# 비동기 상태 피드백 제어를 이용한 TMR 메모리 SEU 극복

論 文
57-8-21

## Asynchronous State Feedback Control for SEU Mitigation of TMR Memory

楊 正 敏† · 郭 成 祐*
(Jung-Min Yang · SeongWoo Kwak)

**Abstract** - In this paper, a novel TMR (Triple Modular Redundancy) memory structure is proposed using state feedback control of asynchronous sequential machines. The main ability of the proposed structure is to correct the fault of SEU (Single Event Upset) asynchronously without resorting to the global synchronous clock. A state-feedback controller is combined with the TMR realized as a closed-loop asynchronous machine and corrective behavior is operated whenever an unauthorized state transition is observed so as to recover the failed state of the asynchronous machine to the original one. As a case study, an asynchronous machine modelling of TMR and the detailed procedure of controller construction are presented. A simulation results using VHDL shows the validity of the proposed scheme.

**Key Words** : Asynchronous machines, Corrective control, Triple modular redundancy(TMR), Single event upset(SEU)

### 1. 서 론

전역 클럭(clock) 없이 동작하는 비동기 순차 머신은 동기 순차 머신에 비해서 짧은 반응 시간, 전력 소비 감소 등 여러 가지 장점을 지니고 있다[1]. 디지털 시스템에서 사용되는 기본적인 논리 회로이외에도 병렬 연산의 효율을 높이기 위한 비동기식 설계[2], 비동기 라우팅 칩(routing chip) 제작[3], 캐시 컨트롤러(cache controller) 등 비동기 방식을 활용한 전용 하드웨어 설계 및 산업계 응용에 관한 연구는 여전히 활발하게 발표되고 있다.

본 논문은 비동기 방식의 상태 피드백 제어를 이용한 회로 보정(補正에) 대해서 다룬다. 교정 제어(Corrective Control)라고도 불리는 비동기 피드백 제어는 과도 상태(transient state)에서 머무르는 시간이 극히 짧은 비동기 시스템의 성질을 이용하여 폐루프(closed-loop) 시스템의 정상 상태(stable state) 동작을 바꾸는 방법이다. 즉 역시 비동기 머신의 형태로 설계한 제어기를 제어 대상 비동기 머신 앞에 두고 입력과 머신의 출력 피드백 값을 받아 제어 입력을 생성하면 폐루프 시스템은 원하는 동작을 보인다. 이러한 기법을 이용하여 비동기 머신 내에 존재하는 레이스(race)를 없애거나[4], 무한 순환(infinite cycle)에 빠진 머신을 정상으로 되돌리는 연구[5] 등이 발표되었다. 한편 저자들의 선행 연구 [6]에서는 외란 입력이 존재하는 비동기 머신의 모델 매칭(model matching)을 위한 제어기가 제안되었다.

본 논문의 목적은 TMR(Triple Modular Redundancy) 메모리 SEU(Single Event Upset) 극복을 위한 비동기 상태 피드백 제어 시스템을 제안하는 일이다. TMR 메모리는 우주, 원자로 내부 등 외부 외란이 존재하는 열악한 환경에서 메모리 비트(bit)가 반전되는 현상을 극복하기 위해서 만들어진 것으로[7] 세 개의 메모리 셀(cell)에 동일한 데이터를 저장하고 출력 시에는 세 메모리 데이터를 다수결 회로(majority voter circuit)를 거쳐 외부로 출력하는 방식이다. 종래의 TMR 메모리는 모두 전역 클럭이 존재하여 일정 시간마다 다수결 회로의 결과를 피드백 하여 리셋(reset) 시키는 방식을 사용하였다. 하지만 메모리 반전 오류의 발생은 일반적으로 비동기적으로 발생하므로 한 클럭 안에서 여러 개의 SEU가 발생하면 TMR 값이 틀리게 바뀌어버리는 약점이 있다.

본 논문에서는 TMR 메모리를 비동기 머신으로 모델링하고 SEU 고장을 외란 입력 집합으로 설정하여 문제를 해결한다. SEU 고장이 발생하여 머신이 원하지 않는 상태 천이를 겪는 즉시 상태 피드백 제어기는 머신을 원래 상태로 되돌리는 교정 동작을 실시한다.

본 논문의 순서는 다음과 같다. 2장에서는 먼저 외란 입력이 존재하는 비동기 순차 머신을 모델링한다. 선행 연구 [6]에서는 비동기 머신이 외란 입력과 정상 입력 두 개의 입력을 동시에 취하는 기법을 제안하였으나 본 연구에서는 모두 하나의 입력 집합 내에 두는 방식을 취한다. 3장에서는 외란 입력에 대한 교정 제어기를 설계한다. 비동기 머신이 정상 상태에 머무를 때에만 입력과 상태 변수가 변경될 수 있다는 기본 모드(fundamental mode) 조건이 만족시키는 상태 피드백 제어기의 동작을 기술한다. 4장과 5장에서는 비동기 방식으로 구동되는 TMR 구조 및 TMR을 위한 비동기 상태 피드백 제어 시스템을 사례 연구로서 제시한다. 6장에서는 소프트웨어 모의실험을 통해서 5장에서 제안한 TMR을 위한 비동기 상태 피드백 제어 시스템의 성능을 검

† 교신저자, 正會員 : 大邱가톨릭大 電子工學科 副敎授 · 工博
E-mail : jmyang@cu.ac.kr
* 正 會 員 : 啓明大 電子工學科 助敎授 · 工博
接受日字 : 2008年 2月 29日
最終完了 : 2008年 6月 25日

증한다. 마지막으로 7장에서 본 논문의 결론을 내린다.

