자동차 전자제어 시스템의 타이밍 검증 방안

ChronVAL/ChronSIM: A Tool Suite for Timing Verification of Automotive Applications

자동차 시스템 타이밍 문제 증가

자동차 전자제어 시스템은 갈수록 복잡해지고 있다. 이에 따라 자동차 안전 요구사항과 타이밍에 관한 제약사항들도 증가하고 있다. 현재 개발 방식에서 타이밍 검증은 주로 구현이 완료되는 시점에 이뤄진다. 개발 후반부에서 발견된 설계의 실수는 많은 추가 비용과 수정 시간을 필요로 한다. 최근 독일 OEM에서는 엔진 밸브의 타이밍을 조정하는 두 ECU 간 동기화 문제로 엔진이 정지해 자체 리콜을 했다. 결국 엄청난 금액의 손실 비용을 감수해야 했고 ECU를 재프로그래밍했다. 물론 기존에도 타이밍 검증 활동은 여러 형태로 진행됐다. 하지만 애플리케이션의 규모가 커짐에 따라 기존의 제한된 방식의 타이밍 검증은 한계에 부딪치게 됐다. 개발 후반부가 아닌 개발 초기부터 정확한 타이밍 검증과 관리가 필요한 이유다.

최근 자동차에 내장된 시스템은 하드 리얼타임(Hard Real-time)의 제약사항을 가지고 전체 시스템의 정확성을 보장하고 있다. 이 중 단대단 응답시간(End-to-end response time)은 반드시 결정돼야 하고, 또 준수돼야 한다. 기준치를 초과한 응답시간은 전체 시스템의 성능 저하를 가져올 수 있을 뿐만 아니라, 자동차의 불안정한 동작을 야기할 수 있다. 또한 다른 제약사항들도 가능한 모든 상황의 조건 하에서 만족돼야 한다.

개발 초기 타이밍 검증의 필요성

자동차 소프트웨어 설계 시 잘못된 의사결정에 따른 결함 수정은 개발 비용에 큰 타격을 줄 수 있다. 이러한 의사결정들은 주로 개발 초기에 일어나지만, 이로 인한 문제는 구현 이후에 발견된다. 실제로 대부분의 타이밍 문제들은 전체 개발 주기에서 가장 후반부인 구현 단계 또는 시스템 통합 단계에서 발견된다. 기존의 타이밍 검증 수단으로는 정형적

도전과제(Challenges)
자동차 시스템의 규모가 커지고 복잡해짐에 따라 타이밍 이슈가 증가하고 있으며, 이에 대한 대응이 필요하다.

