalid Block Identification

Before shipping, every NAND Flash device is tested with specific test patterns under different voltage and temperature conditions in order to identify memory locations containing errors. When errors are detected, the block to which the invalid memory location belongs is marked as an “Invalid Block”.

All device locations are erased (FFh for 8-bit devices, FFFFh for 16-bit devices) except locations where the invalid block information is written.

As illustrated above in Figure 3-1, Figure 3-2, Figure 3-3, and Figure 3-4, the bad block Information is located in the first byte (8-bit devices) or first half word (16-bit devices) in the spare area for Large Page devices and in the sixth byte (8-bit devices) or sixth half word (16-bit devices) in the spare area of Small Page devices.

Manufacturers make sure that every invalid block has non-FFh (8-bit devices) or non-FFFFh (16-bit devices) data in the bad block information location.

Since invalid block information (located in the spare area) written by the manufacturer is not write/erase protected, it can be lost and will be almost impossible to recover. In order to prevent loosing this information, it is highly recommended to proceed to a block status mapping before any write or erase operation.

The flow chart below describes how this can be done by software.

**Figure 3-5.** Bad Block Identification Flow Chart

**Important Note:** Any intentional erasure of the original invalid block information is prohibited.
4. Error Detection and Correction

NAND Flash/SmartMedia devices contain by default invalid blocks which have one or more invalid bits. Over the NAND Flash/SmartMedia lifetime, additional invalid blocks may occur which can be detected/corrected by ECC code. To ensure data read/write integrity, system error checking and correction (ECC) algorithms should be implemented. The AT91SAM9260/9263 and AT91SAM7SE microcontrollers provide ECC hardware support. The embedded ECC controller is capable of single-bit error correction and 2-bit error detection per page (528/1056/2112/4224). When NAND Flash/SmartMedia have more than 2 bits of errors, the data cannot be corrected.

4.1 ECC Calculation Algorithm

For Single-bit Error Correction and Double bit Error Detection (SEC-DED) hsiao code is used. 32-bit ECC is generated in order to perform one bit correction per 512/1024/2048/4096 8- or 16-bit words. Of the 32 ECC bits, 26 bits are for line parity and 6 bits are for column parity. They are generated according to the schemes shown in Figure 4-1 and Figure 4-2.

![Figure 4-1. Parity Generation for 512/1024/2048/4096 8-bit Words](image)

Page size = 512  Px = 2048  
Page size = 1024  Px = 4096  
Page size = 2048  Px = 8192  
Page size = 4096  Px = 16384

\[ P_1 = \text{bit7} + \text{bit5} + \text{bit3} + \text{bit1} \]  
\[ P_2 = \text{bit7} + \text{bit6} + \text{bit3} + \text{bit2} \]  
\[ P_4 = \text{bit7} + \text{bit6} + \text{bit5} + \text{bit4} \]  
\[ P_1' = \text{bit6} + \text{bit4} + \text{bit2} + \text{bit0} \]  
\[ P_2' = \text{bit5} + \text{bit4} + \text{bit1} + \text{bit0} \]  
\[ P_4' = \text{bit7} + \text{bit6} + \text{bit5} + \text{bit4} \]
Figure 4-2. Parity Generation for 512/1024/2048/4096 16-bit Words

Page size = 512, P#: 2048
Page size = 1024, P#: 4096
Page size = 2048, P#: 8192
Page size = 4096, P#: 16384
4.2 ECC Controller Preliminary Requirements

In order to calculate ECC properly during write/read and read processes, the following constraints must be respected:

- at least 1 Hold time must be programmed in the RWHOLD field of the SMC_CSRx register (only AT91SAM7SE family is concerned)
- read/write sequence must start at a page boundary
- read/write accesses must be done through the whole main area since ECC is calculated on the main area data
- data accesses must be performed chronologically through the main area
- the appropriate page size must be programmed in the PAGESIZE field of the ECC_MR register

4.3 ECC Controller Functional Description

4.3.1 Page Write Sequence

The ECC controller is automatically reset as soon as the first write command (80h) is performed to the NAND Flash or the SmartMedia device.

The ECC calculation starts only once the required address cycles (the number of address cycles depends on the device type) is performed to NAND Flash or the SmartMedia device.

The ECC is refreshed at each write access of the page until the last byte or half word of the main area is written.