## 2. 비동기 순차 머신

본 논문에서는 비동기 머신을 다음과 같은 유한 상태 머신으로 표현한다.

$$\Sigma = (A, Y, X, x_0, f, h) \quad (1)$$

위 식에서 A는 입력 집합, Y는 출력 집합, X는 상태 집합, $x_0$는 초기 상태이다. $f: X\times A\rightarrow X$와 $h: X\times A\rightarrow Y$는 각각 상태 천이 함수와 출력 함수이다. 본 연구에서 고려되는 비동기 머신은 머신의 현재 상태가 출력으로 나오는 입력/상태 머신이다. 따라서 항상 h(x)=x의 관계를 가진다. 출력 집합이 상태 집합과 다른 입력/출력 비동기 머신(input/output asynchronous machine)에 대한 모델링 및 제어기 설계는 [12]의 기존 결과를 응용하여 구할 수 있다.

비동기 머신은 입력과 상태 조합에 따라서 과도 조합(transient combination)과 안정 조합(stable combination) 등 두 가지 종류의 상황에 처할 수 있다[8]. 어떤 안정 조합에서 입력 값이 바뀌어 과도 조합이 되면 머신은 연쇄적인 상태 천이를 시작한다. Σ의 과도 조합을 $(x,u)\in X\times A$라 하고 연쇄 상태 천이를 f로 나타내면 다음과 같다.

$$x_1 = f(x,u),\ x_2 = f(x_1,u),\cdots$$

이러한 상태 천이가 끝나지 않고 무한히 반복된다면 머신 Σ은 무한 순환에 빠졌다고 말한다[8]. 무한 순환이 발생하지 않는다고 가정한다면 Σ의 천이가 끝나는 어떤 상태 x′가 존재하는데 이 상태를 (x,u)의 다음 안정 상태(next stable state)라고 정의한다[4]. 함수 f로 다음 안정 상태를 표시하면 x′=f(x′,u), 즉 x′는 입력 u에 대한 f의 고정점이 된다.

비동기 머신은 과도 상태에서 아주 빠르게 상태 천이가 일어나므로(이론적으로 0초) 외부 사용자가 관찰할 수 있는 동작은 안정 상태 사이의 천이밖에 없다. 비동기 머신 Σ의 안정 상태에서의 동작만을 추출하기 위해서 stable recursion 함수 s를 다음과 같이 도입한다[4].

$s(x,u) = x'$(x′는 (x,u)의 다음 안정 상태)

통상 s는 확장되어 길이가 2 이상인 입력 스트링(string)을 변수로 가질 수 있다. $x\in X$이고 $u\in A, t\in A^+$라면 s는 다음과 같이 재귀적으로 정의된다.

$$s(x,ut) = s(s(x,u),t) \quad (2)$$

f 대신 s를 사용하여 식 (1)을 다시 표현한 아래의 머신 $\Sigma_{|s}$을 Σ에 대한 stable-state 머신이라고 정의한다[4].

$$\Sigma_{|s} = (A, Y, X, x_0, s, h) \quad (3)$$

비동기 머신에 존재하는 외란 입력을 표시하기 위해서 본 논문에서는 입력 집합 A가 서로 소인 두 부분 집합의 합으로 이루어진다고 설정한다.

$$A = A_c \cup A_d \ (A_c \cap A_d = \varnothing)$$

$A_c$는 정상 입력 집합이며 $A_d$는 외란 입력 집합이다. 일반적으로 $A_d$에 속하는 입력 알파벳의 발생은 관측될 수 없다. 하지만 정상 입력은 항상 관측 가능하므로 정상 입력의 값은 그대로이고 상태 피드백의 값이 바뀌었다면 제어기는 외란 입력이 발생하였다는 사실을 감지할 수 있다.

## 3. 외란 입력에 대한 교정 제어

외란 입력은 사용자가 원하지 않는 상태 천이를 야기한다. 본 논문의 목적은 외란 입력에 의해서 상태가 바뀌었을 때 머신을 원래 상태로 복구시키는 교정 제어기를 설계하는 일이다. 그림 1의 교정 제어기 C는 외부 입력 v와 상태 피드백 y(=x)를 동시에 입력으로 취하고 머신 Σ에 제어 입력 u를 넣는다. 제어기 역시 비동기 순차 머신의 형태를 가지므로 C는 다음과 같은 유한 상태 머신으로 정의된다[4],[5].

$$C = (X\times A, A, \Xi, \xi_0, \phi, \eta) \quad (4)$$

위 식에서 Ξ는 C의 상태 집합, $\xi_0\in\Xi$는 초기 상태이며 $\phi: \Xi\times X\times A\rightarrow\Xi$와 $\eta: \Xi\times X\times A\rightarrow A$는 각각 제어기의 천이 함수와 출력 함수를 가리킨다.

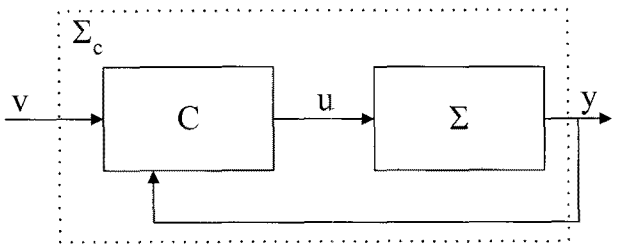


**그림 1** 교정 제어 기본 구조.

**Fig. 1** Basic configuration of corrective control.

제어기 C의 설계를 예시하기 위해서 비동기 머신 Σ의 동작을 다음과 같이 설정한다. Σ이 어떤 상태 $z\in X$와 안정 조합을 이룰 때 외란 입력 $b\in A_d$가 들어올 수 있다고 하자. b가 발생하면 Σ는 다음 안정 상태 z′로 천이한다. 즉 s(z,b) = z′이다. 제어기 C의 역할은 머신 Σ가 외란 b에 의해서 z′로 천이하는 즉시 교정 동작을 작동하여 머신의 상태를 z로 되돌리는 일이다.

