UTMC APPLICATION NOTE

UT80CRH196KD Error Detection and Correction Functionality

The UT80CRH196KD provides flow through Error Detection and Correction (EDAC) for external memory accesses. When enabled, the EDAC will produce check bits for any external memory write operation. Additionally, external memory reads will use the externally stored check bits to determine if any errors have occurred. The UT80CRH196KD will detect both single and double bit errors, and correct single bit errors in the data. Furthermore, the UT80CRH196KD can perform memory scrubs by reading the external memory and writing the corrected data back into memory.

This application note discusses how the UT80CRH196KD encodes and decodes data, creates an Error Syndrome Code Word (ESCW), and uses the ESCW to detect and correct corrupted data. Further, you will learn how the UT80CRH196KD reacts to single, double, and triple bit errors.

1.0 Generating Check Bits (Data Encoding)

The EDAC engine will create six check bits to encode the information word. Each check bit represents the parity of a specific subset of data bits. The check bit generation scheme for encoding data is described by the matrix in Table 1. Every check bit is created by calculating the even parity (XORing) the data bits specified by the X's in their respective rows. UTMC suggests that you tie the EDAC check bits high or low to prevent them from floating to a voltage switching level during long read cycles. If these inputs float, you run the risk of partially turning on both the P- and N-channel MOSFETS for the respective inputs.

TABLE 1. Check Bit Generation

	16 Bit Data Word															
Check Bits	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
CB0			X			X	X	X				X	X		X	X
CB1		X			X			X		X	X		X	X		X
CB2	X			X			X		X		X	X		X	X	
CB3				X	X	X			X	X				X	X	X
CB4	X	X	X						X	X	X	X	X			
CB5	X	X	X	X	X	X	X	X								

1.1 Check Bit Generation for 8-bit Buswidth Memory Cycles

The UT80CRH196KD can perform EDAC on both 8-bit and 16-bit wide bus cycles. While using 8-bit wide bus cycles, the UT80CRH196KD will always force CB5 to output a '0' logic value for write operations, and will internally mask CB5 to a '0' logic value during read operations. Additionally, the upper data byte (bits 15 through 8), used in the parity generation for check bits CB0 to CB4, will be zeroed.

1.2 Check Bit Generation for 16-bit Buswidth Memory Cycles

The UT80CRH198KD will also perform EDAC operations on 16-bit wide bus cycles. The check bit generation is strait forward, and operates according to the check bit generation matrix in Table 1. Furthermore, do not use single byte read/write operations when using EDAC on 16-bit wide bus cycles because both the high and low bytes of data must be known in order to produce valid EDAC check bits. Single byte operations will flag errors and/or corrupt data when read back.

2.0 Error Correction

The EDAC engine will create an Error Syndrome Code Word (ESCW) every time it reads data from memory. The Error Syndrome Code Word is derived from the bitwise exclusive-nor (XNOR) of the check bits read from memory and the check bits generated from the received data. The ESCW is then used to locate and correct all single bit data errors prior to using them internally. The error syndrome decoding scheme is shown in Table 2. Every column in this table represents a unique Error Syndrome Code Word. Once the code word has been created, a bitwise comparison between each column and that code word is performed. If the ESCW matches a column, then the data bit (or check bit) represented by that column is in error. If the Error Syndrome Code Word is 3Fh (11 1111 b) then no errors have occurred in the received data. Additionally, any other Error Syndrome Code Words that are not represented in Table 2, are flagged as Double Bit Errors.

TABLE 2. Error Determination

Error Syndrome Code	Error Syndrome Code Words for Information Bits Read-In																						
	Error Syndrome Code Words for Data Bits																Error Syndrome Code Words for Check Bits						ESCW for No Errors
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	5	4	3	2	1	0	
ES0	H	H	L	H	H	L	L	L	H	H	H	L	L	H	L	L	H	H	H	H	H	L	H
ES1	H	L	H	H	L	H	H	L	H	L	L	H	L	L	H	L	H	H	H	H	L	H	H
ES2	L	H	H	L	H	H	L	H	L	H	L	L	H	L	L	H	H	H	H	L	H	H	H
ES3	H	H	H	L	L	L	H	H	L	L	H	H	H	L	L	L	H	H	L	H	H	H	H
ES4	L	L	L	H	H	H	H	H	L	L	L	L	L	H	H	H	H	L	H	H	H	H	H
ES5	L	L	L	L	L	L	L	L	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H

L = Logic Low
H = Logic High

3.0 16-Bit Bus Cycle Error Detection and Correction Examples

3.1 Data Write in a Sixteen Bit Bus Cycle

If the UT80CRH196KD wants to write the value 9148h (1001 0001 0100 1000 b) to external memory, then the generated check bits will be 32h (11 0010 b). Table 3 shows how the check bits are generated for the data value 9148h.

