결함허용시스템의 디바이스 드라이버에 관한 연구

신덕호, 이종우, 황종규, 정의진
한국철도기술연구원

A Study on Device Driver for Fault-Tolerant System

Shin Duck Ho, Lee Jong Woo, Hwang Jong Kue, Jung Eui Jin
Korea Railroad Research Institute

Abstract - This paper show Device Driver for RTOS-FT Systems. In this paper, a definition of Device Driver is introduced which is used in RTOS-FT Systems. The structure of Device Driver is briefly divided into physical layer and logical layer. The specific characteristic of physical field and logical field which is discussed in this paper is suggested for the system which is satisfied with fault-tolerant theory.

1. 서 론

고도의 산업화와 통신기술의 발달 등으로 임베디드 시스템에 대한 수요가 급증하고 있다. 과거의 인명보호를 위한 결함허용 구조의 시스템 외에도 최근에는 인명과 직결되는 시스템의 신뢰성과 가용성에 대한 요구에 의해 결함허용구조를 갖는 시스템에 대한 연구가 활발하게 진행되고 있다. 과거의 논리회로와 순차회로의 하드웨어적 결함허용구조는 마이크로컨트롤러에 의존하는 복잡화된 현재의 시스템에 적용하기에는 어려움이 많게 되었고, 현재 시스템의 특성에 따라 결함을 검출하고 격리하기 위한 논리회로 등의 하드웨어와 소프트웨어가 유기적으로 동작해야만 결함을 극복할 수 있는 구조로 발전하고 있다. 따라서 신뢰할 수 있는 디바이스 드라이버에 관한 연구가 필요하게 되었다.

본 논문에서는 실시간 결함허용 임베디드 시스템에서 주로 사용하는 실시간 운영체제의 최하위 레벨인 디바이스 드라이버를 결함허용 구조로 작성하기 위한 기법에 대한 연구와 여분을 갖는 결함허용 구조의 하드웨어에 알맞은 디바이스 드라이버에 대해 연구하였다.

2. 본 론

2.1 디바이스 드라이버(Device Driver)

디바이스 드라이버는 어플리케이션 소프트웨어가 하드웨어를 제어하기 위한 다리역할을 하는 형태의 소프트웨어로써 완전한 소프트웨어라기보다는 하드웨어와 소프트웨어의 중간적인 부분인 펌웨어(Firm ware)이다. 디바이스 드라이버의 내용으로는 제어대상인 시스템의 구성에 따라 약간의 차이가 있으나 대부분의 실시간 결함허용시스템(RTOS-FTS)에서는 시스템의 부팅을 위한 부트스트랩, 시리얼 드라이버, 이더넷 드라이버, 타이머 드라이버 등을 이야기한다. 실시간 운영체제가 동작하기 위해서 운영체제의 타임카운터를 위해 사용되는 타이머와 어플리케이션의 업로드 및 브리지동작을 위한 이더넷, 그리고 컨트롤러의 상태를 보기 위한 시리얼 콘솔의 드라이버를 작성해 주는 부분이다. 일반적으로 이러한 역할을 하는 소프트웨어를 BSP(Board Support Package)라 부르기도 한다.

이러한 디바이스 드라이버를 결함허용 시스템에 설치하기 위해서는 결함허용 시스템에서 정의되는 몇 가지 논리를 알아야 한다.

2.2 실시간 결함허용시스템의 특징

결함허용 시스템의 특성은 아래와 같다.

- 여분(Redundancy)
  모든 시스템의 요소는 최소 한 개의 대기계를 가지고 있다.
- 결함검출(Fault detection)
  시스템의 요소에서 고장이 발생하면 시스템은 자동적으로 결함을 발견한다.
- 결함격리(Fault isolation)
  시스템은 자동적으로 고장난 요소를 격리한다.
- 극복(Recovery)
  단일 시스템의 요소가 고장나도 시스템은 중단 없이 동작을 계속 수행해야 한다. 두 번째 고장이 발생하면 시스템은 동작을 연속적으로 수행할 수 없다.
- 복구(Repair)
  시스템은 100ms이상의 서비스 중단 없이 복구된 요소의 재구성을 가능하게 해야한다.