이러한 교정 동작을 만들기 위해서 필요한 조건은 비동기 머신 Σ가 z′에서 z로 가는 도달가능성(reachability)을 보유해야 한다는 사실이다. 식 (3)의 stable-state 머신에서 $s(z',\alpha) = z$인 입력 스트링 $\alpha\in A_c^+$가 존재하면 z는 z′로부터 stably reachable하다고 부른다[4]. 여기서 주목해야 할 것은 α가 정상 입력 집합 $A_c$에 속한 알파벳으로만 이루어진 스트링이어야 한다는 점이다. 바꾸어 말해서 Σ가 외란 입력에 의해서 상태 천이 되었으나 천이된 상태로부터 원래 상태까지 정상 입력만으로 이루어진 제어 경로(control path)가 없다면 교정 동작을 이루는 제어기가 존재하지 않는다.

초기 상태 $\xi_0$에 있던 제어기 C는 Σ가 z와 안정 조합을 이룰 때 상태 $\xi_0(x)$로 천이한다. 제어기는 $\xi_0$에서 $\xi_0(x)$로 이동함으로써 z에서 일어날 수 있는 외란 입력 b에 의한 상태 천이를 대비한다. 따라서 C의 천이 함수 φ는 아래와 같이 정의된다.

$$\phi(\xi_0,(x,v))=\xi_0 \qquad \forall\,(x,v)\notin(z,U(z)) \tag{5}$$
$$\phi(\xi_0,(z,v))=\xi_0(x) \qquad \forall\,v\in U(z)$$

위 식에서 $U(z)\subset A_c$는 상태 z와 안정 조합을 이루는 모든 정상 입력 집합을 말한다. 제어기 C가 아직 아무 교정 동작을 하지 않으므로 출력 함수 값으로 외부 입력을 그대로 전달해준다.

$$\eta(\xi_0,(x,v))=v \qquad \forall\,(x,v)\in X\times A \tag{6}$$

$\xi_0(x)$에서 제어기는 외란 입력의 발생을 기다린다. 외부 입력이 바뀌지 않는 한 $\Sigma$를 계속 상태 z에 머무르게 하기 위해서 C는 $t\in U(z)$인 정상 입력 t를 하나 설정하여 출력 함수 값으로 한다.

$$\eta(\xi_0(x),(z,v))=t \qquad \forall\,v\in A \tag{7}$$

상태 피드백이 z에서 z′으로 바뀌는 순간 제어기 C는 외란 입력이 일어났다는 사실을 감지하고 교정 동작을 실행한다. 앞에서 $s(z',\alpha)=z$인 입력 스트링 α가 존재한다고 가정하였다. $\alpha=u_1u_2\cdots u_m$라 하면 z′에서 z까지의 상태 천이는 아래와 같이 표현된다.

$$(z',b)\overset{u_1}{\longleftrightarrow}(x_1,u_1)\overset{u_2}{\longleftrightarrow}(x_2,u_2)\cdots\overset{u_m}{\longleftrightarrow}(z,u_m) \tag{8}$$

$\Sigma$는 $u_1u_2\cdots u_m$가 들어오면 $x_1,\cdots,x_{m-1}$의 중간 상태를 거쳐서 마지막에 z로 간다. 위 식에서 $(x_i,u_i)$와 $(z,u_m)$은 모두 안정 조합을 이룬다. 교정 제어의 핵심은 제어기의 동작을 통해서 폐루프 시스템 $\Sigma_c$ 안에서 이러한 안정 조합이 과도 조합의 특성을 보이게 만드는 일이다. 즉 머신 $\Sigma$가 외란 입력에 의해서 상태 z′로 천이된 즉시 (8)의 상태 천이가 연쇄적으로 일어나게 해서 극히 짧은 시간에 머신이 원래 상태 z로 돌아가게 한다. 먼저 $\xi_0(x)$에서 제어기의 동작은 아래와 같이 정의된다.

$$\phi(\xi_0(x),(z,v))=\xi_0(x) \qquad \forall\,v\in U(z) \tag{9}$$
$$\phi(\xi_0(x),(z,v))=\xi_0 \qquad \forall\,v\notin U(z)$$
$$\phi(\xi_0(x),(z',b))=\xi_1$$

$\xi_1\in\Xi$은 외란 입력을 감지한 후 상태 복구를 시작하기 위해서 제어기 C가 이동하는 첫 번째 상태이다. 그런데 비동기 머신은 안정 조합에서 변수 두 개 이상이 동시에 변할 수 없다는 기본 모드의 원리를 반드시 준수해야 한다. 따라서 (9)의 세 번째 식은 상태와 입력이 (z,t)에서 (z′,b)로 동시에 변하는 게 아니라 먼저 외란 입력 b가 발생하고 $\Sigma$의 상태가 z′로 바뀌어 피드백 값으로 전달되고 난 후 제어기 C의 천이가 벌어진다고 해석해야 한다.

$\xi_1$로 천이한 후 C는 $\Sigma$에 입력 스트링 α의 첫 번째 알파벳 $u_1$을 제어 입력으로 넣는다. $u_1$을 받은 $\Sigma$는 식 (8)에 나와 있는 대로 상태 $x_1$으로 천이된다. $x_1$을 상태 피드백으로 받은 C는 다시 다음 상태 $\xi_2$로 옮겨 가고 두 번째 입력 알파벳 $u_2$를 제어 입력으로 준다. 이러한 연쇄적인 C와 $\Sigma$의 동작을 구현하면 다음과 같다.

$$\phi(\xi_i,(x_i,b))=\xi_{i+1} \tag{10}$$
$$\eta(\xi_i,(x_i,b))=u_i, \qquad i=1,\cdots,m-1$$

마지막으로 상태 $\xi_m$로 옮겨간 C는 $\Sigma$의 상태를 z로 복구시킨다. 외부 입력이 다시 다른 값으로 바뀔 때까지 C는 α의 마지막 알파벳 $u_m$을 제어 입력으로 넣어 $\Sigma$를 z에 머무르게 한다.

$$\phi(\xi_m,(z,b))=\xi_m \tag{11}$$
$$\phi(\xi_m,(x,v))=\xi_0 \qquad \forall\,(x,v)\neq(z,b)$$
$$\eta(\xi_m,(x,v))=u_m \qquad \forall\,(x,v)\in X\times A$$

제어기는 현재의 상태와 입력이 (z,b)가 아닌 다른 값으로 들어오면 초기 상태 $\xi_0$로 돌아가 또 다른 상태 복구를 준비한다((11)의 두 번째 식 참조).

