위성체용 2비트 오류검출 및 1비트 정정 FPGA 구현

A SEC-DED Implementation Using FPGA for the Satellite System

노 영 환, 이 상 용
(Young Hwan Lho and Sang Yong Lee)

Abstract : It is common to apply the technology of FPGA (Field Programmable Gate Array), which is one of the design methods for ASIC (Application Specific IC), to the active components used in the data processing at the digital system of satellite, aircraft, missile, etc, for compact, lightness and integration of Printed Circuit Board (PCB). In carrying out the digital data processing, the FPGAs are designed for the various functions of the Process Control, Interrupt Control, Clock Generation, Error Detection and Correction (EDAC) as the individual module.

In this paper an FPGA chip for Single Error Correction and Double Error Detection (SEC-DED) for EDAC is designed and simulated by using a VLSI design software, LODECAP.

Keywords : FPGA, EDAC, bit, SEC-DED

I. 서론

다목적실용위성의 위성본체시스템에서 원격구동장치(Remote Drive Unit)는 컴퓨터 처리장치로서 각종 센서로부터 정보를 수집하고 처리하는 기능을 가지고 있으며 각종 구동기에 명령을 보낸다. 원격구동장치의 메모리는 RAM과 EEPROM으로 구성되며 모든 RAM에 데이터를 저장하거나 EEPROM의 데이터를 읽거나 저장할 때 바이트(byte) 단위로 운용 소프트웨어(Flight Software)와는 관계없이 EDAC을 이용하여 1비트(bit) 오류를 정정하고 2비트 오류를 검출한다.

최근 오류 정정 코드는 컴퓨터 메모리 서브시스템의 신뢰성 및 데이터 처리능력을 향상시키고 데이터 전송 시 채널 코딩기법에서 많이 활용되고 있다. 특히, 반도체 메모리 시스템 설계에서 고집적 및 메모리 용량을 확장시키는데 활용되고 있으며 오류 정정 기능을 가짐으로써 데이터의 신뢰성을 높이고, 오류발생이 오류 정정 능력보다 많을 때는 오류데이터가 발생하므로 데이터 처리능력을 극대화하기 위해 정정할 수 없는 오류를 검출할 수 있어야 한다. 1960년대 컴퓨터 기억장치에서 사용된 오류 정정 코드는 R. W. Hamming 이 발명한 SEC-DED 코드의 부류이었는데 이 코드는 한 워드(Word)에서 오류를 정정하고 검출할 수 있는 기능을 가지고 있다.

컴퓨터 메모리에서 EDAC을 설계할 시 XOR 게이트를 적게 사용하여 회로도에서 처리(processing)속도를 최대로 높이고 에러를 최소화하는데 많은 관심을 가져왔다. 1비트 및 2비트 오류 검출코드[1, 2]는 메모리 소자에서 기능을 수행하는데 충분하지 않으며, [3]에서는 바이트 단위의 오류검출을 할 수 있는 방법을 제시하였다. 이 방법은 IBM 시스템 3081 과 3033을 설계하는데 사용되었다. 1980년대부터 바이트 단위의 SEC-DED코드가 [4], [5] 와 [6]에서 더욱 심도 있게 연구가 진행되었는데 형태는 Hamming(해밍)코드와 같고 비용 및 디코딩로직의 속도 및 신뢰성에서 우수함을 제시하고 있으며 [7]에서는 2 바이트의 체크 비트를 사용하여 64개의 정보를 검출하고 있다. 원격구동장치[8,10]에서 사용되는 컴퓨터 메모리 소자들은 일반적으로 전력 소모가 적게 요구되며 바이트 구성이 8 비트가 될 수 있으며 16 비트 데이터 버스와 적어도 6개의 체크 비트를 가진 시스템은 1비트 및 2비트 오류검출을 수행한다. 이때 바이트 구성이 4 비트나 8 비트인 경우 다른 2개의 체크 비트가 늘어남으로써 추가 메모리를 요구하지 않는다. 우주환경에서 위성체는 방사선에 노출되어 손상을 입을 때 단돌발적인 뒤집힘(single event upset)이 발생되어 디지털 회로는 SEC-DED 기능을 가지고 있어야 된다.

디지털 프로세싱을 수행하는 과정에서 신호의 왜곡, 잡음의 혼재, 전송선에서 데이터의 손실, RAM의 기억소자에서 장시간 저장으로 인한 데이터의 파손을 원래 데이터로 재생하기 위하여 인코더와 디코더를 사용하는데 다이나믹 RAM은 콘덴서에 축적된 전하량에 의하여 논리값 「0」과 「1」을 구별하고, 스태틱 RAM은 비트의 기억을 트랜지스터 및 플립플롭으로 하기 때문에 대기억 용량의 메모리를 제조하는데 사용되며 데이터를 읽거나 쓰는 과정에서 논리값에 오류가 발생할 수 있다. 데이터의 전송이 올바르지 않을 경우 오류를 검출하고 정정을 수행하기 위해 체크 비트를 활용한 16비트 변조(modified) 해밍코드[8]를 사용한다.