TABLE 3. Check Bit Generation for 9148H

		16 Bit Data Word															
		Bit 15	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Check Bits		**1**	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
CB0	0			0			0	0	1				0	1		0	0
CB1	1		0			0			1		1	0		1	0		0
CB2	0	1			1			0		0		0	0		0	0	
CB3	0				1	0	0			0	1				0	0	0
CB4	1	1	0	0						0	1	0	0	1			
CB5	1	1	0	0	1	0	0	0	1								

3.2 Single Data Bit Read Error in a Sixteen Bit Bus Cycle

Now, assume that the UT80CRH196KD reads the same location written to memory in Section 3.1. In other words, we want to read back the value 9148h from memory. However, the external memory that holds this value took an SEU hit before we could retrieve the data. As a result, data bit twelve in the memory cell was changed from a 1 to a 0. Consequently, the data read from the external memory location is 8148h (1000 0001 0100 1000 b). This value will be loaded into the EDAC engine where a new set of check bits will be generated. The generated check bit value for the data value of 8148h is 1Eh (01 1110 b). Refer to Table 4 to see the check bit generation matrix for the data value 8148h.

TABLE 4. Check Bit Generation for 8148H

		16 Bit Data Word															
		Bit 15	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Check Bits		**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
CB0	0			0			0	0	1				0	1		0	0
CB1	1		0			0			1		1	0		1	0		0
CB2	1	1			0			0		0		0	0		0	0	
CB3	1				0	0	0			0	1				0	0	0
CB4	1	1	0	0						0	1	0	0	1			
CB5	0	1	0	0	0	0	0	0	1								

Subsequently, the EDAC engine will find the bitwise exclusive-nor (XNOR) of the read-in check bits with the newly generated check bits. This will give an Error Syndrome Code Word of 13h (01 0011 b) for the data value 8148h. The Error Syndrome Code Word generation is shown in Table 5.

TABLE 5. ESC Generation

	XNOR of Check Bits					
Bit Locations	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Read-In CBs	1	1	0	0	1	0
Generated CBs	0	1	1	1	1	0
ESCW	0	1	0	0	1	1

Next, the EDAC engine will determine which bit is in error. The error determination is performed by comparing the Error Syndrome Code Word to each column in Table 2. As the ESCW is compared to each column, a Data Correction Word (DCW) is created. Every 0 in the DCW means that the ESCW did not match the column representing that bit in Table 2. For this example, the DCW is 1000h (0001 0000 0000 0000 b). This indicates a single bit error occurred at bit twelve of the data word. Since this is a single bit error, the EDAC engine can correct the error by performing a bitwise exclusive-or (XOR) operation on the data word read in, and the Data Correction Word. After taking the bitwise XOR of 8148h and 1000h, you will see that the original data word of 9148h is obtained. Table 6 shows the correcting process for the corrupted data word 8148h.

TABLE 6. Correcting Data

	XOR of Data and Data Correction Word (DCW)															
Bit Locations	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Read-In Data	1	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0
DCW	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Corrected Data	1	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0

3.3 Single Check Bit Read Error in a Sixteen Bit Bus Cycle

This time, let us assume that the UT80CRH196KD attempts to read a data word from external memory like it did in Section 3.2; but this time, the corrupted information bit was a check bit instead of a data bit. We will say that CB3 flipped from a logic 0 to a logic 1. Therefore, the EDAC engine will receive 9148h as the data word, and 3Ah (11 1010 b) for the check-bit string. Once this EDAC engine receives the information, it will calculate a new string of check bits for the received data word (9148h). This generated check bit string is 32h (11 0010 b), and is shown in Table 3. Now the Error Syndrome Code Word will be created as shown in Table 7.