(z,b) 조합 외의 다른 상태와 외란 입력에서 벌어지는 원하지 않는 상태 천이에 대한 교정 제어기도 위에서 기술한 것과 비슷하게 설계된다. 만약 k 개의 교정 제어기 $C_1,\cdots,C_k$가 설계된다고 하면 기존 연구 [5]에서 사용되었던 *join* 연산 "∨"을 이용하여 각 제어기를 결합하여 하나의 제어기 C를 만들 수 있다.

$$C=C_1\vee C_2\vee\cdots\vee C_k$$

join 연산은 각 제어기의 초기 상태 $\xi_0$와 두 번째 상태 $\xi_0(x)$를 각각 하나로 결합하고 약간의 동작 조정을 통해서 이루어진다[5]. 교정 제어기의 자세한 설계 예는 다음 장에서 설명한다.

## 4. TMR 메모리

TMR 메모리가 필요한 환경의 예로 위성체가 작업하는 우주를 들 수 있다[9]. 먼 우주로부터 오는 우주선(cosmic ray)에 의한 고에너지 입자, 태양 활동에 의해 분출되는 입자, 또는 지구 자기장에 붙잡혀 있는 여러 에너지 입자들에 의해서 위성체에 탑재된 전자회로가 영향을 받게 된다. 이 중에서 메모리 또는 플립플롭(flip-flop)에 발생하는 중요한 영향은 이들 고에너지 입자 때문에 메모리에 저장되어 있던 정보가 "1"→"0" 또는 "0"→"1"로 변경되는 SEU 현상이다[7]. 이와 같은 SEU에 대처하기 위해 위성체에서 사용되는 메모리 소자는 SEU 현상이 발생하지 않도록 하는 특수한 공정으로 제조된 우주용 메모리 소자를 사용하거나, 일반 공정으로 제조된 메모리 소자에 TMR과 같은 오류 극복 기법을 채용하는 방식을 취하고 있다. 특히 최근 각광받고 있는 고집적 SRAM 기반의 FPGA의 경우 우주용의 특수 공정을 적용하는 것이 제한적이어서 TMR 등의 오류 극복 기법을 탑재하여 우주용으로 사용되는 예가 많다[9]-[11].

TMR 구조 메모리는 그림 2에서와 같이 세 개의 메모리 셀에 동일한 데이터를 저장하고 출력 시에는 세 메모리 데이터를 다수결 회로를 거쳐 외부로 출력하는 방식이다. 다수결 회로는 세 메모리의 데이터를 받아서 이들 중 다수 값에 해당하는 출력을 내보낸다. 예를 들어 메모리 셀에 저장된 데이터가 각각 "0", "0", "1" 인 경우 다수 값 "0"이 결정 회로의 출력으로 나온다. 오류가 없는 상황에서는 TMR 메모리에 저장된 세 데이터는 동일한 값을 가진다. SEU 하나가 발생할 경우 이들 중 한 셀에서의 값이 변하지만 다수결 회로를 거치면서 오류가 복구되어 최종 출력에서는 오류가 없는 값을 얻을 수 있다.

하지만 메모리 셀에 발생한 SEU를 복구하지 않고 그대로 둘 경우 또 다른 SEU가 다른 메모리 셀에서 발생하면 두 셀에서의 데이터가 변경되어 다수결 회로만으로는 오류를 복구할 수 없게 된다. 따라서 TMR 메모리 셀에서 SEU에 의해 각 셀의 데이터가 변하는 경우 즉시 이를 복구하는 회로의 필요성이 대두된다. 즉 TMR 메모리 셀에 존재하는 SEU가 누적되지 않도록 제어할 필요가 있다. 본 논문에서는 비동기 머신 교정 제어를 이용하여 SEU가 누적되지 않고 발생 후 빠른 시간 내에 오류를 복구할 수 있는 폐루프 시스템의 구조를 제안한다.

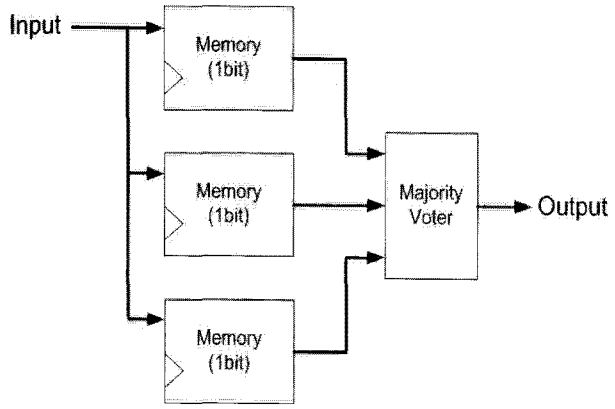


**그림 2** TMR 메모리 셀
**Fig. 2** TMR Memory Cell.

## 5. TMR 비동기 상태 피드백 제어

### 5.1 비동기 방식의 TMR 모델링

그림 3은 (1)의 유한 상태 머신 $\Sigma$으로 모델링된 TMR 메모리이다. 세 개의 메모리 셀을 0 또는 1로 표시하면 모두 8개의 조합이 나오므로 상태 집합은 X={000,001,...,111}이다. 그림에서 n은 모든 메모리 셀을 1로 "on" 시키는 입력이며 f는 모든 셀을 0으로 "off" 시키는 입력이다. n과 f는 제어기가 낼 수 있는 정상 입력이므로 $A_c = \{n, f\}$이다.