데이터 비트와 패리티 체크(parity check) 비트를 분리하여 처리하는 것이 systematic 코드인데 해밍코드[9]를 systematic 코드로 수정한 것이 변조 해밍코드이다. 이 경우 SRAM이나 Mass Memory 용 EDAC 에서 1비트 오류정정과 2비트 오류를 검출하는데 SRAM 경우 바이트 단위로 Mass Memory 경우 16비트 워드 단위로 시행한다. 채널에서 오류 제어 방법은 3가지 유형으로 나눌 수 있다. 첫 번째는 모든 채널코딩기법에서 사용되는 FEC(Forword Error Control) 방법으로 데이터를 수신한 곳에서 오류를 자체적으로 정정하고 두 번째는 수신측에서 수신한

접수일자 : 1999. 6. 7., 수정완료 : 1999. 11. 9.
노영환, 이상용 : 우송대학교 컴퓨터전자정보공학부

데이터를 송신측으로 재 전송하여 송신측에서 오류발생여부를 확인하여 오류발생 시 재 전송하는 ARQ(Automatic Retransmission Request) 방법과 세 번째는 FEC와 ARQ를 혼용하여 사용하는데 오류 정정 기능 및 많은 오류 검출 기능을 갖는 경우에 FEC를 오류발생 시 송신측으로 재 전송하는 경우 ARQ를 이용한다. 이 논문에서는 MASS Memory와 SRAM에서 데이터를 정정하고 검출하는데 있어 FEC 방법으로 디지털 데이터의 오류가 발생한 지점(비트)까지를 찾아 오류를 정정해 준다. 4비트 데이터가 전송될 경우 3개, 16비트 데이터는 6개의 패리티 체크 비트가 필요함을 [11]에서 보여주고 있다.

비트수의 발생을 체크하며 오류를 검출하고 오류 정정과 진단기능을 수행하기 위한 블록도는 그림3과 같다. 체크 비트 생성기에서 표2와 같이 16개의 데이터 중 8개의 데이터에 대한 패리티 비트를 짝수 혹은 홀수 개 생성한다. 그리고 16비트 데이터 포맷은 그림1과 같다. 이 논문에서 SEC-DED 설계 시 최소한의 XOR 게이트를 활용하여 간단히 설계함으로써 데이터처리속도를 줄일 수 있다.

데이터		체크 비트					
BYTE1	BYTE0	CX	C0	C1	C2	C4	C8

그림 1. 16비트 데이터 포맷.

Fig. 1. Data Format of 16 bit.

II. SEC-DED 설계

오류 정정 시 오류가 생긴 비트의 위치를 찾아내기 위하여 변조 해밍코드를 이용하는데 어떤 비트에서 데이터의 오류가 생기면 이 데이터 비트의 위치를 나타내는 코드를 생성하기 위한 체크 비트의 생성이 필요하다. 체크 비트의 수를 늘이면 더욱 많은 비트의 오류 정정 및 검증이 가능하다. 그러나 무한정 체크 비트의 수를 늘이는 것도 현실적으로 문제가 있다.

1. 2진 선형 블록코드 구조

2진 선형 블록코드(binary linear block code)는 열(column)이 n 차원의 벡터공간에서 k차원의 부분공간이다. 여기서 패리티 행렬 H는 r×n으로 구성되어 있는데 r=n-k이다. 모드(modulo) 2의 합산을 해서 $HV'=0$를 만족하면 V는 코드워드(codeword)가 되며 n 비트 벡터로 구성되고 V'는 V의 변환(transpose) 행렬이다. 부호화 과정은 k 데이터 비트에 대하여 r 비트의 체크 비트를 형성하며 H=[P, I]로 구성된다. $P_{r\times k}$는 크기가 r×k 인 2진 행렬을 나타내고 $I_{r\times r}$은 단위행렬을 의미하며, 어떤 2진 $H_{r\times n}$ 행렬이 r의 rank를 가지고 있어 r×r 단위행렬 형태로 변형될 수 있다.

메모리로부터 읽혀진 데이터를 U, 메모리에 저장되어 있는 데이터를 V, 그리고 읽는 과정에서 생겨나는 오류를 E라면, U=V+E의 관계가 성립되는데 만약 i 번째에 오류가 있게되면 i 번째에 오류가 0이 아닌 값을 갖는다. 여기서 U, V, 그리고 E는 n비트 벡터를 나타낸다.

