2005년도 대한전기학회 하계학술대회 논문집 2005. 7. 18 - 20

정형기법을 이용한 PLC RTOS 검증

최창호, 송승환, 윤동화, 황성재
포스콘 기술연구소

PLC Real Time OS Verification & Validation in Formal Methods

Chang-Ho Choi , Seung-Hwan Song , Dong-Hwa Yun , Sung-Jae Hwang
POSCON R&D Center

Abstract - Currently, Programmable Logic Contorller(PLC) uses Real Time Operation System(RTOS) as basic OS. RTOS executes defined results as to defined time. General features of RTOS emphasize the priority in each task, high-speed process of external interrupt, task scheduling, synchronization in task, the limitation of memory capacity. For safety critical placement, PLC software needs Verification and Validation(V&V). For example, nuclear power plant. In this paper, PLC RTOS is verified by formal methods. Particularly, formal method V&V uses verification tool called 'STATEMATE', and shows the results.

1. 서 론

실시간 운영체제(Real Time Operating System, RTOS)는 실시간으로 작업을 처리해야 하는 시스템에 필요한 일종의 관리자이다. 실시간 시스템이 정해진 시간 내에 정해진 결과를 이끌어야 하는 시스템이라고 정의되므로 이를 관리하는 RTOS 역시 deterministic한 연산으로 그 동작시간이 예측 가능하여야 한다. 이러한 RTOS의 동작은 시스템의 부하를 덜기 위해 가능한 간단하게 작성되어 고속의 응답시간을 보여야 한다. RTOS의 일반적인 특징으로는 태스크 간의 우선순위가 강조되고, 외부 인터럽트의 처리를 고속화, 태스크 스케줄링 및 동기화가 중요하다. 그리고, RTOS가 주로 사용되는 곳이 임베디드 시스템이므로, 성격상 RTOS의 크기도 작아야 한다. [1]

현재 (주)포스콘에서는 원자력발전소에 사용 가능한 안전등급 PLC를 개발하고 있다. 기존의 PLC는 따로 설계 규격이 없었지만, 원자력발전소에 적용되는 안전등급 PLC는 안전성, 투명성, 신뢰성 등이 필수적인 원전 설계 규격을 적용하여 설계된다. 하드웨어는 'CLASS 1E', 소프트웨어는 'Safety Critical'에 준한 설계가 필수적이며, 이에 대한 검증도 심도있게 실행되어야 한다.[2]

정형기법(formal method)은 복잡한 소프트웨어 시스템이 지니는 제반 문제들을 해결하기 위해 제안된 기법으로서 크게 정형명세(formal specification)와 정형검증(formal verification)으로 구분된다. 정형명세는 소프트웨어 개발 초기단계에서 개발자가 모든 요구 사항들을 생략하지 않고 명확하게 명세하도록 유도함으로써 소프트웨어의 안전성을 크게 향상시킬 수 있는 기법으로 인정 받고 있다. 정형검증 기법은 이러한 정형명세를 기반으로 하여 모델체킹(model checking)이나 정리증명(theorem proving) 등의 검증을 통해서 정형명세된 소프트웨어의 안전성을 증명하는 기법이다. [3][4]

본 논문에서는 당사에세 개발중인 안전등급 PLC(POSAFE-Q)에 사용되는 RTOS를 정형기법 구현을 위한 Tool인 'STATEMATE'를 이용해서 원전 설계 규격에 맞춰 설계하고 검증하는 과정을 설명한다.

2. 본 론

2.1 정형기법을 이용한 안전등급 PLC RTOS 설계

그림 1은 앞으로 보여질 그림들의 전체구조를 나타낸

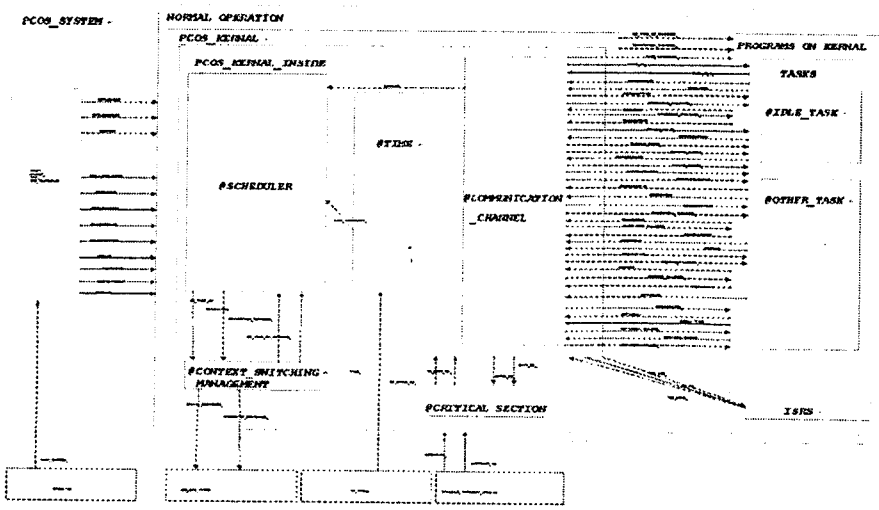

그림 1 안전등급 PLC Real Time Kernel의 정형기법 설계

것으로, 안전등급 PLC의 전체적인 RTOS를 정형기법을 적용하여 표현하였다.