TABLE 7. ESC Generation

	XNOR of Check Bits					
Bit Locations	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Generated CBs	1	1	0	0	1	0
Read-In CBs	1	1	1	0	1	0
ESCW	1	1	0	1	1	1

The Error Syndrome Code Word will now be compared to each column in Table 2 where it will create a Data Correction Word (DCW) for both the data bits and the check bits. The DCW will be 0000h (0000 0000 0000 0000 b) for the data bits and will be 08h (00 1000 b) for the check bits. Finally, a bitwise XOR between the received information, and the DCW, will be performed. The resulting data word will be 9148h, and the check-bit string will be 32h.

3.4 Double Data Bit Error in a Sixteen Bit Bus Cycle

Starting with the same data loaded into external memory by the UT80CRH196KD in Section 3.1, we will assume that the memory location took two SEU hits causing bits two and twelve of the data word to change state. Consequently, the hexadecimal value of the received data is 814Ch (1000 0001 0100 1100 b) instead of 9148h. The EDAC engine will generate a check bit string whose hexadecimal value is 10h (01 0000 b). Refer to Table 8 to see the check bit generation matrix for the data word 814Ch.

TABLE 8. Check Bit Generation for 814CH

		16 Bit Data Word															
		Bit 15	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Check Bits		**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**0**
CB0	0			0			0	0	1				0	1		0	0
CB1	0		0			0			1		1	0		1	1		0
CB2	0	1			0			0		0		0	0		1	0	
CB3	0				0	0	0			0	1				1	0	0
CB4	1	1	0	0						0	1	0	0	1			
CB5	0	1	0	0	0	0	0	0	1								

Once the new check bits are created, the EDAC engine will find the ESCW by taking the bitwise XNOR of the check bits received from memory with the generated check bits. Table 9 shows how this operation is performed.

TABLE 9. ESC Generation

	XNOR of Check Bits					
Bit Locations	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Read-In CBs	1	1	0	0	1	0
Generated CBs	0	1	0	0	0	0
ESCW	0	1	1	1	0	1

Now the ESCW will be compared to each column in Table 2 to create the DCW. In this case, the DCW is 0000h (0000 0000 0000 0000 b) because the Error Syndrome Code Word could not find a match in any column. Since the DCW is 0000h and the ESCW is not 3Fh (11 1111 b), the EDAC engine indicates that it found a double (or more) bit error. Furthermore, it is obvious that the EDAC can not correct either of the errors, it can only detect and flag the occurrence of a double bit error.

4.0 8-Bit Data Bus Error Detection and Correction Examples

4.1 Data Write in an Eight Bit Bus Cycle

If the UT80CRH196KD wants to write the value A6h (1010 0110 b) to external memory, then the generated check bits will be 09h (00 1001 b). Table 10 shows how the check bits are generated for the data value A6h.

TABLE 10. Check Bit Generation for A6H

		8 Bit Data Word															
		Bit 15	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Check Bits		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**1**	**0**
CB0	1			0			0	0	0				0	0		1	0
CB1	0		0			0			0		0	1		0	1		0
CB2	0	0			0			0		1		1	0		1	1	
CB3	1				0	0	0			1	0				1	1	0
CB4	0	0	0	0						1	0	1	0	0			
CB5	0	0	0	0	0	0	0	0	0								

Note that when using an eight bit bus cycle, the EDAC engine will always put the data being read or written in the LSB and set the MSB to 00h. Therefore, CB5 is always zero in eight bit bus cycles.

4.2 Single Data Bit Read Error in an Eight Bit Bus Cycle

Now, assume that the UT80CRH196KD reads the data placed in memory during Section 4.1. In other words, we want to read back the value A6h from memory. However, the external memory location holding this value took an SEU hit before we could retrieve the data. As a result, data bit zero in the memory cell was changed from a 0 to a 1. Consequently, the data read from the external memory location is A7h (1010 0111 b). This value will be loaded into the EDAC engine where a new set of check bits will be generated. The generated check bit word for the data value of A7h is 02h (00 0010 b). Refer to Table 11 to see the check bit generation matrix for the data value A7h.