디코딩과정은 U에 오류가 포함되어 있는지를 판단하고 오류벡터 E를 결정한다. U에 오류가 있는지를 결정하기 위하여 r 비트의 패턴 S를 계산하면 $S = H U' = H (V' + E') = H E'$의 관계가 성립된다. S가 모두 0 벡터이면 워드 U는 오류가 없는 경우이고, S가 0이 아니면 U에 오류가 있는 경우이고 오류벡터 E를 결정하는데 이용된다.

반도체 메모리의 설계에서는 부호화(encoding)와 디코딩(decoding)이 동시에 병행해서 일어난다. 부호와 과정에서 체크 비트가 생성되고 디코딩 과정에서는 부호화 과정과 같은 하드웨어를 이용하여 오류검출을 위한 패턴이 생성되는데 오류벡터가 생성된 후에 오류가 있는 비트의 데이터를 반전시켜서 오류를 수정한다.

2. SEC-DED 코드 구조

반도체 메모리 설계에 적용되는 SEC-DED 코드는 데이터 비트와 체크 비트의 관계를 구성하기 위하여 우선 설계에 필요한 특성을 만족하는 범위에서 가능한 한 코드의 길이를 줄여야한다. SEC-DED 코드의 구성은 최소 4비트 이상이어야 한다. H행렬의 3개의 열은 선형적으로 독립적이어야 하며 아래조건을 만족해야한다.

첫째, H 행렬의 열벡터는 0이 아니어야 하고 중복되지 않아야 한다.

둘째, H 행렬의 2개의 열의 모드2의 합은 0이 아니고 다른 열과 같지 않아야 한다.

반도체 메모리에 데이터를 저장하기 위해서는 체크 비트가 $HV'=0$과 H=[P, I]에 의해서 생성되고 읽기 동작에서는 읽어들인 데이터에서 $S=H \cdot E'$에 의해 읽혀진 워드로부터 패턴비트가 동시에 생성된다. 보통 체크 비트와 패턴비트를 생성하는 회로는 그림2와 같은 XOR 구조를 갖는다. SEC-DED 기능을 수행하기 위해서는 S가 아래 조건을 만족해야한다.

첫째, S가 0인지 확인하고 0이면 워드에 오류가 없다.

둘째, S가 0이 아니면 H 행렬의 하나의 열과 S와 완전하게 일치하는 항이 있는지 확인한다. 만약 i 번째 비트에 대한 H행렬과 일치하면 i 번째 비트가 오류가 있다.

셋째, S가 H의 어떤 열과 일치하지 않고 S가 모두 0이 아니면 다수의 오류를 포함하고 있음을 의미한다.

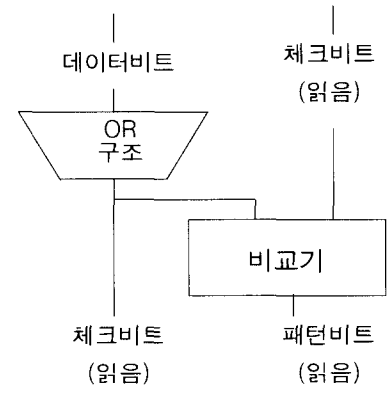


그림 2. 체크비트와 패턴비트의 발생구조.

Fig. 2. Generating structure of check and pattern bit.

16비트의 데이터의 오류정정을 위해서는 최소한 6비트의 체크 비트가 필요하다. 오류가 생긴 비트의 위치를 나타내기 위하여 4개의 체크 비트를 이용할 때 16가지 경우를 나타낼 수 있으나 패리티 체크에서 2비트 오류가 생길 경우 오류를 검증하지 못하는 단점이 있다. 이 경우 1비트 오류를 검증 및 교정이 가능하나 2비트 오류는 검증 자체도 불가능하다. 이러한 문제점을 해결하기 위하여 체크 비트에는 항상 일정한 수의 1이 있어야 한다. 체크 비트에서 1의 개수가 적을수록 여분(redundancy)을 줄일 수가 있어 더 많은 비트의 오류 검증이 용이하지만, 이러한 경우 필요로 하는 체크 비트의 수가 늘어나는 단점이 있다. 표1은 체크 비트 수에 따른 검증 가능한 상태의 수를 보이고 있다. 데이터 비트 수가 8, 16, 32 비트의 경우 각각 5, 6, 7비트의 체크 비트가 필요하고, 체크 비트에는 1의 숫자가 각각 2, 3, 3개 필요로 하는 것을 알 수 있다. 표2에서는 16비트의 데이터 오류교정을 위하여 6비트의 체크 비트가 1의 개수가 3개임을 보여주고 있다.

표 1. 체크 비트와 데이터 비트의 상관관계.
Table 1. Relationship between check bit and data bit.