Internal Activity인 'SCHEDULER', 'TIME', 'OTHER_TASK' 'COMMUNICATION_CHANNEL' 등은 각각 내부의 Sub-chart를 포함하는 형태로 구성되며, Internal Activity 간의 통신을 위하여 많은 제어 정보와 데이터 정보가 사용된다. 예를 들어 태스크와 세마포어 대기명령과의 관계는, 현재 수행중인 태스크의 우선순위를 확인하는 작업을 하기 위해 'TASK_ID_'라는 데이터 정보 흐름이 쓰였고, 세마포어 대기명령과 태스크의 상태를 변화시키기 위해 'OSSEMPEND"와 'OSSEMWAITING"라는 제어 정보 흐름을 사용하였다. 여기서 'OSSEMWAITING'은 시뮬레이션을 위해 추가되는 항목이다.[4]

2.1.1 Schedule 의 구현

그림 2는 Scheduler의 정형기법 설계를 나타낸 그림으로, 그림 1에서 '@SCHEDULER' 블록의 Sub-chart를 나타낸 것이다.

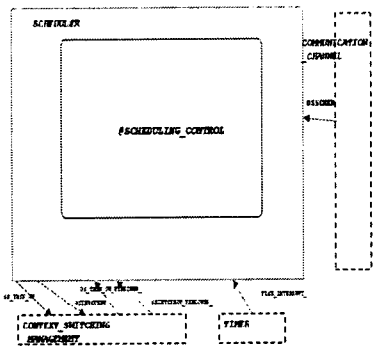

그림 2 Scheduler의 정형기법 설계-1

그림 3은 Scheduler의 기능을 태스크 Schedule과 인터럽트 Schedule로 구분하여 나타낸 그림으로, 그림 2에서 '@SCHEDUKING_CONTROL' 블록의 Sub-chart를 나타낸 것이다.

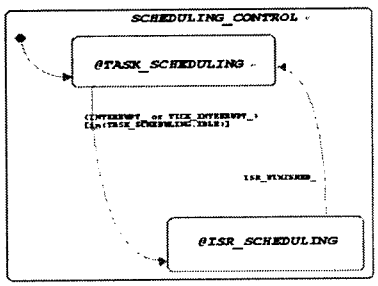

그림 3 Scheduler의 정형기법 설계-2

그림 2에서 'COMMUNICATION_CHANNEL'이 그림 1에서는 Internal Activity로 사용되었지만, 그림 2에서는 External Activity로 작성되어 있다. 이는 다른 Activity chart 들이 그림 2의 중심인 'SCHEDULER'의 관점에서는 External Activity, 즉 외부조건으로 보여진다는 것을 나타낸다.

2.1.2 Scheduler의 구현

RTOS의 Scheduler는 크게 태스크를 처리하는 부분과 인터럽트를 처리하는 부분으로 나눌 수 있다. 이 절에서는 태스크 처리하는 부분을 정형기법으로 명세한 것을 보여준다.[5]

그림 4는 RTOS의 기능 중 태스크 처리 부분으로 그림 3에서 '@TASK_SCHEDULER' 블록의 Sub-chart를 나타낸 것이다.

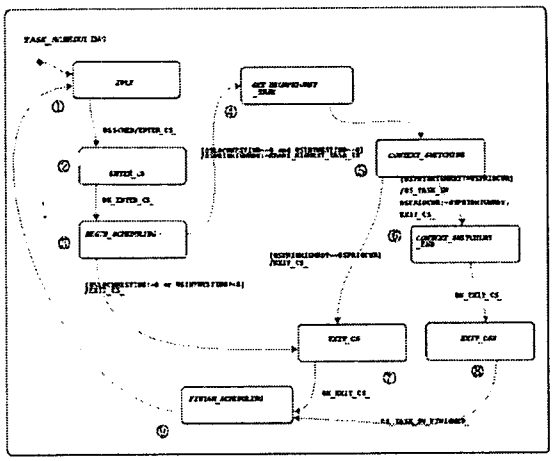

그림 4 Task Scheduling의 정형기법 설계

앞의 그림 2에서 제어 정보 흐름으로 'OSSCHED'가 쓰였는데, 이는 broadcast 적으로 사용되어 최종적으로는 그림 4의 입력으로 영향을 주게 된다. 이 부분은 실제로 작성되어지는 소프트웨어에서 어떤 함수가 'OSSched()'라는 함수를 호출할 때, 그림 4에서는 'OSSCHED'가 실행이 되서 다음 연결로 넘어가는 것을 나타낸다.