그림 3에서 $n_{ijk}$와 $f_{ijk}$는 각각 메모리 셀의 값을 반전시키는 외란 입력이며 i, j, k는 세 개의 TMR 메모리 셀을 가리킨다. 두 개 이상의 TMR 메모리 셀이 동시에 반전되는 외란 입력은 극히 드물게 발생하므로[9],[11] 본 논문에서 $n_{ijk}$와 $f_{ijk}$는 각각 하나의 메모리 셀 값이 on 되거나($n_{ijk}$) off되는($f_{ijk}$) 외란 입력이라고 정의된다. (즉 i, j, k 중 한 개만 1의 값을 가지며 나머지는 모두 0이다.) 예를 들어 시스템이 상태 000에 있을 때 외란 입력 $n_{001}$이 발생한다면 TMR 메모리의 세 번째 셀 값이 반전되는 고장이 일어났다는 뜻이므로 그림 3에서 볼 수 있듯이 상태 001로 천이된다. 또 시스템이 모든 메모리 셀 값이 1인 상태 111에 있을 때 $f_{001}$이 발생한다면 세 번째 셀 값이 1에서 0으로 반전되므로 시스템은 상태 110으로 옮겨간다. 외란 입력 집합 $A_d$을 다시 쓰면 아래와 같다.

$$A_d = \{n_{001}, n_{010}, n_{100}, f_{001}, f_{010}, f_{100}\}$$

그림 3의 비동기 머신에서 n과 f에 의한 상태 천이 외에 발생할 수 있는 모든 천이는 외란 입력에 의한 것들이라고 간주한다. 즉 비동기 상태 피드백 제어기는 $n_{ijk}$와 $f_{ijk}$의 외란 입력에 의해서 머신 $\Sigma$가 000과 111 이외의 상태로 천이하는 즉시 교정 동작을 작동해야 한다. $n_{ijk}$와 $f_{ijk}$ 입력이 없는 원하는 동작(모델) $\Sigma'$의 상태천이도를 그리면 그림 4와 같다. 외란 입력이 존재하지 않는 이상적인 환경이라면 TMR은 000과 111 등 두 개의 상태만 가질 것이다. 또한 전체 비트를 0이나1로 만들어주는 정상 입력 n과 f에 의해 한 상태에서 다른 상태로 천이가 발생하여 입력 값이 바뀌기 전까지는 현재의 안정 조합을 유지한다.

참고로 그림 3의 비동기 머신 $\Sigma$은 $\Sigma = \Sigma|_s$, 즉 그 자체로 stable-state 머신이다. 이것은 두 개 이상의 메모리 셀이 동시에 바뀌지 않는다는 가정의 결과이다. $\Sigma$ 안에 과도 상태가 존재하면 머신의 해석 및 제어기 설계는 더욱 복잡해진다. 과도 상태가 존재하는 TMR 메모리의 제어 문제는 추후 연구에서 다루기로 한다.

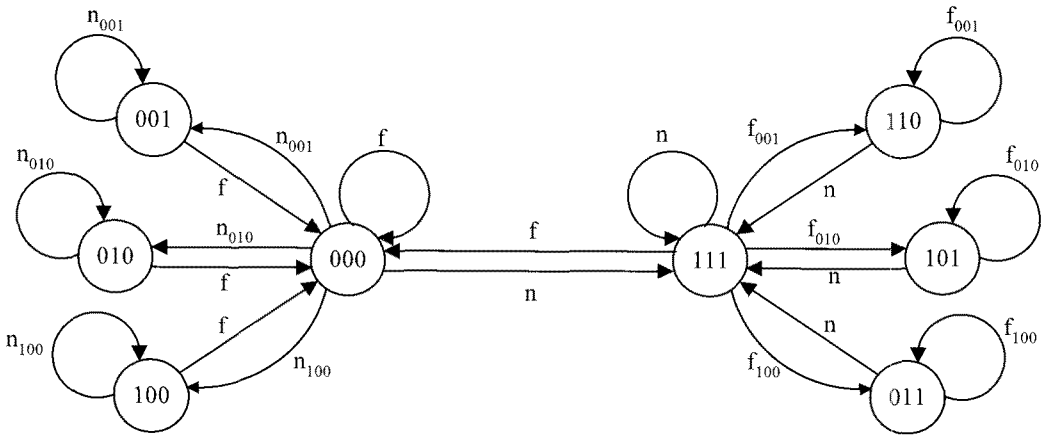


**그림 3** 외란 입력이 존재하는 TMR $\Sigma$의 상태천이도.
**Fig. 3** State flow diagram of TMR $\Sigma$ with disturbance input.

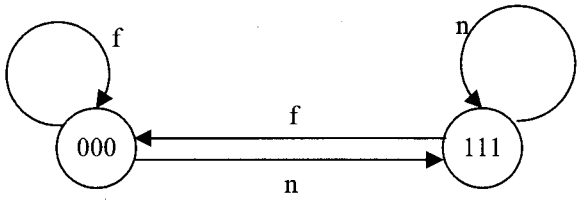


그림 4 TMR 이상 모델 Σ′의 상태천이도.
Fig. 4 State flow diagram of TMR model Σ′.

### 5.2 상태 피드백 제어기 설계

그림 3의 비동기 머신 Σ가 정상 상태에서 그림 4의 모델 Σ′이 내는 동작과 일치하게 하는 비동기 상태 피드백 제어기를 설계한다. 앞서 기술한 대로 피드백 제어기는 식 (4)의 유한 상태 머신 구조를 가진다. 제어기가 해결해야 하는 문제는 머신 Σ가 상태 000으로 되돌아가야 하는 것($n_{ijk}$ 발생)과 111로 되돌아가야 하는 것($f_{ijk}$ 발생) 등 두 가지로 분류할 수 있다.