TABLE 11. Check Bit Generation for A7H

		8 Bit Data Word															
		Bit 15	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Check Bit		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**1**	**1**
CB0	0			0			0	0	0				0	0		1	1
CB1	1		0			0			0		0	1		0	1		1
CB2	0	0			0			0		1		1	0		1	1	
CB3	0				0	0	0			1	0				1	1	1
CB4	0	0	0	0						1	0	1	0	0			
CB5	0	0	0	0	0	0	0	0	0								

Subsequently, the EDAC engine will find the bitwise exclusive-nor (XNOR) of the read-in check bits with the newly generated check bits. This will give an Error Syndrome Code Word of 34h (11 0100 b) for the data value A7h. The Error Syndrome Code Word generation is shown in Table 12.

TABLE 12. ESC Generation

	XNOR of Check Bits					
Bit Locations	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Read-In CBs	0	0	1	0	0	1
Generated CBs	0	0	0	0	1	0
ESCW	1	1	0	1	0	0

Next, the EDAC engine will determine which bit is in error. The error determination is performed by comparing the Error Syndrome Code Word to each column in Table 2. As the ESCW is compared to each column, a Data Correction Word (DCW) is created. Every 0 in the DCW means that the ESCW did not match the column representing that bit in Table 2. For this example, the DCW is 0001h (0000 0000 0000 0001 b). This indicates a single bit error occurred at bit zero of the data word. Since this is a single bit error, the EDAC engine can correct the error by performing a bitwise exclusive-or (XOR) operation on the data word read in, and the Data Correction Word. After taking the bitwise XOR of 00A7h and 0001h, you will see that the original data word of 00A6h is obtained. Table 13 shows the correcting process for the corrupted data word 00A7h.

TABLE 13. Correcting Data

	XOR of Data and Data Correction Word (DCW)															
Bit Locations	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**But 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Read-In Data	0	0	0	0	0	0	0	0	1	0	1	0	0	1	1	1
DCW	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Corrected Data	0	0	0	0	0	0	0	0	1	0	1	0	0	1	1	0

4.3 Double Data Bit Error in an Eight Bit Bus Cycle

Starting with the same data loaded into external by the UT80CRH196KD in Section 4.1, we will assume that the memory location took two SEU hits causing bits zero and two of the data word to change state. Consequently, the hexadecimal value of the received data is A3h (1010 0011 b) instead of A6h. The EDAC engine will generate a check bit string whose hexadecimal value is 0Ch (00 1100 b). Refer to Table 8 to see the check bit generation matrix for the data word A3h.

TABLE 14. Check Bit Generation for A3H

		8 Bit Data Word															
		Bit 15	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Check Bit		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**1**	**1**
CB0	0			0			0	0	0				0	0		1	1
CB1	0		0			0			0		0	1		0	0		1
CB2	1	0			0			0		1		1	0		0	1	
CB3	1				0	0	0			1	0				0	1	1
CB4	0	0	0	0						1	0	1	0	0			
CB5	0	0	0	0	0	0	0	0	0								

Once the new check bits are created, the EDAC engine will find the ESCW by taking the bitwise XNOR of the check bits read in from memory with the generated check bits. Table 15 shows how this operation is performed.

TABLE 15. ESC Generation

	XNOR of Check Bits					
Bit Locations	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Read-In CBs	0	0	1	0	0	1
Generated CBs	0	0	1	1	0	0
ESCW	1	1	1	0	1	0

Now the ESCW will be compared to each column in Table 2 to create the DCW. In this case, the DCW is 0000h (0000 0000 0000 0000 b) because the Error Syndrome Code Word could not find a match in any column. Since the DCW is 0000h and the ESCW is not 3Fh (11 1111 b), the EDAC engine indicates that it found a double (or more) bit error.

5.1 Three Bit Error Examples

Three bit errors will cause indeterminate results. The hamming code used in the UT80CRH196KD is explicitly a single correct, double detect code. If a triple bit error occurs, the EDAC engine may think that it found a single bit error and correct a single bit, or flag the system that a double bit error occurred. However, we have not attempted all possible triple bit errors to see how the EDAC engine behaves, but we do no believe that a triple bit error can ever slip through with a no error indication.

5.2 Triple Bit Error Resulting in a Single Bit Error Indication

Assume that you are flying commercial memory devices in a medium earth orbit. This orbit is extremely radiated with high energy protons. Consequently, this makes your memory devices very susceptible to single event effects. For example, if the UT80CRH196KD writes the value 0000h (0000 0000 0000 0000 b) into an external memory location, the resulting check bits, according to Table 1, are 00h (00 0000 b). However, if the memory location holding this value

took three SEU hits that change the memory contents to 8108h (1000 0001 0000 1000 h), then the EDAC engine would calculate a check bit value of 04h (00 0100 b) for the received data 8108h. Table 16 shows the check bit calculation for the data value of 8108h.

