Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT

Table 1: Cross Reference of Applicable Products

Product Name	Manufacturer Part Number	SMD #	Device Type	Internal PIC Number
UT700 LEON 3FT	UT700	5962-13238	Memory Controller	WQ03

1.0 Overview

The UT700 LEON 3FT consists of multiple on-chip modules and a fault tolerance memory controller that supports three types of memory devices, PROM, SRAM, and SDRAM respectively. These three types of memory devices are mapped into its specific memory regions defined by the UT700 LEON 3FT architecture. Please see **Figure 1** for the types of memories mapping onto the memory map.
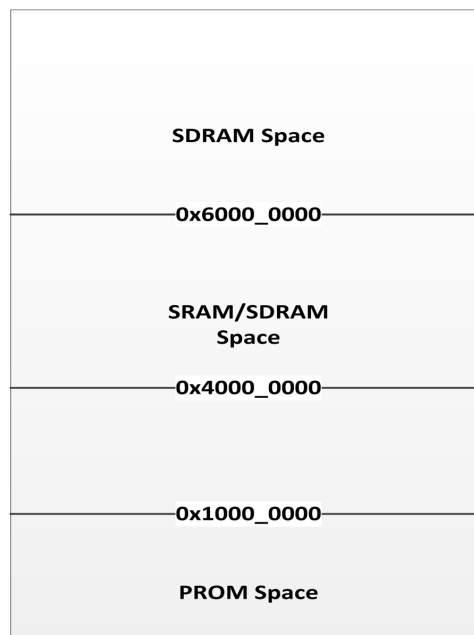

Figure 1: Volatile and Non-Volatile Memory Map

Note: For maximum PROM, SRAM, and SDRAM support sizes, please refer to the UT700 LEON 3FT Functional Manual.

The memory controller employs two types of fault-tolerance techniques. Each type of the fault-tolerance techniques can be applied to the specific memory types using the GPIOs' setting and the memory configuration registers (MCFGX) provided by the memory controller. The memory controller also supports memory devices with data bus width of 8, 16 and 32-bits.

This Application Note (AN) will provide concise ways to enable EDAC for the different types of memory devices mapped to the UT700 LEON fault tolerance memory controller both in hardware and software.

CAES PIONEERING ADVANCED ELECTRONICS

4350 Centennial Blvd. | Colorado Springs CO 80907 USA | 1-800-645-8862 | cobhamaes.com

Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT

2.0 UT700 LEON 3FT Memory Controller

The fault tolerance memory controller provides both the BCH EDAC and Reed-Solomon EDAC features to improve the data integrity of the memory devices.

The BCH EDAC algorithm generates a 7-bit checksum for every 32-bit word and provides a single-bit-error correction and a dual-bit-error detection to that data word. BCH EDAC can be used on PROM, SRAM, and SDRAM memory by setting the corresponding EDAC enable bits in the MCFGX and the respective GPIO settings. For more details about the BCH EDAC, please refer to the UT700 LEON 3FT Functional Manual **3.10.1** BCH EDAC.

The Reed-Solomon EDAC algorithm provides a block error correction that generates a 16-bit checksum for every 32-bit data word and is capable of correcting any two adjacent 4-bit-nibble errors. The Reed-Solomon EDAC only supports SDRAM. For more details about the Reed-Solomon EDAC, please refer to the UT700 LEON 3FT Functional Manual **3.10.2** Reed-Solomon EDAC.

Table 1 provides a breakdown of all the memory device configurations that are supported by the BCH and the Reed-Solomon EDAC.

Table 2: Types of Memory Device Configurations

Memory Type	ECH EDAC	Remarks	Reed-Solomon EDAC	Remarks
8-bit PROM	Supported	Configuration using GPIO[2:0] and mkprom2	Not Supported	N.A
32-bit PROM	Supported		Not Supported	
8-bit SRAM	Supported	Configuration using MCFGX	Not Supported	
32-bit SRAM	Supported		Not Supported	
32-bit SDRAM	Supported	Configuration using MCFGX	Supported	Configuration using MCFGX

3.0 Memory Configuration

The following sections will show how each memory device is configured to function with the UT700 LEON 3FT memory controller with EDAC enabled. Hardware block diagrams will be referenced that show how the memory devices are connected to the UT700 LEON 3FT Memory Controller. Software examples will be provided to show how to set the MCFGX to enable the memory devices in different operation modes as shown in **Table 2**.