#### 5.2.1 "000"으로의 복귀

TMR 비동기 머신이 안정 상태 000에 있을 때 외란 입력 $n_{001}$가 일으키는 원하지 않는 상태 천이를 복구하는 제어기 $C_1$을 먼저 설계한다. 초기 상태 $\xi_0$에 있던 $C_1$는 Σ가 000와 안정 조합을 이룰 때 상태 $\xi_0(x)$로 천이한다. 그림 3에서 000와 안정 조합을 이루는 입력 집합은 U(000)={f}이므로 $C_1$의 천이 함수 ϕ는 식 (5)로부터 아래와 같이 정의된다.

$$\begin{aligned}\phi(\xi_0,(x,v)) &= \xi_0 \qquad \forall\,(x,v)\notin(000,f)\\ \phi(\xi_0,(000,f)) &= \xi_0(x)\end{aligned} \tag{12}$$

제어기가 아직 아무 교정 동작을 하지 않으므로 식 (6)과 같이 출력 함수 값으로 외부 입력을 그대로 전달해준다.

$$\eta(\xi_0,(x,v)) = v \quad \forall\,(x,v)\in X\times A \tag{13}$$

외부 입력이 바뀌지 않는 한 Σ를 계속 상태 000에 머무르게 하기 위해서 $\xi_0(x)$에서 C는 정상 입력 f를 계속 머신에 넣어 준다.

$$\eta(\xi_0(x),(000,v)) = f \tag{14}$$

$\xi_0(x)$에서 제어기는 외란 입력 $n_{001}$의 발생을 기다린다. 앞서 기술했듯이 입력 $n_{001}$은 제어기가 관측할 수 없는 사건(event)이다. 하지만 그림 3에 나와 있는 대로 Σ는 $n_{001}$이 발생하면 001로 천이되므로 상태 피드백을 받은 제어기 $C_1$는 피드백 값의 변화를 보고 해당 고장이 발생했음을 감지한다. 본 논문에서 고려하는 모든 입력 외란은 이런 식으로 관측 가능하다(그림 3 참조).

상태 피드백이 000에서 001로 바뀌는 순간 제어기 $C_1$은 외란 입력 $n_{001}$이 일어났다는 사실을 감지하고 교정 동작을 실행한다. 그림 3에서 001에서 000으로 돌아가는 정상 입력 스트링 α는 α=f이다. α의 길이가 1이므로 제어기 $C_1$은 하나의 추가 상태 $\xi_1$만 정의하면 된다. $\xi_0(x)$에서 $\xi_1$로 가는 제어기의 동작을 식 (9)로부터 구현하면 아래와 같다.

$$\begin{aligned}\phi(\xi_0(x),(000,f)) &= \xi_0(x)\\ \phi(\xi_0(x),(000,v)) &= \xi_0 \qquad \forall\, v\notin f\\ \phi(\xi_0(x),(001,n_{001})) &= \xi_1\end{aligned} \tag{15}$$

다시 한 번 강조하면 외란 입력 $n_{001}$는 관측되지 않는 것이며, 상태 피드백 값이 001로 바뀌고 난 후 고장을 감지한 제어기가 즉시 위 동작을 작동한다.

$\xi_1$로 천이한 후 제어기는 Σ에 입력 스트링 α, 즉 제어입력 f를 전달한다. f를 받은 Σ는 그림 3에 나와 있는 대로 상태 000으로 복구된다. 000을 다시 상태 피드백으로 받은 $C_1$는 교정 동작이 완료되었음을 알고 외부 입력 값이 바뀌기 전까지 f를 머신에 계속 넣어주면서 상태 $\xi_1$에 머무른다. 식 (11)로부터 제어기의 동작을 설계하면 다음과 같다.

$$\begin{aligned}\phi(\xi_1,(000,n_{001})) &= \xi_1\\ \phi(\xi_1,(x,v)) &= \xi_0 \quad \forall\,(x,v)\neq(000,n_{001})\\ \eta(\xi_1,(x,v)) &= f \quad \forall\,(x,v)\in X\times A\end{aligned} \tag{16}$$

외란 입력 $n_{010}$과 $n_{100}$이 발생했을 때 머신을 원상태로 되돌리는 제어기를 각각 $C_2$와 $C_3$라 명명하면 이 교정 제어기들의 동작도 $C_1$의 동작과 거의 일치한다. $C_2$와 $C_3$에 대한 자세한 서술은 생략한다.

#### 5.2.2 "111"으로의 복귀

외란 입력 $f_{001}$, $f_{010}$, $f_{100}$의 발생에 대비해서 설계되는 교정 제어기를 각각 $C_4$, $C_5$, $C_6$라고 정의하자. 이 제어기들의 동작도 앞에서 구한 $C_1$과 유사하다. 상태 111에서 발생한 $f_{001}$이 만드는 천이 111→110을 되돌리는 제어기 $C_4$를 식 (12)~(16)을 응용하여 설계하면 다음과 같다. ($C_1$과 구분하기 위해 $C_4$의 상태를 $\psi_0$, $\psi_0(x)$, $\psi_1$ 등으로 표시한다.)