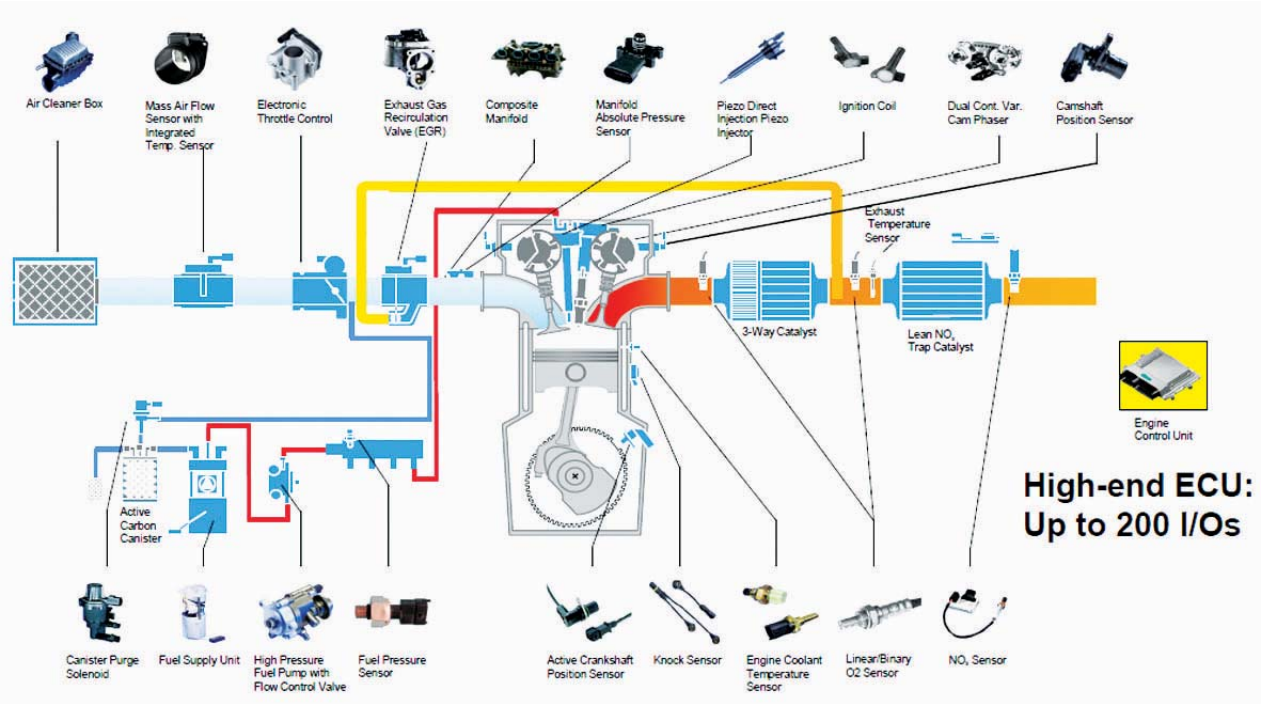
해결방안(Solutions)
개발 초기부터 정확한 타이밍 검증을 수행하고, 분석적 기법과 시뮬레이션 기법이 적절히 활용돼야 한다.

장점(Benefits)
타이밍 설계 문제로 인한 수정 비용 및 시간의 낭비를 크게 줄이고, 시스템 내 다른 모듈 및 업체 간의 효과적인 협업 대응을 할 수 있다.

자동차 시스템의 타이밍 분석을 수행하기 위해서는 기본적으로 분석적인 기법과 시뮬레이션 기법이 모두 필요하다. 이 글은 타이밍 검증을 하기 위한 기법 또는 도구에는 어떠한 요구사항들이 필요한지 소개한다. 또 독일 인크론(INCHRON GmbH)의 타이밍 검증 도구인 ChronSIM/ChronVAL(INCHRON Tool-suite)이 타이밍 검증에 필요한 요구사항을 어떻게 지원하며, 자동차 시스템에 대한 정확한 타이밍 분석을 가능케 하는지 소개한다.

글 | **사우센 안시**(Saoussen Anssi), Continental Automotive France
칼스텐 알베르스(Karsten Albers), **마리아스 되르펠**(Matthias Dörfel), Incrhon GmbH
세바스티엔 게라드(Sébastien Gérard), CEA LIST

역 | MDS테크놀로지 **박 지 웅** 주임 〈jwpark@mdstec.com〉



〈출처: Denis Claraz, "Real Time Architecture of Engine Systems", Conti Software Conference 2011〉