3.1 Programmable Read-Only Memory, PROM

At Power-On-Reset (POR), the UT700 goes through a series of internal initialization steps followed by the reading of the PROM for the next flow control instruction required by the system to function.

Therefore, using the MCFGX to configure the PROM bus width will not work; the UT700 needs to know how to read the instruction from the PROM to perform the configuration.

The correct way to configure the PROM data bus width and enable BCH EDAC is to drive the GPIO [2:0] to its appropriate states as shown in **Table 3**.

Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT

Table 3: PROM Data Bus and ECH EDAC Configuration

PROM Data Bus Width	GPIO[2]	GPIO[1]	GPIO[0]	Remarks
8-bit/No ECH EDAC	0	0	0	--
8-bit/With ECH EDAC	1	0	0	mkprom2
32-bit/No ECH EDAC	0	1	0	--
32-bit/With ECH EDAC	1	1	0	mkprom2
Not Used	X	1	1	Illegal state

3.1.0 PROM Configuration, BCH EDAC

Please refer to Memory Controller with EDAC chapter block diagrams (**Figure 3.2 and Figure 3.4**) for 8-bit and 32-bit Memory Interface Examples in the UT700 LEON 3FT Functional Manual. In addition to these block diagrams, the PROM also requires that the GPIO [2:0] be set to its appropriate states based on the data bus width of the PROM devices.

The GPIO pins setting are shown in **Table 3**. To set the GPIO to logic '1', a pull-up to 3.3V with a 1K and higher resistance resistor is needed. To set the GPIO to logic '0', a pull-down with a 10K resistor is needed.

There is no software setting requirement to configure the PROM memory interface. However, to support ECH EDAC, a special utility known as the **MKPROM2** is required to generate the 7-bit checksums. These checksum bits will be stored in the upper 20% region of the PROM, a unique feature offered by the memory controller to reduce the number of chip counts and reduce the size of the PCB.

8-bit PROM

In 8-bit mode, the mkprom2 utility offers two ways to generate the BCH EDAC checksums: a 4:1 space ratio or a 3:1 space ratio BCH checksum output file (they are the same except their space allocation). Appended are the command lines and parameters required to generate BCH checksums as follows:

```
mprom2 –romwidth 8 –romsize 2048 –romcs 1 Appl –mv8 –bch8  // 4:1 space ratio
mprom2 –romwidth 8 –romsize 2048 –romcs 1 Appl –mv8 –bch8q // 3:1 space ratio
```

The 4:1 space ratio BCH checksum requires a 20% of the upper PROM space while the 3:1 space ratio BCH checksum requires a 25% of the upper PROM space.

Once the checksum file is generated, you can use **GRMON (flash load <file>)** to program the application file and the checksum file to the PROM device. The code and the checksum bits are organized in the PROM as shown in **Figure 2** and the coloring shows its relationship.
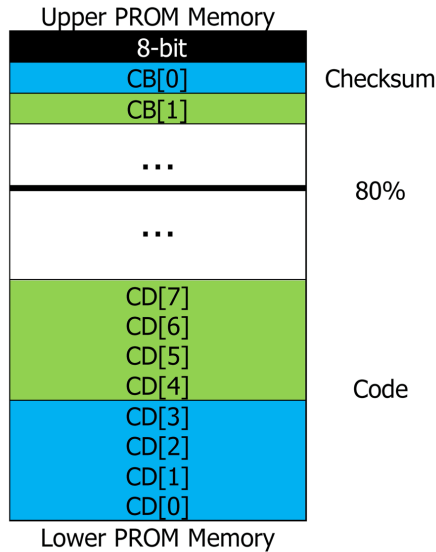
Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT

Upper PROM Memory

8-bit
CB[0]
CB[1]
...
...
CD[7]
CD[6]
CD[5]
CD[4]
CD[3]
CD[2]
CD[1]
CD[0]

Checksum

80%

Code

Lower PROM Memory

Figure 2: 8-bit PROM Code and BCH Checksum organization

32-bit PROM

MKPROM2 and GRMON do not support 32-bit PROM mode. For more information on 32-bit PROM mode, please contact your area FAE for more information.