체크비트수	5			6				7			
1의 갯수	2	3	4	2	3	4	5	2	3	4	5
표현가능 경우의수	10	10	5	15	20	15	6	21	35	35	21
데이터비트수	8	8	4	8	16	8	4	16	32	32	16
체크비트 + 데이터 비트	13	13	9	14	22	14	10	23	39	39	23
4비트 구성시 필요한 비트	16	16	12	16	24	12	12	24	40	40	24
8비트 구성시 필요한 비트	16	16	16	16	24	16	16	24	40	40	24

표 2. 16비트 데이터와 체크 비트의 연계 테이블.
Table 2. Related Table of 16 bit data and check bit.

데이터 비트 / 체크비트	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D1	D1	D1	D1	D1	D1
cx		x	x	x		x			x	x		x			x	
c0	x	x	x		x		x		x		x		x			
c1	x			x	x			x		x	x			x		x
c2	x	x				x	x	x				x	x	x		
c4			x	x	x	x	x	x							x	x
c8									x	x	x	x	x	x	x	x

하나의 데이터 비트에서 오류가 발생되면 이 비트의 위치에 해당하는 체크 비트가 1로 된다. 16비트의 오류를 모두 표현하기 위해서는 48개의 1이 필요하며, 오류가 일어날 확률은 모든 데이터 비트에서 동일하므로 하나의 체크 비트에는 8개 데이터 비트에 대한 패리티 비트를 생성한다. 또한 같은 방법으로 32비트의 데이터에 대한 오류교정을 위해서는 표 1에 의하면 체크 비트 7개에 1의 숫자가 3개 있는 경우에 해당되어 96개의 1이 필요하므로 7개의 체크 비트에 균일하게 분포하는 조합을 구하여 체크 비트 생성 테이블을 작성할 수 있다.

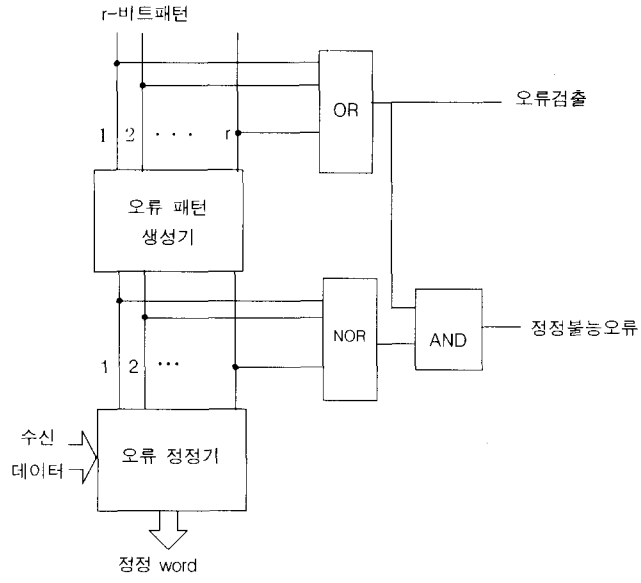


그림 3. 오류 검출 및 정정 블록도.
Fig. 3. Block diagram for error detection and correction.

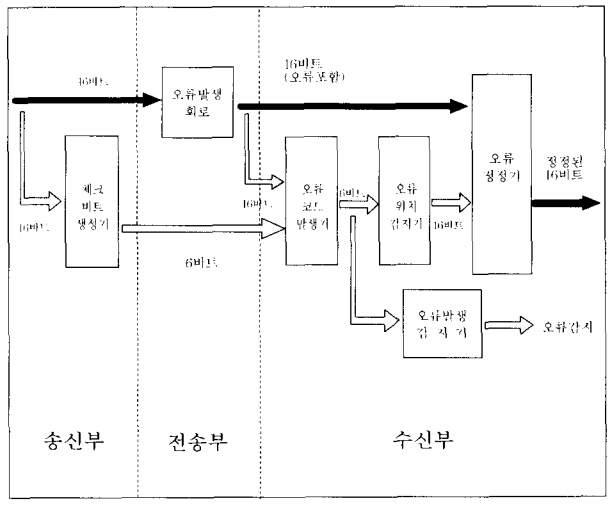


그림 4. SEC-DED의 블록도.
Fig. 4. Block diagram of SEC-DED.

그림 4에서 체크 비트와 함께 전송된 신호는 수신부에서 송신부에서와 같은 알고리즘으로 체크 비트를 생성할 수 있다. 전송되어온 체크 비트와 새로 생성된 체크 비트

를 비교하면 전송도중 생성된 오류의 위치를 표시하는 오류 데이터가 생성된다. 하나의 비트에서 오류가 발생하면 그 데이터 비트에 해당하는 체크 데이터가 생성되고, 이 데이터를 기준으로 어떤 비트에서 오류가 발생되었는지 알 수 있다. 그리고 오류 정정 기능을 갖는 회로를 구성하여 오류가 감지된 비트의 데이터를 역으로 바꾸어 주면 오류교정이 이루어진다.