[그림 1] 타이밍 문제와 직결된 자동차 시스템 구조 예

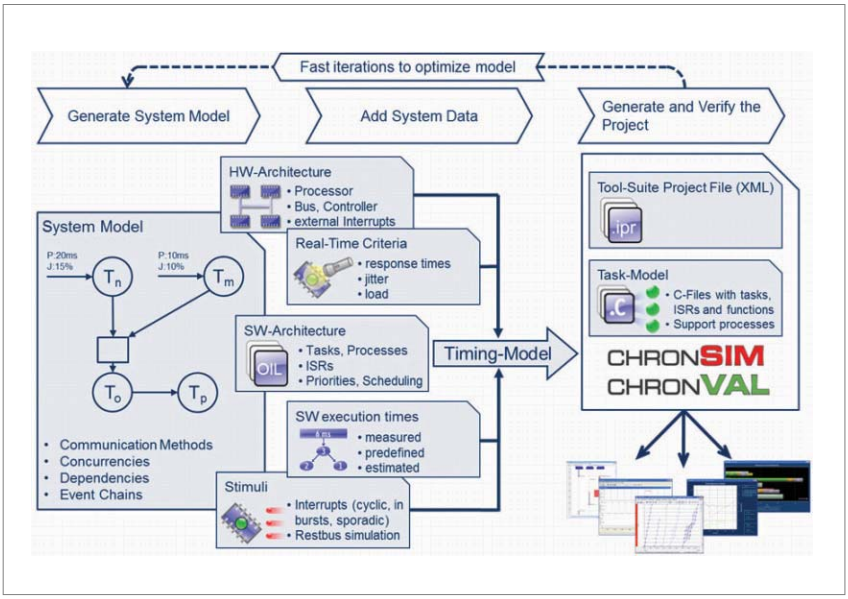
이거나 시스템 차원의 분석 방법이 쓰이지 않고, 상대적으로 제한적인 방식인 측정과 테스팅 기법이 쓰였다. 실제로 국내에서 대부분의 타이밍 검증은 오실로스코프로 응답시간을 측정하는 수준으로 행해지고 있다. 이러한 이유로 자동차 시스템의 혁신적이고 복잡한 새로운 기능들이 비용 효율적인 측면에서 제대로 구현되지 못하고 있다. 궁극적으로 개발 초기부터 정확한 타이밍 분석을 지원하는 기술과 도구들의 필요성이 명백해지고 있다. 이러한 기술과 도구들을 통해 시스템의 타이밍 동작을 예측하고, 설계 시 일어날 수 있는 잠재적인 취약점을 개발 초기에 교정할 수 있게 된다.

타이밍 검증 기법

자동차 시스템의 타이밍 분석을 수행하기 위해서는 기본적으로 분석적인 기법과 시뮬레이션 기법이 모두 필요하다. 본 글에서는 타이밍 검증을 하기 위한 기법 또는 도구에는 어떠한 요구사항들이 필요한지 소개하고자 한다. 또 독일 인크론(INCHRON GmbH)의 타이밍 검증 도구인 ChronSIM/ChronVAL (INCHRON Tool-suite)이 타이밍 검증에 필요한 요구사항을 어떻게 지원하며, 자동차 시스템에 대한 정확한 타이밍 분석을 가능케 하는지 소개하고자 한다.

[표 1] 타이밍 검증 기법/도구의 요구사항

식별자	요구사항
REQ1	타이밍 검증은 프로세서의 점유율을 결정할 수 있는 기술을 가져야 한다.
REQ2	타이밍 검증은 태스크와 함수의 데드라인을 명시해야 한다.
REQ3	타이밍 검증은 함수 또는 태스크의 활성화 시점과 관련한 지터(Jitter)를 명시해야 한다.
REQ4	타이밍 검증은 단대단(End-to-end) 타이밍 제약사항을 명시해야 한다.
REQ5	타이밍 검증은 특정 데드라인(Deadline)을 만족하는지 아닌지 입증할 수 있는 기술을 가져야 한다.
REQ6	타이밍 검증은 단대단(End-to-end) 제약사항을 만족하는지 입증하는 기술을 가져야 한다.
REQ7	타이밍 검증은 주기적, 산발적, 그리고 단발적인 성질의 이벤트와 태스크를 명시해야 한다.
REQ8	타이밍 검증은 주기적인 이벤트와 태스크를 위해 특정 각에 의한 반복(Angular Recurrence)을 명시해야 한다(엔진 동기화 관련 태스크).
REQ9	타이밍 검증은 다수의 ECU들과 통신 버스들에 대한 분산 시스템을 쉽게 모의할 수 있어야 한다.
REQ10	타이밍 검증은 멀티 프로세서 시스템을 분석하는 기술을 가져야 한다.
REQ11	타이밍 검증은 CAN, LIN, FlexRay 통신을 위한 분석 기술을 가져야 한다.
REQ12	타이밍 검증은 프로세서의 오버헤드와 네트워크 오버헤드를 고려해야 한다.
REQ13	타이밍 검증은 연결된 태스크 간의 의존성을 모의할 수 있어야 한다.
REQ14	타이밍 검증은 동일한 우선순위의 태스크들 간의 스케줄링으로 FIFO 알고리즘을 차선으로 사용할 수 있어야 한다.
REQ15	타이밍 검증은 선점형(Preemptive), 협동형(Cooperative) 태스크와 인터럽트를 명시해야 한다.
REQ16	타이밍 검증은 고정적, 가변적인 태스크 오프셋 값을 명시하고 분석할 수 있어야 한다.