In 32-bit mode, the code and BCH checksum bits are organized as shown in **Figure 3.**

8-bit	8-bit	8-bit	8-bit	8-bit/16-bit
CD[0]	CD[1]	CD[2]	CD[3]	CB[0]
CD[4]	CD[5]	CD[6]	CD[7]	CB[1]
CD[8]	CD[9]	CD[10]	CD[11]	CB[2]
CD[12]	CD[13]	CD[14]	CD[15]	CB[3]
...				

Figure 3: 32-bit PROM Code and BCH Checksum organization

Legend:

CB: checksum bits
CD: code

Please refer to the **GRMON** and **MKPROM2** documents for more information. The **GRMON** and **MKPROM2** utilities and it documents can be found in cobhamaes.com.

3.2 Static Random-Access Memory, SRAM

After POR, device initialization loads from the of PROM device. During this state, the stack and heap memory (SRAM/SDRAM) are not yet available to the system. This means that the application in the PROM must not access the volatile memory until it is initialized. Since this is the order of the system flow control, configuring and initializing the volatile memories via the MCFGX is possible.

The SRAM configuration and initialization are shown in the following section.

3.2.0 SRAM Configuration, BCH EDAC

Please reference to **3.1.0** for hardware connection block diagrams. SRAM configuration can be achieved via the MCFGX registers.

In **Table 4**, these are the codes needed to configure the SRAM for its respective operation as follows:

Table 4: SRAM Data Bus and ECH EDAC Configuration

Memory Configuration Register 2

Bit#	Name	8-bit	32-bit	Remarks
14	DE	0x0/0x1	0x0/0x1	SDRAM controller disable automatically enable SRAM 0: SDRAM controller disabled 1: SDRAM controller enabled
13	SI	x/0x0	x/0x0	SRAM enable
12-9	SZ	size	size	Size of each SRAM bank as $8 * 2^{SZ}$ KB 0000: 8KB 0001: 16KB 0010: 32KB ... 1111: 256MB
6	RM	0x1	0x1	--
5-4	SD	0x0	0x2	Data width of the SRAM area
3-2	SW	0x0	0x0	Number of wait states during SRAM write cycles
1-0	SR	0x0	0x0	Number of wait states during SRAM read cycles

Memory Configuration Register 3

Bit#	Name	8-bit	32-bit	Remarks
9	SE	0x1	0x1	Enable EDAC checking of the SDRAM or SRAM area 0: EDAC checking of the RAM area disabled 1: EDAC checking of the RAM area enabled

CAES PIONEERING ADVANCED ELECTRONICS   4350 Centennial Blvd. | Colorado Springs CO 80907 USA | 1-800-645-8862 | cobhamaes.com

Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT

Extracting the data from the **Table 4** for 8-bit mode, the C codes are as appended.

```
FRMCTRL.MCFG2.B.DE = 0x1;      // 0: SDRAM controller enabled
FRMCTRL.MCFG2.B.SI = 0x0;      // 0: SRAM enabled
FRMCTRL.MCFG2.B.SZ = 0xF;      // 256MB
FRMCTRL.MCFG2.B.RM = 0x1;      // Enable read-modify-write
FRMCTRL.MCFG2.B.SD = 0x0       // 8-bit
FRMCTRL.MCFG2.B.SW = 0x0       // 0 wait state (write) change this for different SRAM
FRMCTRL.MCFG2.B.SR = 0x0       // 0 wait state (read) change this for different SRAM

FRMCTRL.MCFG3.B.SE = 0x1       // 1 wait state (read) change this for different SRAM
```

In 8-bit mode, the checksum bits are stored similar to the PROM at the upper 20% of the SRAM. The difference is the checksum bits are created on the fly. For 32-bit mode, another 8-bit SRAM (size=20% of the 32-bit SRAM) is required for storing the checksum bits generated by the BCH engine and retrieving for error detection and correction.