2.3 결함허용 디바이스 드라이버(FT Device Driver)

결함허용시스템에 실시간 운영체제를 포팅하여 결함검출에 사용되는 하드웨어와 어플리케이션을 묶어주기 위해서는 결함허용 디바이스 드라이버의 구조와 특징에 대하여 이해해야 하므로, 본 절에서는 결함허용 디바이스 드라이버의 구조와 특징에 대해 논의한다.

2.3.1 결함허용 디바이스 드라이버의 구조

결함허용을 위한 디바이스 드라이버는 다음의 사항을 만족해야 한다.

- 하드웨어에 어떠한 결함이 발생하여도 시스템이 파괴 또는 잘못된 출력을 발생하지 않게 한다
- 결함이 발생하면 적절한 디바이스가 있는 경우에 여분의 디바이스로 스위칭 시키는 역할을 한다.

2.3.2 결함허용 디바이스 드라이버의 특징

결함허용 디바이스 드라이버가 일반 드라이버와 구별되는 가장 큰 특징은 아래의 두 가지이다.

- Hardening (경화성)
- Switchover (절체)

드라이버의 경화성은 결함의 검출과 격리를 지원하여 디바이스의 결함이 정상적으로 동작중인 영역으로 확산되지 않게 한다. 하지만 상위레벨 서비스는 디바이스를 더 이상 사용할 수 없게 되므로 계속 진행되지 못한다.

드라이버의 절체는 결함의 극복을 지원한다. 상위레벨 서비스는 디바이스가 하나의 디바이스에서 다른 디바이

스로 전환되므로 연속적으로 진행된다. 결함허용 시스템의 드라이버는 논리디바이스에 의해 물리적인 여분의 디바이스를 구성한다. 따라서 이러한 절체의 주체는 디바이스 드라이버의 논리계층이다. 여분의 부분은 시스템이 단일의 디바이스로 구성된 것처럼 응용프로그램에서는 인식한다. 따라서 드라이버는 여분 또는 대기계로 절체하여 하드웨어적인 결함을 극복한다. 드라이버는 디바이스의 상태를 점검하여 데몬을 전환하고 디바이스의 사용이 요구될 때 사용할 수 있도록 공급하는 것을 드라이버 내부의 감시모듈로써 지원한다.

2.4 디바이스 드라이버의 경화성(Hardening)

드라이버의 경화성 처리는 디바이스의 하드웨어결함이 시스템 전체로 확산되는 것을 방지한다. 디바이스 드라이버의 경화작업은 결함허용 달성을 위한 가장 중요하며 기초적인 작업이다. 결함이 자주 발생하는 I/O 부 시스템(sub-system)의 드라이버에만 이러한 경화성이 요구되는 것이 아니라 시스템 전체의 디바이스 드라이버에 적용되어야 한다.

드라이버의 경화를 위해서는 디바이스의 정상상태와 비정상상태의 동작데이터를 모두 조사하여 결함이 발생 즉시 검출 및 차단되도록 코딩하는 것을 의미한다. 이러한 디바이스 드라이버를 작성하기 위해서는 결함검출을 위해 동작하는 결함검출 회로 또는 결함의 확산을 방지하기 위한 결함격리 회로의 입출력과 연관되어 동작해야 한다. 위와 같은 동작은 디바이스 드라이버 계층에서 코딩되므로 일반 사용자의 응용프로그램 작성에는 아무영향도 주지 않는다.