[그림 2] 타이밍 검증 과정 흐름도

보다 정확한 타이밍 검증을 위한 기법 또는 도구가 만족해야 할 요구사항은 표처럼 요약될 수 있다. 결국 이 특징들이 자동차 시스템의 타이밍 분석 요소의 대상들을 식별해 주고, 도출된 요구사항들은 타이밍 검증 도구들에 의해 입증돼야 한다. 이해를 돕기 위해 각각의 요구사항에 대해 식별자를 부여했다.

타이밍 검증 솔루션 인크론(INCHRON Tool-suite)

인크론(INCHRON Tool-suite)은 복잡한 임베디드 시스템의 상세한 타이밍 이슈를 분석하는 도구로 모델 기반의 시뮬레이션 도구(ChronSIM)와 분석적 기법을 사용하는 도구(ChronVAL)를 지원한다. 개발 프로세스 상의 여러 단계에서 적용이 가능하도록 몇 가지 정보만으로 이뤄진 추상적 레벨의 모델링부터 시작해 완전한 C코드 레벨까지 시뮬레이션 할 수 있다.

ChronSIM 모델링

시뮬레이터인 ChronSIM은 태스크, 인터럽트의 활성화 시점들과 분산, 그리고 실행시간을 선택한 스케줄링 정책에 적용해 구체적인 태스크 스케줄링과 타이밍 실행을 결정할 수 있게 해준다. 모델링은 순수한 모델 기반 또는 C코드와의 혼용을 통해 가능하다. 이에 따라 모델의 추상화 레벨은 유동적으로 조정 가능하다. 기본적인 ChronSIM 유저 인터페이스에 들어가면 트리 구조로 전체적인 시스템 구성 요소를 모델로 보여준다. 모델링은 리소스(CPU), 스케줄러, 태스크/인터럽트, Runnable(AUTOSAR 해당)들을 추가하는 것으로 시작된다. 태스크와 인터럽트 사이의 활성화 관계는 둘 사이의 커넥션을 구성하는 것으로 가능하다. 각각의 시스템 리소스는 기본적인 스케줄러를 가지고, 스케줄러의 스케줄링 방식은 고정 우선순위, TDMA, OSEK처럼 선택 가능하다. 그리고 기본 스케줄러 아래 서브 스케줄러를 단계적으로 구성하는 것이 가능하

다. 예를 들어, 최상위 레벨의 스케줄러를 TDMA로 구성하고, 하위 슬롯 중 하나를 고정 우선순위 스케줄링으로 구성할 수 있다.

태스크와 인터럽트의 활성화 시점에 대한 설정은 별도의 화면에서 구성한다(이러한 활성화 시점의 분포를 '자극 – Stimulation'이라 칭함). 자극의 형태는 몇 가지 종류의 패턴이 있다. 그 예로 주기적 패턴과 버스트(Burst) 패턴이 있다. 주기적 패턴의 자극을 구성하는 요소로는 반복 시간, 유입 가능한 이벤트의 수, 지터(Jitter) 값, 초기 오프셋 등이 있다. 시뮬레이션을 위해 지터 값을 지정할 수 있을 뿐 아니라, 임의의 확률에 의해 편차를 발생시킬 수 있다. 버스트 패턴은 일시적으로 많은 자극이 발생하는 경우를 시뮬레이션 할 수 있도록 한다.