The code and its respective checksum bits for 8-bit and 32-bit SRAMs are organized in the same format as the PROM device as shown in **Figure 2** and **Figure 3** respectively.

Please see Appendix A for the fault tolerance memory controller header file.

3.3 Synchronous Dynamic Random-Access Memory, SDRAM

Similar to **3.2**, the SDRAM also needs to be initialized before being used. However, for SDRAM, the memory controller supports both BCH and Reed-Solomon EDAC.

The SDRAM configuration and initialization are shown in the following section.

3.3.0 SDRAM Configuration, BCH and Reed-Solomon EDAC

Please reference to **3.1.0** for hardware connection block diagrams. SDRAM configuration can be achieved via the MCFGX registers.

In **Table 5**, the 32-bit data width SDRAM supports both BCH and Reed-Solomon EDAC. These are the codes needed to configure the SDRAM for its respective operation mode as follows:

Table 5: SRAM Data Bus and ECH EDAC Configuration

Memory Configuration Register 2

Bit#	Name	BCH	Reed-Solomon	Remarks
14	DE	0x0/0x1	0x0/0x1	SDRAM controller disable automatically enable SRAM 0: SDRAM controller disabled 1: SDRAM controller enabled
13	SI	x/0x0,0x1	x/0x0,0x1	0:0x6000_0000, 1:0x4000_0000 (Memory Map address)

Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT

Memory Configuration Register 3

Bit#	Name	BCH	Reed-Solomon	Remarks
28	RSE	0x0	0x1	Reed-Solomon EDAC 0: BCH EDAC available for SDRAM 1: Reed-Solomon EDAC available for SDRAM
9	SE	0x1	0x1	Enable EDAC checking of the SDRAM or SRAM area 0: EDAC checking of the RAM area disabled 1: EDAC checking of the RAM area enabled

Extracting the data from the **Table 5** for Reed-Solomon, the C codes are as appended.

```
FRMCTRL.MCFG2.B.DE  = 0x1;    // 0: SDRAM controller enabled
FRMCTRL.MCFG2.B.SI  = 0x0;    // 0: SDRAM mapped at 0x6000_0000

FRMCTRL.MCFG3.B.RSE = 0x1     // 0: BCH EDAC, 1: Reed-Solomon EDAC
FRMCTRL.MCFG3.B.SE  = 0x1     // 0: BCH EDAC, 1: Reed-Solomon EDAC
```

The C codes above are required to enable EDAC for SDRAM, BCH or Reed-Solomon. Additional configurations are needed to set the other timing requirements for the type of SDRAM used. These settings can be found in MCFG2 and MCFG3.

The code and its respective checksum bits are organized in the same format as the PROM device as shown in **Figure 3** (For Reed-Solomon, CB is 16 bit wide).

Please see Appendix A for the fault tolerance memory controller header file.

4.0 Summary and Conclusion

For more information about our UT700 LEON 3FT/SPARC™ V8 Microprocessor and other products please visit our website, cobhamaes.com.