- 결함의 검출(Fault Detection)

드라이버는 신뢰된 코어에서 동작한다. 신뢰된 코어의 내부 데이터는 안전한 것으로 간주된다.
I/O 부 시스템을 통한 데이터의 입력은 I/O버스의 상태뿐만이 아니라 다음의 사항들을 검출할 수 있다.

a. 버스관련 에러
b. 데이터의 불량
c. 프로토콜 에러

결함허용 시스템에서 결함의 검출을 위해서는 하드웨어적인 검출 기법을 많이 사용하고 있다. 이러한 하드웨어에 의해 시스템에서 결함검출이 발생되면 하드웨어적인 결함발생 신호를 이용하여 시스템을 재시작 또는 영구적으로 차단(shutdown)시키는 신호로써 시스템을 설계하고 있다. 하지만 시스템의 재시작이나 차단을 요구하지 않는 결함이 발생할 경우에는 이러한 결함의 발생을 I/O로 받아들여서 소프트웨어적으로 처리하는 구조를 갖는다. 이러한 처리를 디바이스 드라이버에서 수행한다.

상용 실시간 운영체제에서 버스에러상태에서의 I/O 접근은 드라이버 작성자의 고의적 버스에러 발생을 제외하면 커널패닉을 발생시킨다. 또한 응용프로그램에서 하드웨어의 잘못된 접근에 의해 버스에러가 발생한다. 패닉에러가 발생되면 시스템을 안전 측으로 동작하기 위해 시스템의 하드웨어 구조에 맞는 안전 측 차단 코드를 디바이스 드라이버에 삽입해야 한다.

데이터의 불량은 버스에러 또는 버스 상에서 검출할 수 있는 고장의 발생에 의해 일어날 수 있으나, 모든 데이터의 불량이 드라이버에 치명적이지는 않다. 드라이버 설계자는 드라이버가 데이터 불량에 민감한 부분과 그렇지 않은 부분을 분류하여 드라이버에 독립적으로 코딩한다. 예를 들어 드라이버가 I/O 컨트롤러로부터 데이터를 카피하여 데이터의 불량을 체크하는 것과 같이 드라이버에 치명적이지 않는 동작에서는 데이터의 오염이 드라이버에 영향을 주지는 않는다. 하지만 드라이버가 I/O 컨트롤러로부터 포인터를 읽어들이는 경우에는 데이터를 사용 전에 반드시 체크해야 한다.

프로토콜의 에러는 드라이버가 I/O 부 시스템에 비정상의 동작을 하게 하거나 동작이 실패했을 경우에 발생해야 한다. 시스템의 모든 동작을 예견하여 시스템의 동작에 따른 궤환 값을 허용되는 값과 허용되지 않는 값으로 분리하여, 반환된 값이 허용되지 않는 값을 갖는 경우 시스템과의 인터페이스에 문제가 있다고 판단하여 결함이 발생된 부위를 차단하도록 구성되어야 한다.

- 결함의 격리(Fault Isolation)

드라이버가 결함을 검지하면 디바이스에서 결함이 재발생되지 않도록 디바이스를 사용할 수 없게 하고, 다른 디바이스들이 차단에 의해 자원을 잠식하지 않도록 처리한다. 디바이스를 사용할 수 없게 하는 방법 중 추천되는 방법은 해당 디바이스가 포함되어 있는 모듈의 전원을 차단하는 것이다. 그러므로 디바이스를 전기적으로 격리시킨다. 이러한 전원의 차단을 결정하는 출력을 발생함으로써 스스로 전원을 차단하는 형태이다. 이때 전원차단에 해당하는 출력을 모듈화 시켜서 응용프로그램에서 함수를 호출함으로써 모듈의 전원이 차단되도록 디바이스 드라이버에 등록시켜 준다.

만약 디바이스가 다중계 모듈이고 다른 모듈이 동작중이라면 해당하는 모듈의 전원을 차단한다. 따라서 다중계 모듈의 디바이스 드라이버를 제작하는 경우에는 결함의 격리 기능을 포함하기 위해 드라이버를 변경하지 않아도 된다.

2.5 디바이스 드라이버의 절체기능(switchover)

절체는 디바이스를 여분으로 준비된 다른 디바이스로 넘겨주는 역할을 이야기한다. 이는 여분의 하드웨어를 이용하여 결함을 회복하는 것이다. 절체는 아래와 같은 상황에서 발생한다.