16비트 데이터에서 1비트 오류에 대해서 3개의 체크 비트가 1로 바뀌고 2비트 오류일 경우 생성된 3개의 비트가 서로 겹치지 않는 경우는 체크 비트에 6개의 1이 된다. 같은 방법으로 1개가 겹치는 경우는 4개, 2개가 겹치는 경우는 2개의 1이 나타남을 알 수 있다. 오류가 전혀 없는 경우는 1이 전혀 나타나지 않는다. 이 경우 체크 비트 생성 테이블을 이용하여 어떤 위치의 데이터가 오류인지를 알 수 있고 교정이 가능하다. 그 외의 경우는 2비트 이상의 오류가 생성됨을 알 수 있다. 오류데이터에서 오류위치를 나타내는 디코더를 설계하여 오류가 있는 비트에서 1로 변하게 된다. 오류위치를 나타내는 디코더 출력에 여러 개의 1이 있게 되면 2비트 이상의 오류가 존재하는 경우이다. 2비트의 오류에 대해서는 감지가 가능하지만 위치를 찾을 수는 없다. 그리고 3비트 이상의 오류가 발생하면 오류발생 감지가 불가능하다.

3. SEC-DED의 구성 및 기능

3.1 체크 비트 생성기 (check bit generator)

16비트 전송 데이터를 6비트의 체크 비트를 생성하는 회로는 표2를 기준으로 16비트 데이터 중 8개를 선택하여 패리티 비트를 생성한다.

3.2 오류 발생 회로(error generation circuit)

전송되는 데이터에 오류를 발생시키는 회로로써 실제 데이터 전송 시 발생될 수 있는 오류를 XOR 게이트의 입력 단에 1을 인위적으로 발생시켜 선택된 데이터 비트의 값을 역으로 변환하는 기능을 수행하는 회로이며 16개의 XOR게이트를 이용하여 구성할 수 있다.

3.3 오류 코드 발생기(error code generator)

체크 비트 생성기에서 전송되어온 6비트의 체크 비트와 오류 발생 회로의 16비트 출력 값을 이용하여 표2에 의한 새로운 6비트의 체크 비트를 비교하여 다른 부분의 값을 1로 변환하는 장치로 오류코드를 발생하는 장치이다. XOR를 이용해서 비교기능을 수행하여 서로 다르면 출력이 1, 같으면 0의 출력을 갖는다.

3.4 오류 위치 감지기(error position detector)

표2를 기준으로 발생기에서 출력되는 6비트의 오류 코드에서 1이 발생 시 체크 비트를 생성하는데 필요한 16비트 데이터 비트 중에 8비트의 관련 데이터 비트 중에 1비트의 위치를 감지하는 기능을 수행한다. 예로서, 체크 비트의 C0, C1, C2가 1일 때 데이터 비트 D0 가 오류이다.

3.5 오류 정정기(error corrector)

오류가 감지되면 그 위치의 데이터를 정정하여야 한다. XOR게이트를 이용하여 데이터 비트의 번지가 1로 되면 그 번지의 데이터를 반전시키는 역할을 한다. 그리고 오류가 없는 경우 0의 입력이 인가되고 수신된 데이터는 변환없이 그대로 전달하는 기능을 수행한다.

3.6 오류 발생 감지기(error generation detector)

오류 코드 발생기에서 출력된 6비트를 그림5와 같이 3개의 전가산기[9]를 이용하여 6비트 오류코드에서 1의 개수가 3비트일 때 1비트 오류이며, 1의 개수가 2비트, 4비트, 그리고 6비트인 경우는 2비트 오류에 해당한다. 그리고 나머지의 경우는 다수 비트 오류이다.

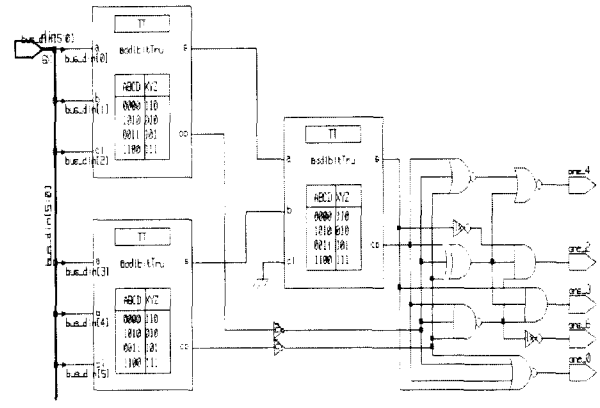


그림 5. 오류발생 감지기.