CAES PIONEERING ADVANCED ELECTRONICS

4350 Centennial Blvd. | Colorado Springs CO 80907 USA | 1-800-645-8862 | cobhamaes.com

Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT

Appendix A: Header File

```c
/*********************************************************************\
* MODULE: FTMCTRL Memory Controller Registers                        *
\*********************************************************************/
#define FRMCTRL_ADDR    0x80000000
struct FRMCTRL_TAG {
        union {
                vuint32_t R;                    // Figure 3.5 Memory Configuration
Register 1
                struct {
                    vuint32_t RES31      :1;    // Reserved
                    vuint32_t PB         :1;    // PROM area bus enable
                    vuint32_t AB         :1;    // Asynchronous bus ready
                    vuint32_t IBW        :2;    // I/O data bus width
                    vuint32_t IB         :1;    // I/O area bus ready enable
                    vuint32_t BE         :1;    // Bus error enable
                    vuint32_t RES24      :1;    // Reserved
                    vuint32_t IW         :4;    // Number of wait states during I/O
accesses
                    vuint32_t IE         :1;    // I/O enable
                    vuint32_t RES18      :1;    // Reserved
                    vuint32_t PZ         :4;    // PROM size is fixed at 256MB (1111)
                    vuint32_t RES1312    :2;    // Reserved
                    vuint32_t PE         :1;    // PROM write enable
                    vuint32_t RES10      :1;    // Reserved
                    vuint32_t PD         :2;    // Data width of the PROM area
                    vuint32_t PW         :4;    // Number of wait states during PROM
write cycles
                    vuint32_t PR         :4;    // Number of wait states during PROM
read cycles
                } B;
        } MCFG1;
        union {
                vuint32_t R;                    // Figure 3.6 Memory Configuration
Register 2
                struct {
                    vuint32_t DR         :1;    // SDRAM refresh
                    vuint32_t DP         :1;    // SDRAM TRP parameter
                    vuint32_t DF         :3;    // SDRAM TRFC parameter
                    vuint32_t DC         :1;    // SDRAM CAS parameter
                    vuint32_t DZ         :3;    // Bank size for SDRAM chip selects
defined as 4MB*2DZ
                    vuint32_t DS         :2;    // SDRAM column size is 2048 when bits
[25:23] = 111
                    vuint32_t DD         :2;    // SDRAM command
                    vuint32_t BW         :1;    // Memory controller data bus width
                    vuint32_t PB         :1;    // SDRAM Page Burst
```

```
            vuint32_t RES1615    :2;     // Reserved
            vuint32_t DE         :1;     // SDRAM enable
            vuint32_t SI         :1;     // SRAM disable
            vuint32_t SZ         :4;     // Size of each SRAM bank as 8*2^SZ KB
```

```
            vuint32_t RES8       :1;     // Number of wait states during PROM
write cycles
            vuint32_t SB         :1;     // Enable read-modify-write cycles on
sub-word writes
            vuint32_t RM         :1;     // Number of wait states during PROM
read cycles
            vuint32_t SD         :2;     // Data width of the SRAM area
            vuint32_t SW         :2;     // Number of wait states during SRAM
write cycles
            vuint32_t SR         :2;     // Number of wait states during SRAM
read cycles
        } B;
    } MCFG2;
    union {
        vuint32_t R;                     // Figure 3.7 Memory Configuration
Register 3
        struct {
          vuint32_t RES3129    :3;     // Reserved
          vuint32_t RSE        :1;     // 1:Reed-Solomon 0:EDAC
          vuint32_t ME         :1;     // Memory EDAC
          vuint32_t RLDVAL     :15;    // SDRAM refresh counter reload value
          vuint32_t WB         :1;     // EDAC diagnostic write bypass
          vuint32_t RB         :1;     // EDAC diagnostic read bypass
          vuint32_t SE         :1;     // Enable EDAC checking of the SDRAM or
SRAM area
          vuint32_t PE         :1;     // Enable EDAC checking of the PROM area
(GPIO[2])
          vuint32_t TCB        :8;     // Test check bits
        } B;
    } MCFG3;
    union {
        vuint32_t R;                     // Figure 3.8 Memory Configuration
Register 4
        struct {
          vuint32_t RES3117    :15;    // Reserved
          vuint32_t WB         :1;     // Reed Solomon EDAC diagnostics write
bypass
          vuint32_t RTC        :16;    // Reed Solomon test check bits
        } B;
    } MCFG4;
};
```

Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT

Appendix B: Acronyms

BCH: BCH codes were invented in 1959 by French mathematician Alexis Hocquenghem, and independently in 1960 by Raj Bose and D. K. Ray-Chaudhuri

EDAC: Error Detection and Correction

MCFX: Memory Configuration Register (1, 2, 3, or 4)

PROM: Programmable Read-only Memory

SRAM: Static Random-Access memory

SDRAM: Synchronous Dynamic Random-Access Memory

Revision History

Date	Rev. #	Author	Change Description
06/15/2016	1.0.0	MTS	Initial Release

The following United States (U.S.) Department of Commerce statement shall be applicable if these commodities, technology, or software are

Enable BCH and Reed-Solomon EDAC UT700 LEON 3FT