- $\psi_0$:

$$\begin{aligned}\phi(\psi_0,(x,v)) &= \psi_0 \qquad \forall\,(x,v)\notin(111,n)\\ \phi(\psi_0,(111,n)) &= \psi_0(x)\\ \eta(\psi_0,(x,v)) &= v \qquad \forall\,(x,v)\in X\times A\end{aligned} \tag{17}$$

- $\psi_0(x)$:

$$\begin{aligned}\phi(\psi_0(x),(111,n)) &= \psi_0(x)\\ \phi(\psi_0(x),(111,v)) &= \psi_0 \qquad \forall\, v\notin n\\ \phi(\psi_0(x),(110,f_{001})) &= \psi_1\\ \eta(\psi_0(x),(111,v)) &= n\end{aligned} \tag{18}$$

- $\psi_1$:

$$\begin{aligned}(\phi(\psi_1,(111,f_{001})) &= \psi_1\\ \phi(\psi_1,(x,v)) &= \psi_0 \quad \forall\,(x,v)\neq(111,f_{001})\\ \eta(\psi_1,(x,v)) &= n \quad \forall\,(x,v)\in X\times A\end{aligned} \tag{19}$$

#### 5.2.3 제어기 통합

전체 상태 피드백 제어기 C는 join 연산[5]을 사용하여 $C=C_1\vee C_2\vee\cdots\vee C_6$ 으로 통합된다. join 연산은 각 제어기의 초기 상태와 두 번째 상태를 각각 결합함으로써 이루어진다. 예를 들어 위에서 구한 $C_1$의 $\xi_0$와 $C_4$의 $\psi_0$, 그

리고 $\xi_0(x)$와 $\psi_0(x)$는 각각 하나의 상태로 결합된다. 식 (12)~(15)와 식 (17), (18)을 보면 이 상태들에서 $C_1$와 $C_4$ 사이에 아무런 겹침 동작이 없으므로 이러한 결합을 통한 제어기 통합은 잘 이루어진다는 것을 알 수 있다.

통합 제어기는 오류 발생 즉시 복구하여 TMR 메모리에 SEU가 누적되는 것을 막을 수 있고 오류 복구를 위한 별도의 전역 클럭을 필요로 하지 않는다는 장점이 있다. 따라서 메모리 오류 빈도가 아주 높은(우주 또는 원자로 내부) 환경에 잘 적용될 수 있을 뿐만 아니라 오류 복구 과정에 부가적인 시간이 소요되지 않으므로 고속 메모리에 적합하다. 비동기 제어기 구성을 위한 별도의 회로가 필요하지만 이를 간략화 하는 방법을 다음 연구에서 진행할 예정이다.

## 6. 모의실험 결과

본 논문에서 제안된 비동기 피드백 TMR 메모리 셀에 대한 소프트웨어 모의실험을 수행하였다. TMR 메모리 셀 및 비동기 제어기는 추후 FPGA로 실제 구현이 가능하도록 VHDL 코드를 이용하여 구성하였다. 또한 TMR 메모리 셀에서 발생할 수 있는 SEU를 모사하기 위하여 별도의 오류 발생기(Fault Injector)를 VHDL로 설계하여 부착하였다. 모의실험에 사용된 VHDL 컴파일러 및 합성기는 Altera의 QUARTUS® II(ver.7.1)이며, target FPGA는 Altera의 EP1C6Q240C8이다.

그림 5는 본 논문에서 설계된 비동기 제어기, TMR 메모리 셀, 그리고 오류 발생기가 결합된 시스템을 구동시킨 시뮬레이션 결과이다. 그림 5에서 "in_on", "in_off"는 입력 신호 n, f를 각각 가리키며 "f001"~"n100"은 여섯 개의 외란 입력 신호 $f_{001}$~$n_{100}$을 가리킨다. 또 "tmr_d1", "tmr_d2", "tmr_d3"은 TMR 메모리 비트 값이며 "state"는 TMR 메모리 셀 상태 변수 x이다. "ctl_on", "ctl_off"는 제어 입력 u로 들어가는 on 및 off 신호를 각각 말한다(그림 1 참조).

시스템은 먼저 20nsec에서 TMR 메모리 셀의 값을 on 시키는 입력 신호 in_on을 받은 후 상태 "1"을 유지하다가 80nsec에서 $f_{100}$ 오류가 발생하여 tmr_d1의 값이 "1"에서 "0"으로 변한다. tmr_d1의 값이 변했지만 정상 입력 신호의 변화가 감지되지 않으므로 외란 입력에 의한 SEU가 발생되었다는 사실이 관측된다. 제안된 비동기 제어기가 즉시 작동하여 98.04nsec에서 제어 신호 ctl_on을 생성하고 tmr_d1의 값을 "0"에서 "1"로 복구시키는 과정을 확인할 수 있다. 또 다수결 회로의 출력값(state)은 오류 발생에도 불구하고 "1"로 계속 유지된다. 이러한 복구 과정은 앞에서 기술했듯이 전역 클럭을 사용하는 기존 방법[10]보다 더 빠르며, 여러 개의 오류가 한꺼번에 일어나는 상황도 더 견실하게 대처할 수 있다.

본 모의실험에서는 시간 150nsec에서 TMR 메모리 셀의 값을 "0"로 설정한 후 210nsec에서 발생하는 $n_{010}$ 오류에 대한 교정 동작도 잘 수행됨을 보여준다(그림 5의 224.48nsec 부분 참조).

## 7. 결 론

본 논문에서는 비동기 머신을 위한 상태 피드백 제어를 응용한 새로운 TMR 메모리 구조를 제안하였다. 제안된 TMR 구조의 핵심은 전역 클럭에 의존해서 메모리 SEU 극복을 하지 않고 비동기 상태 피드백 제어를 이용하여 발생하는 SEU를 즉시 교정한다는 것이다. 본 제어 구조는 비동기적으로 발생하는 SEU의 검출 및 극복 면에서 동기식 기존 TMR 메모리보다 더 우수한 특성을 보인다. 현재 제안된 폐루프 시스템을 디지털 시스템 하드웨어로 구현하는 작업을 진행 중이며 추후 연구에서 발표할 예정이다.