Once the whole main area has been written, the final ECC result is available in the ECC Parity Register (ECC_PR) and ECC NParity Register (ECC_NPR) until a new start condition occurs. It is up to the software application to write the Parity ECC and NParity ECC in the appropriate locations of the device spare area.

Please note that apart from data accesses (ALE = CLE = 0), the ECC controller ignores any other command which is performed to the NAND Flash or the SmartMedia device.

Figure 4-3 below illustrates a full page write sequence with ECC calculation.

Figure 4-3. ECC Calculation During Page Write Sequence without Random Write Spare Area
### 4.3.2 Page Read Sequence

The ECC controller is automatically reset as soon as the first read command (00h) is performed to the NAND Flash or the SmartMedia device.

The ECC calculation starts only once the required address cycles (the number of address cycles depends on the device type) and the second read command (30h) is performed to the NAND Flash or the SmartMedia device.

The ECC is refreshed at each read access of the page until the last byte or half word of the main area is read.

Once the whole main area has been read, the next four data read accesses must be performed to the spare area locations where ECC has been previously stored by the software application. If this condition is not respected, the ECC controller will not be able to check data integrity.

Since Parity ECC and NParity has been previously stored in locations of the spare area which are not contiguous to the main area, it is useful to perform a random read command sequence before performing the four ECC data accesses.

Please note that apart from data accesses (ALE = CLE = 0), the ECC controller ignores any other command which is performed to the NAND Flash or the SmartMedia device.

The ECC controller performs error detection automatically by applying an XOR operation between the calculated ECC and the ECC code stored in the spare area.

In order to determine if an error has been detected by the ECC controller, the software application must check the MULERR, ECCERR and RECERR fields in the ECC Status Register (ECC_SR).

#### 4.3.2.1 No Error

MULERR, ECCERR and RECERR fields in the ECC Status Register (ECC_SR) are all cleared.

XOR between the calculated ECC computation and the ECC code stored in the spare area is equal to 0.

#### 4.3.2.2 Recoverable Error

Only the RECERR field in the ECC Status register (ECC_SR) is set. The corrupted word offset in the read page is defined by the WORDADDR field in the ECC Parity Register (ECC_PR). The corrupted bit position in the concerned word is defined in the BITADDR field in the ECC Parity Register (ECC_PR).
4.3.2.3 **ECC Error**

The ECCERR field in the ECC Status Register (ECC_SR) is set. An error has been detected in the ECC code stored in the Spare area of the device. The position of the corrupted bit can be found by applying an XOR operation between the ECC Parity and the ECC NParity codes previously stored in the spare area of the device.

4.3.2.4 **Non Recoverable Error**

The MULERR field in the ECC Status Register (ECC_SR) is set. Several errors have been detected in the data stored in the device. The block to which this page belongs should be declared as invalid.

**Figure 4-5** below illustrates a full page read sequence with ECC error detection.

**Figure 4-5.** ECC Error Detection During Page Read Sequence with Random Read Spare Area

**Figure 4-6.** ECC Error Detection During Page Read Sequence without Random Read Spare Area
5. High-Level File System Software Compatibility

High-level software drivers for managing file systems in NAND Flash devices are available from different sources. These drivers provide support for wear leveling, bad block management, ECC etc...

File Systems available on the market usually manage ECC as:

- 3 bytes ECC for 256 bytes of data per page
- 3 bytes ECC for 512 bytes of data per page

The AT91SAM ECC controller manages ECC as:

- 4 bytes ECC for 512/1024/2048/4096 bytes of data per page

Since the ECC offset in the spare area and the number of ECC per page is not yet normalized, it is highly recommended to manage ECC by software when using a high-level file system.

6. Software Example

A software example managing Bad Block Information and ECC error detection for Large Page Devices can be downloaded from the Atmel web site via the following link:

http://atmel.com/dyn/resources/prod_documents/an-nand_flash_sam7se_software_example.zip
7. Revision History

<table>
<thead>
<tr>
<th>Doc. Rev</th>
<th>Comments</th>
<th>Change Request Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>6320A</td>
<td>First issue</td>
<td></td>
</tr>
<tr>
<td>6320B</td>
<td><strong>Figure 4-3 and Figure 4-4 updated, Section 3.2 &quot;Spare Area Assignment&quot;, updated sentence referring to figures.</strong></td>
<td>4440</td>
</tr>
</tbody>
</table>