시뮬레이션 모델에는 C코드를 추가하는 것이 가능하다. 태스크는 자신의 동작을 더 상세히 보여 줄 수 있는 C코드 함수를 가질 수 있다. 시뮬레이션을 하는 동안 C코드는 호스트 상에서 실행된다. 하지만 타이밍 동작은 실제로 타깃 플랫폼 위에서 실행된다. ChronSIM에서는 C코드에 삽입 가능한 특수 매크로 명령(DELAY)을 제공함으로써 타깃의 타이밍 동작을 실제 타깃과 최대한 유사하게 모의할 수 있도록 한다. 측정한 실행시간은 타이밍 이슈와 관련된 영향과 더불어 에러를 재현하고 원인을 식별하는데 도움을 준다.

실행 시간 모의

실행 시간을 모의하는 방법으로는 몇 가지 옵션이 존재 한다. 첫째, GUI 기반으로만 모델링된 태스크와 ISR(Interrupt Service Routine)에 대해 최소(Best-case), 최대(Worst-case), 임의확률의 실행시간을 선택할 수 있다. 둘째, C코드로 된 함수가 제공될 경우 ChronEST라는 도구가 코드의 실행시간을 예측해 계산한다. 예측 실행시간은 기본적인 블록 레벨로 분석돼, 하드웨어의 아키텍처, 컴파일러, 운영체제 사양을 고려해 계산된다. 마지막으로, C코드 내에 특수 매크로 명령(DELAY)을 추가하는 방법이다. 매크로는 시간 값, 시간 단위, 임의 확률(균등 분포, 표준 분포) 옵션을 지정해 삽입할 수 있다. 이 매크로를 삽입해 시뮬레이션을 수행함으로써 실행시간을 모의할 수 있다. 하나의 함수 내에서 여러 개의 매크로를 삽입 가능하며, 매크로를 조건문 내에 삽입해 조건적으로 실행시간을 다르게 할 수도 있으며, 시간 값을 지정할 때 상수가 아닌 변수 사용이 가능해 가변적으로 실행시간을 줄 수도 있다.

이러한 세 가지 모델링 옵션은 하나의 시뮬레이션 모델에서 혼용해 사용할 수 있다. 예를 들어, 전체적인 시스템의 구조는 간단한 GUI 기반의 모델링으로 구성한다. 그리고 중요한 함수 한두 가지만 실제 C코드를 사용해 구성한다. 초기 개발 단계일 경우, 간단한 GUI 기반의 모델에서 시작해 개발이 점차 진행되면서 단계적으로 구체적인 모델과 C코드로 교체해 나가는 방법도 있다. 초기에 C코드를 구성해 내부에 특수 매크로(DELAY)만 사용해 구성한 후, 개발이 진행되면서 실제 코드로 교체해 나가는 방법도 사용할 수 있다.

추가적으로 실행시간을 모의하는 매크로는 이벤트 송수신, CAN, FlexRay 상에서의 메시지 송수신, 이벤트 체인의 추적 등에도 사용할 수 있다. 시뮬레이션을 수행할 때 각기 다른 시스템 리소스를 위한 클록을 모델링할 수 있다. 그래서 자극 설정과 실행시간은 절대시간에 반드시 맞추지 않고, 설정된 클록 기반의 로컬 시간에 맞춰 모의가 가능하다. 클록의 기준 값은 서로 다른 클록 간의 오프셋, 드리프트 값 등에 서로 연관돼 있다. 예를 들어, 실제 FlexRay 버스는 자신의 시간 기준을 가지며, 이것은 버스에 연결된 각기 다른 시간 기준들을 가진 ECU들로부터 발생된다. 이러한 시스템을 정확히 모델링하는 것이 가능하며, 이를 통해 시간 기준이 불완전하게 동기화돼 발생하는 문제들을 발견할 수 있다.