2.2 메시지 메일박스의 구현

그림 5는 RTOS에서 메시지 메일박스 전달에 관계되는 함수인 OSMboxPost()를 flowchart로 나타낸 것이다.

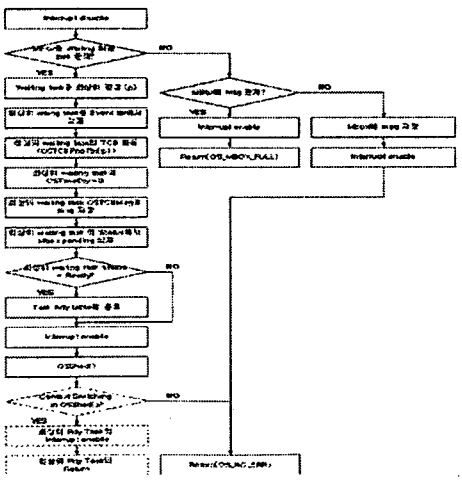

그림 5 메시지 메일박스 전달의 Flowchart 설계

그림 6은 정형기법으로 OSMboxPost()의 기능을 구현하여 나타낸 차트이다.

그림 5에서 보는 것처럼 OSMboxPost()함수가 사용되면 그에 대한 결과는 세 가지가 있다. 첫 번째 결과는 메시지를 기다리고 있는(=pending) 태스크가 이미 존재하는 경우로 그 태스크에게 바로 메시지를 전달하는 것이다. 그림 6의 'SOMEONE_WAITING'를 거치는 흐름이 이 부분을 나타낸다. 두 번째 결과는 메시지를 기다리고 있는 태스크가 존재하지 않는 경우로 메시지 메일

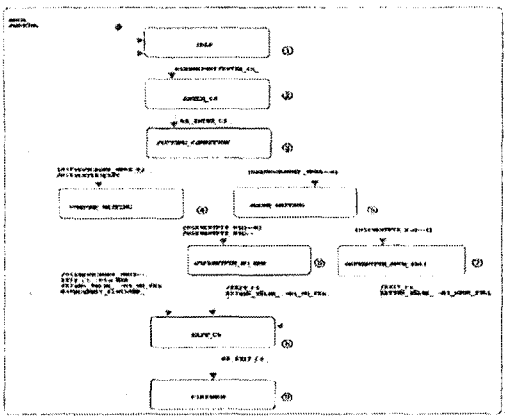

그림 6 메일박스 전달의 정형기법 설계

박스를 확인한다. 그러나 그곳에 이미 다른 메시지가 들어있다면 새로운 메시지는 버려지게 된다. 그림 6의 'OSEVENTPTR_MBOX_FULL'를 거치는 흐름이 이 부분을 나타낸다. 세 번째 결과는 메시지를 기다리고 있는 태스크가 존재하지 않는 경우로 메시지 메일박스를 확인하는 부분까지는 두 번째 결과와 같다. 그러나 다른 메시지가 들어있지 않는다면 메시지를 저장시키는 점이 다르다. 그림 6의 'OSEVENTPTR_NO_ERR"를 거치는 흐름이 이 부분을 나타낸다.

2.3 시뮬레이션

정형기법으로 RTOS를 설계한 다음에 그 표현이 정상적으로 구현되었는지 또는 비정상적인 동작이 존재하는지를 확인하기 위하여 시뮬레이션 작업이 필요하다.

안전등급 PLC RTOS의 태스크 상태는 크게 DORMANT, READY, RUNNING, WAITING, ISR의 5가지 상태로 구분되어진다.[6] 본 논문에서는 이 태스크의 상태를 바탕으로 시뮬레이션을 구현하였다.

2.3.1 시뮬레이션 실행

그림 7은 태스크의 5가지 상태를 정형기법으로 구현한 차트이다.