TABLE 16. Check Bit Generation for 8108H

		16 Bit Data Word															
		Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Check Bit		**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
CB0	0			0			0	0	1				0	1		0	0
CB1	0		0			0			1		0	0		1	0		0
CB2	1	1			0			0		0		0	0		0	0	
CB3	0				0	0	0			0	0				0	0	0
CB4	0	1	0	0						0	0	0	0	1			
CB5	0	1	0	0	0	0	0	0	1								

Now, the EDAC engine will attempt to determine if an error occurred by first creating the ESCW as shown in Table 17; and then by comparing the ESCW to the possible codes in Table 2. The calculated ESCW is 3Bh (11 1011 b). When compared to the columns in Table 2, the ESCW matches the code word for check bit 2 (CB2). As a result, the EDAC engine thinks it found a single bit error, and corrects the problem. Notice that CB2 is changed by the EDAC engine. However, CB2 was never corrupted. Consequently, the corrupted data word is still used by the UT80CRH196KD. Therefore, the EDAC engine can not resolve triple bit errors.

TABLE 17. ESC Generation

	XNOR of Check Bits					
Bit Locations	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Read-In CBs	0	0	0	0	0	0
Generated CBs	0	0	0	1	0	0
ESCW	1	1	1	0	1	1

5.3 Triple Bit Error Resulting in a Double Bit Error Indication

There are cases where a triple bit error will be flagged as a double bit error. This is a pleasant result of using EDAC. Although this hamming code can not detect all un-correctable triple bit errors, it does manage to prevent some errors from getting used by the UT80CRH196KD. For example, if we load the same data into memory as we did in Section 5.2, and assume that three bits in the memory location change state such that the new memory contents are 1110h (0001 0001 0001 0000 b), then we have a case where the EDAC engine will realize that it found an un-correctable double (or more) bit error. The first thing that the EDAC engine will do after reading the data and check bits will be to calculate a new set of check bits. The new check bit string (3Ah (11 1010 b)) will now be used, in conjunction with the received check bits, to create the Error Syndrome Code Word. The ESCW for this example is 05h (00 0101 b). Now, the EDAC engine will compare the ESCW to all the possible code words shown in Table 2. For this case, the EDAC engine does not find a match for the ESCW, and realizes that the ESCW does not equal 3Fh (11 1111 b). Therefore, the EDAC engine decides that it found an un-correctable error, and flags the

system. Note that the EDAC engine knows it can not correct the error, but, it does not know how many errors have occurred.

Although the behavior of the UT80CRH196KD and the EDAC engine to triple bit errors is indeterminate, the chance that it will prevent some multiple error scenarios from penetrating the system is a valuable asset in comparison to the alternative...having no detection and correction capabilities at all. The ability to have EDAC in your system is very important because it offers the system developer the comfort of knowing that most errors that occur in space applications (single and double bit errors) will not be used by the system. As a result, the system protection development time and cost is significantly diminished.

6.0 Flagging Errors

The UT80CRH196KD asserts the EDAC Bit Error interrupt (INT01, 2002h) for single bit errors, double bit errors, and single bit error overflows depending on which type of interrupt condition is enabled in the EDAC_CS register, and INT01 is unmasked in the INT_MASK register. You can control how the UT80CRH196KD will react to the above interrupt conditions by setting bits 5 and 6 of the EDAC Control and Status Register (EDAC_CS) to a logic '1'. Additionally, you can read the EDAC_CS Register to determine how many single bit errors occurred (up to 15); and whether a double bit error has occurred. Table 18 shows the bit definitions of the EDAC_CS Register.

TABLE 18. EDAC_CS[1] Register

Bit Number	Reset Value	Description
7	0	Reserved
6	0	Single Bit Error Overflow Interrupt Enable
5	0	Single Bit Error Interrupt Enable
4	0	Double Bit Error Indication - Set to a logic '1' if a DBE occurred
3-0	0000	Single Bit Error Count (Maximum of 15)

Notes: 1. The EDAC_CS Register is located at location 15h of HWindow 1.