시뮬레이션 결과

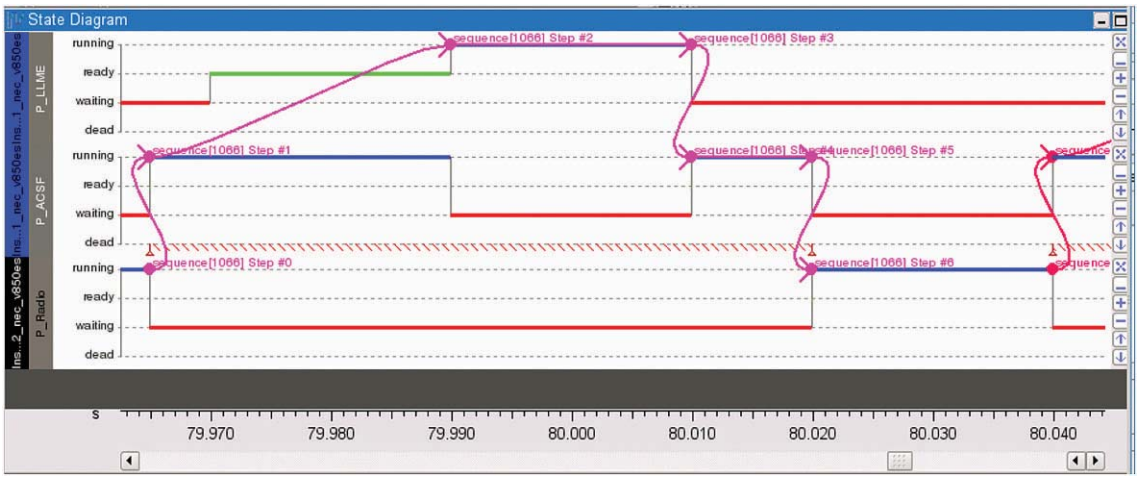
타이밍 시뮬레이션 결과는 Trace 파일 형태로 저장된다. 이것은 다양하고 유용한 다이어그램들을 만들어 내는 소스가 된다.

상태 다이어그램(State Diagram)은 각각의 태스크 상태를 시간에 따라 보여준다. 사용자는 이를 통해 특정 시점에서 어떠한 태스크가 실행 중이며, 실행 가능한 상태인가를 알 수 있다. 스케줄링 방식과 선택한 우선순위가 주는 영향을 확인할 수 있다. 태스크의 우선순위, 스케줄링 방식, 할당된 리소스의 변경에 따른 영향을 쉽게 파악할 수 있도록 해준다.

부하 다이어그램(Load Diagram)은 전체적인 시스템의 부하를 보여주고, 선택 가능한 시간 범위와 입상도(Granularity)에 따라 부하를 분석할 수 있다. 사용자는 리소스 점유율이 고르게 분포됐는지, 특정 태스크에 높은 점유율(Full)이 발생하면서 동시에 다른 부분에 낮은 점유율(Idle)이 발생하는지를 알아낼 수 있다. 이를 통해 사용자는 비교적 낮은 점유율을 가지는 태스크를 이동시켜 응답시간을 줄일 수 있다.

시뮬레이션을 진행하며 생성 가능한 히스토그램을 통해 시뮬레이션 결과를 기반으로 한 통계 정보를 확인할 수 있다. 히스토그램의 대상 값으로 여러 종류를 지정 가능하다. 예로 응답시간 또는 지터 값, 서로 다른 태스크의 특정 이벤트 사이의 시간 등을 지정해 분포도를 출력할 수 있다.

이벤트 체인에 대한 결과는 이벤트 체인 다이어그램이 별도로 보여준다. 각각의 이벤트 체인은 데이터가 시스템을 거치며 처리되는 단계를 보여준다. 이벤트 체인 다이어그램을 통해 데이터의 중복 사용 또는 손실을 발견할 수 있다. 이벤트 체인은 또한 상태 다이어그램 내에서도 시연돼 태스크의 상태와 함께 확인할 수 있다. 이벤트 체인은 히스토그램에서도 분포를 보여준다. 히스토그램에서 통계하는 값의 예로 이벤트 체인의 첫 단계와 끝 단계 사



[그림 3] 상태 다이어그램을 사용한 타이밍 문제 검출

이의 시간 간격, 그리고 다른 이벤트 체인의 동일한 단계 사이의 시간 간격이 있다. 그 밖에 다른 다이어그램은 스택 다이어그램, 중첩 함수 호출 다이어그램, 이벤트 체인 다이어그램 등이 있다.