- 디바이스가 고장나서 대기중인 디바이스의 드라이버를 가동한다.
- 시스템의 동작 중에 디바이스의 결함검사를 위한 주기적인 절체가 발생한다.
- 동작중인 디바이스에 off-line이 요구되면 드라이버는 대기중의 디바이스로 절체된다.

결함이 검출되고 절체되는 동안 데이터는 소실된다. 이러한 특징은 결함허용 I/O 부분시스템이 결함 소멸시에도 데이터의 전송을 완전히 신뢰하지 않는다는 것을 의미한다.

이러한 데이터의 소실을 방지하기 위해서 하드웨어 구조에 맞는 다중계 통신 데이터 백업기능을 디바이스 드라이버에 내장한다. 예를 들어 공유메모리를 사용하여 데이터를 공유하는 경우에는 메모리의 디바이스 드라이버 부분에 다중계 모듈간의 데이터 공유 등의 알고리즘을 삽입하여, 시스템 절체 시에 시스템이 데이터를 복구하는 별도의 루틴을 수행하지 않고, 시스템을 즉시 가동시키는 대기 이중계(Hot-Standby Sparing System) 방식으로 구성한다.

2.6 물리계층과 논리계층

물리 디바이스는 논리 드라이버들에 의해 단일 시스템의 디바이스처럼 응용프로그램에서 사용하도록 구성한다. 디바이스 드라이버는 논리계층과 물리계층 디바이스에 대한 드라이버로 이분화 할 수 있다. 사용자가 드라이버에 대한 기능적인 분리를 하지 않더라도 드라이버는 물리적 계층과 논리적 계층을 갖는다.

VxWorks나 VRTX의 BSP를 예로 들면, 디바이스의 레지스터 번지를 직접 지정하고 레지스터를 세팅하는 파일들이 물리계층에 해당하며, 지정된 변수를 이용하여 드라이버의 동작특성을 코딩한 부분이 논리계층에 해당한다.

■ 논리계층의 구조

드라이버의 논리계층은 물리계층에서 지정된 레지스터 값을 사용하여 결함검출회로의 입력을 받아서 처리하는 기능과, 시스템의 반환 값이 허용되지 않는 부분으로 선언된 값이 반환되는 경우, 결함의 심각성에 따른 차단과 경고발생 등의 역할을 수행하는 부분이다.

■ 물리계층의 구조

드라이버의 물리계층은 I/O 부 시스템으로부터의 입력 값의 기본적인 처리와, 출력된 명령이 올바르게 동작하였는지 등의 반환 값 검사 등을 수행하는 부분으로 대부분의 코드가 어셈블러 레벨의 함수지정과 레지스터 세팅, 그리고 출력과 출력데이터 궤환 값의 비교 등으로 구성된다.

3. 결 론

본 논문에서는 실시간 결함허용 시스템에서 주로 사용하는 디바이스 드라이버에 대하여 결함허용 드라이버에 대하여 정의하고, 결함허용 디바이스 드라이버의 내부를 계층으로 나누어 각 계층의 특성을 분석하였다. 이러한 구조로 디바이스 드라이버를 사용하여 결함허용 시스템을 구현하면 하드웨어의 구조가 변경되어도, 논리계층에서의 작업만으로 시스템에 실시간 운영체제를 포팅할 수 있으며 응용프로그램 작성 시 별다른 고려 없이도 실시간 결함허용 기능을 수행할 수 있다.

[참 고 문 헌]

[1] SAFETY-CRITICAL COMPUTER SYSTEMS
Neil Storey, England Warwick, 1996

[2] Computers and Communications, 1998. ISCC '98. Proceedings. Third IEEE Symposium on , 1998

[3] Fault isolation in a class of nonlinear uncertain input-output systems
Xiaodong Zhang; Polycarpou, M.; Parisini, T.
American Control Conference, 2001. Proceedings of the 2001