Fig. 5. Error Generation Detector.

III. 시뮬레이션

그림 6은 SEC-DED 전체 회로로서 그림 4의 모듈별 블록도이다. 송신부에서 체크 비트 생성기는 16비트 입력 데이터를 이용하여 체크 비트를 생성하고 전송부에서 오류발생회로는 16비트 데이터와 16개의 선택단자(sel) 값을 XOR를 이용하여 인위적인 오류가 포함된 16비트 데이터를 생성한다. 오류코드 발생기는 체크 비트 생성기와 같은 기능을 수행하는 체크 비트 발생기와 비교기로 구성된다. 오류 발생회로의 출력 16비트와 체크 비트 생성기의 출력 6비트를 이용하며 비교기의 출력은 6비트로 오류위치 감지기에 입력된다. 1비트 오류일 때 6개의 체크 비트 중에 3개의 1이 생성되어 오류위치 감지기와 오류발생 감지기에 입력된다. 2비트 오류일 때 체크 비트 범위 내에서 1의 개수가 짝수개이다. 오류 위치 감지기의 출력은 16비트이며 오류 정정기의 입력으로 들어가 오류 정정기에서 감지된 비트의 데이터를 정정하는 기능을 수행하는 회로로서 XOR 게이트를 이용하여 데이터 비트의 번지가 1로 되면 해당 비트 데이터를 반전하고 0이 되면 데이터를 정정할 필요가 없다. 그리고 오류 발생 감지기는 오류코드 발생기의 비교기에서 출력된 6비트의 데이터를 입력받아 3개의 전가산기[그림5]와 기본게이트의 조합으로 구성되어 2비트까지 오류를 감지하는데 16비트 데이터를 입력받아 출력이 1일 때 오류가 발생한다. 그림7은 그림6의 SEC-DED 회로에 대한 16비트 데이터 오류감지 및 정정 결과를 시뮬레이션으로 보여주고 있으며 내용을 정리하면 표3과 같다. 구성 내역은 7종류로서 윗상단부터 hexa 코드로 된 16비트 오류 데이터(/err_sel[15:0]), 16비트 전송데이터

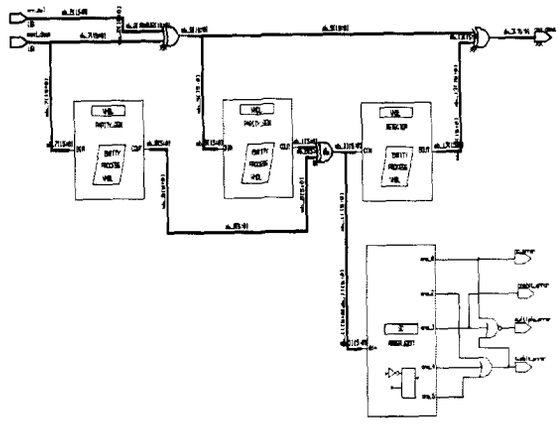


그림 6. SEC DED 회로도.
Fig. 6. Circuit of SEC DED.

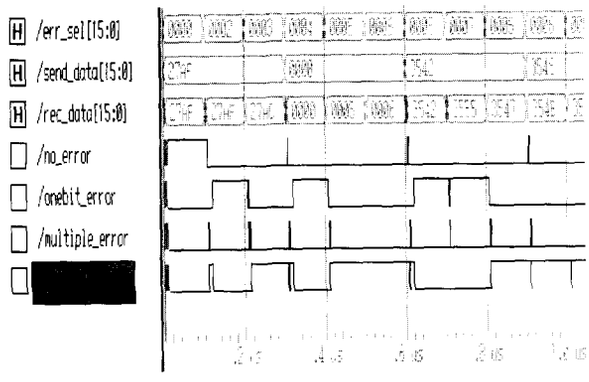


그림 7. 16비트 데이터의 오류감지 및 정정 결과.
Fig. 7. Result of error detection and correction of 16 bit data.

표 3. 16비트 데이터의 오류감지 및 정정 결과.
Table 3. Result of error detection and correction of 16 bit data.

입력데이터 (/send_data[15:0])	27AF			0000			3542			3548		
오류데이터수 (/err_sel[15:0])	0 (비트)	1	2	1	2	2	1	3	2	2	2	2
무오류(/no_error)	×											
2비트이상오류 (/multiple_error)			×		×	×		△	×	×	×	×
1비트오류 (/onebit_error)		×		×			×					
정정데이터 (/corrected_data[15:0])	27AF		□	0000	□	□	3542	□	□	□	□	□