ChronVAL

지난 10여년 동안 실시간 시스템을 위한 타이밍 검증을 위해 많은 분석적 기법의 접근 방법이 개발돼 왔다. 한 예로 스케줄링 가능성 분석 기법은 개발 초기 단계에서 비기능적인 요소를 분석하는데 유용하다. 단일 선점 방식 프로세서에 대한 정확한 스케줄링 가능성 시험 방법은 레하츠키(Lehoczky)에 의해 처음으로 소개됐다. 스케줄링을 수행할 대상은 각자의 주기를 가지고 있고, 실행시간이 주기 내에 마치도록 설계된 태스크들의 집단으로 이뤄져 있다. 이 시험에서는 RMA(Rate Monotonic Algorithm)을 사용해 어떤 스케줄링 집단이 가장 적합한지 결정한다. 이후 어드슬리(Audsley)에 의해 RMA 기법이 보완돼 DMA(Deadline Monotonic Algorithm)를 개발했고, 여러 사람에 의해 지속적으로 보완되고 발전돼 왔다. 이 시험 방법은 향후 팔렌시아(Palencia)와 곤잘레스(Gonzalez)에 의해 분산 시스템으로 확장됐다.

또 다른 실시간 시스템 분석 기법으로 RTC(Real-time Calculus)가 있다. 이 기법은 티엘(Thiele) 외 다수에 의해 개발됐고, Network Calculus라는 기본 이론에서 비롯됐다. 이 기법은 분석에 요구되는 복잡한 함수와 기본적인 함수들이 혼합 정의된, 실행 간격을 기반으로 하는 곡선들이 사용된다. 이 곡선들에 의해 실시간 속성들이 모델링되고, 이 곡선들을 기반으로 한 대수법들을 제공한다. RTC는 이 곡선들을 통해 복잡한 시뮬레이션에 대한 보다 정확한 모델링을 할 수 있는 장점을 지닌다. 이는 강력한 대수법을 기반으로 하기 때문에 주기와 실행 간격만으로 계산한 결과인 RMA 기법보다 더 정확하다. 단점은 비교적 긴 계산 실행시간이다.

ChronVAL은 RTC 기법과 더불어 실제 시스템에 대한 부가적인 보완책을 기반으로 한 분석을 수행한다. 예로 협동 스케줄링을 위한 분석 기법과 오프셋 분석 방법 등을 지원한다. ChronVAL은 RTC 분석 결과를 기반으로 각 태스크의 최대(Worst-case), 최소(Best-case) 응답시간을 계산해 주며, 이를 데드라인과 비교

INCHRON GmbH와 Real-time Congress 개최

INCHRON GmbH는 2003년 설립된 독일 회사로 타이밍 검증 분야에서 활발한 활동을 하고 있으며, INCHRON Tool-suite를 개발해 2005년 파일럿 프로젝트를 수행했다. INCHRON GmbH는 2009년 이래로 Real-time Congress를 개최해 타이밍 이슈에 대한 기술 정보와 솔루션 및 사례를 공유하고, 여기에는 유수의 OEM과 개발 솔루션 업체 들이 참여하고 있다.

해 준다. 또한 ChronVAL은 이벤트 스펙트럼 그래프를 통해 각 태스크에 대한 이용 가능량과 잔여 가용량을 보여준다. 이벤트 스펙트럼의 최소 잔여 가용량을 무한대 범위에서 출력하면 바로 리소스에 대한 점유율을 산정할 수 있으며, 이를 최종 보고서에 출력할 수 있다.

타이밍 검증 요구사항 평가 (INCHRON Tool-suite)

다음 표는 인크론(INCHRON Tool-suite)이 타이밍 검증 기능의 요구사항을 어떻게 지원하는지 나타낸다.