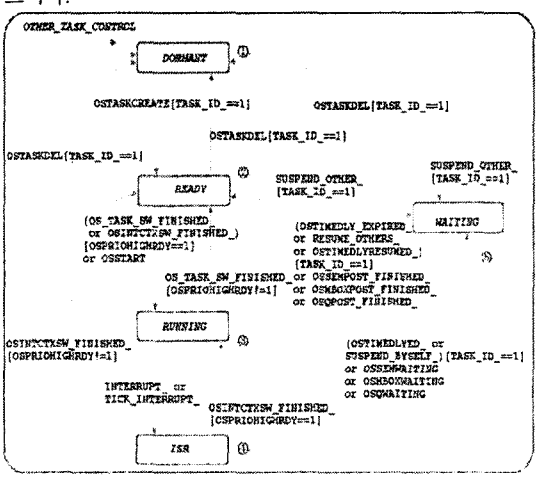

그림 7 태스크 상태의 정형기법 설계

그림 8은 태스크의 상태가 그림 7번의 'DORMANT'에서 'READY'상태로 변할 때 이와 관련된 제어 흐름 정보와 데이터 흐름 정보를 나열한 시뮬레이션 모니터이다.

그림 9는 태스크의 상태가 그림 7번의 'READY'에서 'RUNNING' 상태로 변할 때 이와 관련된 제어 흐름 정보와 데이터 흐름정보를 나열한 시뮬레이션 모니터이다.

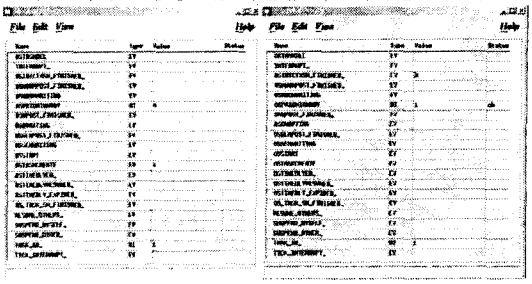

그림 8 'DORMANT'에서 'READY' 상태로 변할 때 시뮬레이션 모니터

그림 9 'READY'에서 'RUNNING'로 변할 때 시뮬레이션 모니터

위의 그림 8과 9는 그림 7을 바탕으로 시뮬레이션을 실행할 때, 특정한 조건 변화에 따른 결과를 가지고 정보들이 어떻게 변하는지를 표현한 시뮬레이션 모니터 상태를 나타낸다. 그림 9는 ② 'READY' 상태에서 ③ 'RUNNING'로 넘어가기 위한 정보들을 나타낸 것으로서 그림 8과 그림 9를 비교해보면 'OSTASKCREATE'의 제어 정보 흐름은 사라졌고, 제어 정보 흐름 'OSINTCTXSW_FINISHED_'와 데이터 정보 흐름 'OSPRIOHIGHRDY'의 데이터가 변한 것을 알 수 있다. 이 과정을 거치면 그림 7의 ③ 'RUNNING'에 실행을 알리는 표시가 위치하고 'OSINTCTXSW_FINISHED_'와 'OSSTART'의 정보는 한 번 실행 후 사라지게 된다.

2.3.2 시뮬레이션 결과

태스크 상태들의 시뮬레이션을 바탕으로 실질적인 안전등급 PLC의 RTOS를 정형기법으로 구현함으로써, 태스크들이 어떠한 상태로 변화하는지 확인 할 수 있었다. 그 결과로서 오직 'RUNNING' 상태를 가질 수 있는 태스크는 오직 한 태스크라는 것과 인터럽트 발생시 태스크는 ISR 처리 상태로 간다는 것 등을 확인 할 수 있었다.

3. 결 론

본 논문에서는 정형기법을 이용하여 원자력 발전소에 적용될 예정인 안전등급 PLC RTOS의 검증작업을 수행하였고, 각종 상태에 대한 조건들을 변경하면서 시뮬레이션을 실행하였다. 이러한 과정을 통하여, 안전등급 PLC에 검증을 위한 직접적인 수정 작업 없이 논리적인 검증작업의 시뮬레이션을 수행하여 각종 상황에 대한 변화추이를 손쉽게 파악할 수 있었다. 또한, 다양한 경우의 수에 따른 RTOS의 동작 상태를 전통적인 PLC 분야의 비효율적인 방식에 비해 용이하게 검증 할 수 있었다. 최종적으로 정형기법을 이용하여 당사의 안전등급 PLC의 성능, 신뢰성과 안전성 및 투명성을 명시적으로 확보할 수 있었다.

[참 고 문 헌]

[1] 포스콘 기술연구소 PLC part, "pCOS 명세서", 2002
[2] 한국원자력연구소, "원자력발전소 안전계통에 적용하기 위한 PLC 일반 요건 및 규격", 2005
[3] 유준범 외 3인, "원전 분야를 위한 KAIST 소프트웨어 개발 방법론", 제29회 한국정보과학회 추계학술대회
[4] 김창희 외 3인, "정형기법을 이용한 KNICS 디지털 원자로 보호계통 소프트웨어 요구사항명세 개발", 제3회 원자력학회-전기학회 공동주최 계측제어기술 Workshop
[5] I-Logix, "STATEMATE MAGNUM Manual", 2002
[6] Jean J Labrosse, "MicroC/OS-2", CMP books, 2002