× ; 2비트 이하 오류감지
□ ; 정정 안된 데이터
△ ; 오류감지 안됨

(/send_data[15:0]), 16비트 정정 데이터(/rec_data[15:0]), 무오류(/no_error), 1비트 오류감지(/one bit_error), 다수비트 오류감지(/multiple _ error), 2비트 오류감지(/two bit_error)이다. 16비트 데이터에서 1비트 및 2비트 오류를 정확히 감지함을 보여주는데 데이터의 입력이 27AF일 때 오류 데이터가 0000으로 없거나 0002로 1비트 있을 때 무오류 및 1비트 오류의 감지신호가 high 상태임을 보여주며 정정 데이터가 27AF로 정정되어 출력된다. 오류데이터 0003과 같이 2비트 오류일 경우 감지 신호가 high로 감지는 되지만 교정된 데이터가 27AC로 정정이 되지 않으며, 입력이 3542 일 때 오류데이터가 0007과 같이 3비트 오류일 경우 감지조차도 되지 않고 정정 데이터가 3555로 원래 3542로 정정이 되지 않음을 보여주고 있다.

IV. FPGA 구현

FPGA를 구현하기 위해 디지털 CAD인 LODECAP의 VHDL 논리합성 및 회로도에서 지원하는 로직을 이용하여 설계한 후 결과를 ALTERA의 설계 소프트웨어인 MAX PLUS II의 EDIF 형태로 변환하였다. 생성된 EDIF file을 MAX PLUS II에서 읽어들여 시뮬레이션을 수행함으로써 회로의 동작상태를 검증하였다.

FPGA를 구현하기 위하여 ALTERA flex40k의 테스트 보드에 설계 결과를 다운로드하고 동작상태를 검증하기 위하여 16비트 데이터 송/수신 중에 발생되는 오류를 인위적으로 생성한다. 테스트보드의 구성은 상단에 7 segment를 이용하여 전송부와 수신부의 데이터처리 결과를 보여주고 있다. 좌측 4개의 표시기는 전송부이고 우측 4개는 수신부의 데이트를 표시한다. 우측 하단 16개의 누름 버튼은 16비트 데이터에 오류를 인위적으로 발생시키기 위한 장치다. 그림 8 및 9에서 보여주듯이 버튼을 누를 경우에 16비트의 데이터 중에 그 번지에 해당되는 데이터에 오류가 생성되는데 1비트 오류는 교정이 되고 2비트 이상의 오류일 경우 교정이 되지 않음을 시뮬레이션 결과와 같은 결과를 확인하였다. 그림 8에서는 1비트 오류의 발생을 버튼 1개를 눌러 16비트의 데이터중에 해당번지에 오류가 생성

그림 8. 16비트 데이터에서 1비트 오류 시 FPGA 구현.
Fig. 8. Realization of FPGA for 1 bit error of 16 bit data.

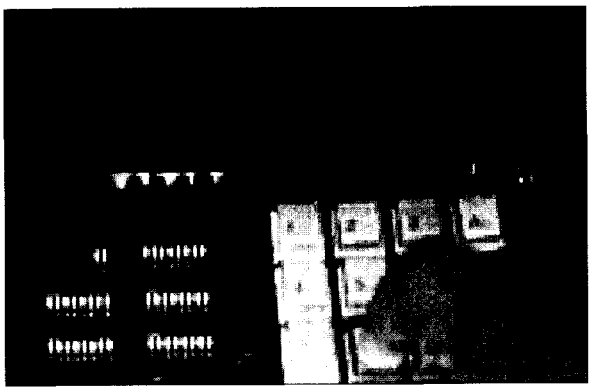


그림 9. 16비트 데이터에서 2비트 오류 시 FPGA 구현.
Fig. 9. Implementation of FPGA for 2 bit error of 16 bit data.

되는데 입력 8CA8가 출력 8CA8로 교정이 되고, 그림 9에서는 2비트 이상의 오류의 발생을 버튼 2개 이상을 눌러 생성시키는데 2비트 이상의 오류일 경우 입력 E796가 출력 6696로 교정이 되지 않음을 시뮬레이션 결과와 같음을 확인하였다.

V. 결 론

위성체의 원격구동장치의 디지털 프로세싱을 수행하기 위해 16종의 FPGA가 사용되는데 이 논문에서는 메모리에 데이터를 저장하고 읽을 때 오류 정정 및 감지기능을 수행하는 EDAC용 FPGA를 설계하였다. EDAC 설계 시 사용된 게이트 수는 163개이며 2단자 입력 NAND 게이트로 환산하면 136.5 등가 게이트로 표시되고 전력은 약 0.5 [W] 소요된다. 원격구동장치는 1553B 데이터 버스를 이용하여 각종 FPGA 데이터를 관련 서브시스템에 송/수신한다. EDAC의 기능은 16비트 데이터의 1비트 오류정정과 2비트의 오류감지를 수행하는 회로이다. LODECAP을 이용하여 회로 설계 및 시뮬레이션을 수행함으로써 회로의 동작을 검증하였으며 논리합성을 이용한 회로 설계 기법을 이용하여 간단히 설계함으로써 데이터 처리속도를 줄일 수 있고 설계 시간을 단축할 수 있다.