## 참 고 문 헌

[1] S. Hauck, "Asynchronous design methodologies: an overview," Proceedings of the IEEE, vol. 83, no. 1, pp. 69-93, 1995.

[2] N. Nishimura, "Efficient asynchronous simulation of a

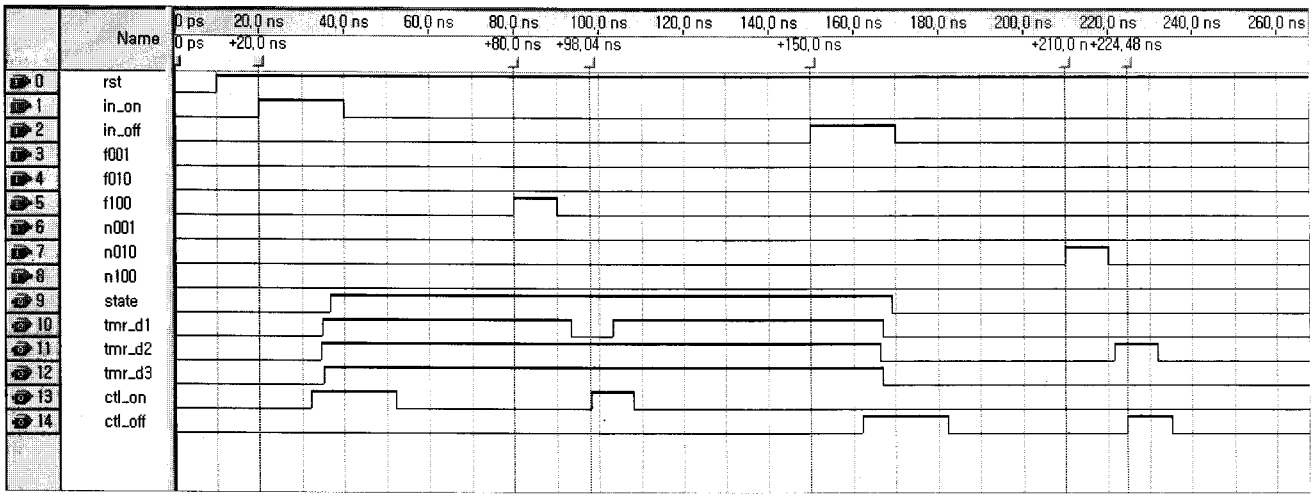


**그림 5** 소프트웨어 모의실험 결과.

**Fig. 5** Result of software simulation.

class of synchronous parallel algorithms," Journal of Computer and System Sciences, vol. 50, no. 1, pp. 98-113, 1995.
[3] A. Davis, B. Coates and K. Stevens, "The post office experience: designing a large asynchronous chip," in Proceeding of the 26th Hawaii International Conference on System Sciences, vol. 1, pp. 409-418, 1993.
[4] T. E. Murphy, X. Geng and J. Hammer, "On the control of asynchronous machines with races," IEEE Transactions on Automatic Control, vol. 48, no. 6, pp. 1073-1081, 2003.
[5] N. Venkatraman and J. Hammer, "On the control of asynchronous sequential machines with infinite cycles," International Journal of Control, vol. 79, no. 7, pp. 764-785, 2006.
[6] 양정민, "외란 입력이 존재하는 비동기 순차 머신의 모델 매칭", 전기학회논문지, 제57권, 제1호, pp. 109-116, 2008.
[7] L. Sterpone, M. Violante and S. Rezgui, "An analysis based on fault injection of hardening techniques for SRAM-Based FPGAs", IEEE Transactions on Nuclear Science, vol. 53, no. 4, pp. 2054-2059, 2006.
[8] Z. Kohavi, Switching and Finite Automata Theory (2nd ed.), New York: McGraw-Hill, 1978.
[9] 곽성우, 박홍영, "과학기술위성 1호 탑재 컴퓨터에서의 SEUs 극복을 위한 메모리 운용 및 해석", 항공우주학회지, 제3권, 제1호, pp. 98-105, 2004.
[10] L. Sterpone and M. Violante, "Analysis of the robustness of the TMR-architecture in SRAM-based FPGAs", IEEE Transactions on Nuclear Science, vol. 53, no. 5, pp. 1545-1549, 2005.
[11] P. L. Murray and D. VanBuren, "Single event effect mitigation in reconfigurable computers for space applications", Proceedings of IEEE Aerospace Conference, pp. 1-7, 2005.
[12] X. Geng and J. Hammer, "Input/output control of asynchronous sequential machines," IEEE Transactions on Automatic Control, vol. 50, no. 12, pp. 1956-1970, 2005.

## 저 자 소 개

**양 정 민 (楊 正 敏)**

1971년 3월 31일생. 1993년 2월 한국과학기술원 전기 및 전자공학과 졸업. 1995년 2월 한국과학기술원 전기 및 전자공학과 졸업(석사). 1999년 2월 한국과학기술원 전기 및 전자공학과 졸업(공박). 1999년 3월~2001년 2월 한국전자통신연구원 컴퓨터·소프트웨어연구소 선임연구원. 2001년 3월~현재 대구가톨릭대학교 전자공학과 부교수. 주관심분야: 비동기 순차 머신 제어, 걸음새 연구 등.
Tel : 053-850-2736
Fax : 053-850-2704
E-mail : jmyang@cu.ac.kr

**곽 성 우 (郭 成 祐)**

1970년 3월 10일생. 1993년 한국과학기술원 전기및전자공학과 졸업(학사) 1995년 동 대학원 전기및전자공학과 졸업(석사). 2000년 동 대학원 전기및전자공학과 졸업(공박). 2000년~2002년 인공위성연구센터 선임연구원, 연구교수. 2003년~현재 계명대 전자공학과 전임강사, 조교수.
Tel : 053-580-5926
Fax : 053-580-5165
E-mail : ksw@kmu.ac.kr