결론

이 글에서는 자동차 시스템의 타이밍 검증을 위한 주요한 요구사항들을 소개하고, 이 요구사항을 만족하기 위한 타이밍 검증 도구인 인크론(INCHRON Tool-suite)을 소개했다. 타이밍 검증 방법에는 크게 분석적인 기법과 시뮬레이션 기법이 존재한다. 자동차와 같은 복잡한 실시간 시스템에서는 분석적인 기법들이 높은 품질의 입증 방법임에도 불구하고 제한된 가용성으로 인해 적용이 어렵다. 반면, 시뮬레이션 기법은 크고 복잡한 설계에 적용하기 용이하다. 따라서 정확한 타이밍 검증을 위해서는 적용 대상에 따라 최적의 결과를 얻을 수 있는 기법을 선택적으로 적용해야 한다. 예를 들어, 한 모듈에 대해 안전 관점의 최대 응답시간(Worst-case Response time)을 구하고자 할 때는 분석적인 계산법을 사용하고, 전체 시스템 레벨에서의 평균 응답시간을 산정하기 위해서는 시뮬레이션 기법을 사용한다. 인크론에서 제공하는 ChronVAL과 ChronSIM 도구를 적재적소에 적용하면 최대한 정확한 타이밍 검증 결과를 얻어낼 수 있다. **AE**

[그림 4] 분석적 타이밍 검증 기법을 사용한 응답시간 예측

[표 2] 타이밍 검증 요구사항에 대한 인크론 지원 기능

식별자	지원 기능
REQ1	ChronVAL – 이벤트 스펙트럼을 통한 점유율 분석 기능 지원 ChronSIM – 부하 다이어그램(Load Diagram) 분석 기능 지원
REQ2	ChronVAL/SIM – 데드라인 요구사항을 태스크/ISR에 삽입 가능
REQ3	ChronVAL/SIM – 자극(Stimulation) 개념을 사용하여 지터(Jitter) 모의 가능
REQ4	ChronVAL/SIM – 요구사항으로 사용자 정의의 단대단(end-to-end) 제약사항 삽입 가능
REQ5	ChronVAL/SIM – 입력한 데드라인 요구사항과 실제 결과를 실시간으로 비교 가능
REQ6	ChronVAL/SIM – 입력한 단대단 요구사항과 실제 결과를 실시간으로 비교 가능
REQ7	ChronVAL/SIM – 태스크의 주기적, 산발적 활성화를 지터와 함께 모의 가능, 단발성 태스크는 반복주기가 없는 활성화 설정을 통해 가능
REQ8	ChronVAL – 매 클록마다 주기적인 이벤트 활성화 모의 가능. 다른 이벤트에 의해 활성화되는 이벤트 모의 가능. ChronSIM – 클록 모델링 가능. C 코드 매크로 활용으로 복잡한 시나리오 모델링
REQ9	ChronVAL/SIM – 분산 시스템 모델링 지원. 다수의 ECU와 Bus 모의 지원.
REQ10	ChronVAL/SIM – 멀티 프로세서/코어 시스템 분석 지원
REQ11	ChronVAL/SIM – CAN, FlexRay, LIN, 이더넷 스위치 등 여러 통신 인터페이스 모의 지원.
REQ12	ChronVAL/SIM – 각 리소스에 대한 컨텍스트 스위치 오버헤드와 네트워크 오버헤드 시연.
REQ13	ChronVAL – 태스크 내 커넥션 필드 삽입으로 태스크 간 연결성 모의 ChronSIM – 이벤트 체인 구성으로 데이터 연결성 모의
REQ14	ChronVAL/SIM – 서로 다른 스케줄링 방식으로 계층적인 스케줄러 구성 가능, FIFO 방식 지원
REQ15	ChronVAL/SIM – 선점형, 협동형 태스크 동작 모의 지원
REQ16	ChronVAL/SIM – 자극 설정에서 태스크의 오프셋 값 지정 가능(고정, 가변, 임의 가능)