LODECAP에서 설계된 결과를 ALTERA FPGA의 데이터 형태로 변환시켜 ALTERA FLEX 칩을 이용하여 회로 구현 및 동작을 확인하였다. 체크 비트 수를 증가시켜 32비트 이상의 데이터의 오류 및 정정 기능을 수행할 수 있으며 설계에서 사용되는 알고리즘으로 보다 많은 데이터의 오류 및 검증을 용이하게 할 수 있는 기반을 구축 하리라본다. 또한, 디지털 프로세싱을 수행하는 메모리 시스템 설계 시 이 논문에서 제시한 오류검출 및 정정 기법이 활용될 수 있으리라 본다.

참고문헌

[1] D. C. Bossen, L. C. Chang, and C. L. Chen, "Measurement and generation of error correcting codes for package failures", *IEEE Trans. on Comp.*, vol. c-27, no. 3, pp 201-204, Mar., 1978.

[2] S. M. Reddy, "A class of linear codes for error control in byte-per-card organized digital systems", *IEEE Trans. on Comp.*, vol. c-27, no. 5, pp. 455-459, May, 1978

[3] C. L. Chen, "Error-correcting codes with byte error-detection capability", *IEEE Trans. on Comp.*, vol. c-32, no. 7, pp. 615-621, July, 1983

[4] S. Kaneda, "A class of odd-weight column sec-ded-sbed codes for memory system applications", *IEEE Trans. on Comp.*, vol. c-33, no. 8, pp. 737-739, Aug., 1984.

[5] L. A. Dunning, "SEC-BED-DED codes for error control in byte-organized memory systems", *IEEE Constructions of Some Bit and Byte Error*

[6] W. E. Clark, L. A. Dunning, and D. G. Rogers, "Control codes using partial steiner systems", *Trans. on Comp.*, vol. c-34, pp. 557-562, Jun., 1985. IEEE Trans. on Infor. Theory, vol. IT-35, no. 11, pp. 1305-1310, Nov., 1989.

[7] L. A. Dunning, "A SEC-BED-DED code with byte plus bit error detection", *Proc. of 24th Inter. Conf. on Fault-Tolerant Computing*, pp. 208-211, June, 1994.

[8] TRW Civil & International Systems Division Space & Electronics Group, "KOMPSAT equipment specification for remote drive unit", April, 1996

[9] 16비트 중심의 "마이크로 컴퓨터" 조경록, 오무송, 김형래, 박규성 편저, 광문각, 1994.

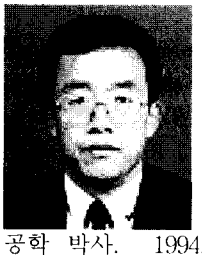
[10] 다목적 실용위성의 "원격구동장치의 하드웨어 설계 연구(제4차년도 연차보고서)", 1998. 5. 16, 산업자원부, 과학기술부, 정보통신부.

[11] AM2960/AM2960A, 16-bit Error Detection and Correction Unit, Adavanced Micro Devices (AMD) Databook.

[12] 노영환, 이상용, 이재학, "위성체의 접속회로의 WORST CASE를 고려한 최적설계", 제2권 pp. 1091-1094, *Proceedings of the 13th KACC*, October, 1998, Pusan, Korea.

[13] 노영환, 이상용, 김진철, "위성체의 에러검출 및 정정 FPGA 구현", 제2권 pp. 1095-1097, *Proceedings of the 13th KACC*, October, 1998, Pusan, Korea.

[14] LODECAP 사용자 설명서, ETRI, 1997.



노 영 환

1954년 6월 2일생. 1982년 경북대학교 전자공학과 졸업. 1981.11~1985.12 LG정보통신(주) 근무. 1988년 University of New Mexico 전기공학 석사. 1993년 Texas A&M University 전기공학 박사. 1994.2~1995.2 한국항공우주연구소 근무., 1995.3~현재 우송대학교 컴퓨터전자정보공학부 부교수. 관심분야는 견실제어, 적응제어, 신호처리, 디지털회로 설계등.

이 상 용

1960년 4월 27일생. 1983년 서강대학교 전자공학과 졸업. 1985년 동대학원 석사. 1985~1988.8 전자통신 연구원 연구원. 1993년 Texas A&M University 전기공학 박사. 1994.2~1996.2 삼성전자 마이크로사업부 연구원. 1996.3~현재 우송대학교 컴퓨터전자정보공학부 조교수. 관심분야는 고내압소자 설계, 소자모델링 및 ASIC 설계등.