노이즈 환경 하에서 태스크와 메시지 스케줄링

Scheduling ofTasks and Messages under Noise Environment

김 형 육*, 윤 건, 박 홍 성
(Hyoung Yuk Kim, Gun Yoon, and Hong Seong Park)

Abstract : Nowadays, control systems consist of smart sensors, smart actuators, and controllers connected via fieldbus. Some devices such as motors in plant environments generate high degrees of EMI or noise. This noise may cause communication errors and make the successful transmission of data longer. Therefore, the noise condition has to be considered at the design of a reliable control system based on a network. This paper presents a scheduling method of task and message to guarantee the given end-to-end constraints under noise environments. A noise model with multi-sources of noise is used, and the analysis method of message's response time is presented when the noise model is applied to CAN (Controller Area Network). Two kinds of noise models are applied to an example system, and the effect to each control loop's end-to-end response time is analyzed.
We believe that the proposed method help system designers design the control system guaranteeing its requirements under noise environment.

Keywords : real-time, scheduling, fieldbus, noise

I. 서론

현재의 제어시스템은 내장형 프로세스가 장착된 스마트 센서 및 구동기가 Profibus, FIP, CAN, LonWorks 등과 같은 필드버스를 통해 제어알고리즘을 수행하는 제어노드 또는 제어기와 연결되는 구조가 일반적이다. 그러므로 점대점 연결 방식을 통해 구성되던 제어시스템과는 달리 센서, 구동기, 제어기가 서로 분산되어질 수 있기 때문에 센서에서 제어기, 제어기에서 구동기로의 데이터 전송이 네트웍 특성에 따라 임의 지연시간을 갖게 된다. 또한 노드의 특성 및 노드에서 수행되는 태스크 수와 태스크 간의 관계에 따라 각 노드에서 수행되는 태스크의 기동에서 수행완료까지의 지연시간 또한 일정치 않다. 그러므로 네트웍 지연 및 각 노드에서의 태스크 수행지연 등을 고려하여 시스템을 설계하여야 한다[1-3].

이러한 문제를 해결하기 위한 연구로서, [4]에서는 어떠한 태스크도 두 개 이상의 메시지를 받지 않는다는 가정 하에서 TDMA를 사용하는 분산 제어시스템을 위한 휴리스틱(heuristic) 스케줄링 방법이 제안되었다. 또한, 태스크 기반의 스케줄링 방법[5]을 CAN으로 구성되는 분산 제어시스템으로 확장한 연구[6]에서는 여러 가지 소거법을 통하여 분산 제어 시스템을 위한 스케줄링 방법을 제안하였다. [7]은 클러스터링 알고리즘과 유전적 알고리즘을 사용하여 CSMA/CA와 TDMA 방식의 네트웍으로 구성되는 분산 시스템을 위한 스케줄링 방식을 제안하였다[8] 는 분산 제어 시스템의 메시지 우선순위 설정 방법과 태스크 주기할당 방법을 제안하고 이를 통해 여러 개의 제어루프로 구성되는 일반적인 분산 제어 시스템을 위한 태스크와 메시지 스케줄링 방법을 제안하였다.

* 책임저자(Corresponding Author)
논문접수 : 2003. 5. 20., 채택확정 : 2004. 1. 8.
김형육, 윤건, 박홍성: 강원대학교 전기전자정보통신공학부
(petrus@control.kangwon.ac.kr/yoongun@control.kangwon.ac.kr/
hspark@cc.kangwon.ac.kr)
※ 본 논문은 2003 년도 강원대학교 BK21 사업에 의하여 지원 되었음.

그러나 이러한 연구들은 노이즈로 인한 통신에러를 전혀 고려하지 않고 제안되었으므로 노이즈 환경 하의 시스템에 적용 시에는 문제가 있을 수 있다. 대부분의 분산 제어시스템은 노이즈를 생성하는 대용량 모터 등의 기기들이 작동하는 공장환경에 설치되어 운용되게 되므로 운영환경으로부터의 강한 전자계 간섭(EMI)에 노출되게 된다. 이러한 간섭은 통신매체 상에 노이즈를 발생시켜 데이터 전송 시 에러를 유발하고 노드간 통신이 지연되게 된다[9,10]. 노드간 통신지연은 시스템의 성능 저하 및 치명적인 오류를 야기시킬 수 있다. 일반적으로 에러로 인해 메시지가 유실되거나 메시지에 에러가 발생되면 해당 메시지를 재전송함으로써 해당 메시지를 복구하게 된다. 이러한 재전송을 통해 정확한 메시지 전송이 완료될 때까지의 메시지 응답시간 지연이 발생하며, 결국 해당 메시지를 기다리고 있는 태스크는 정확한 메시지가 수신될 때까지 기다려야 하므로 수행지연이 발생하게 된다. 또한 재전송으로 인한 네트웍 부하 증가가 해당 네트웍을 경유하는 다른 메시지들의 응답시간에도 영향을 미치게 된다. 다시 말하자면, 노이즈로 인한 통신에러가 제어루프의 양극단 응답시간을 지연시켜 데드라인을 만족시키지 못하게 되는 제어 시스템의 치명적인 오류를 야기할 수 있다. 그러므로 노이즈 환경에서 운용되는 분산 제어시스템 설계 시에 시스템의 실시간 성능을 보장하기 위해 노이즈로 인한 통신에러를 고려하는 새로운 방법이 필요하다.

전송에러로 인한 메시지 응답시간 지연에 대한 연구로서, [11]에서는 산발적으로 발생하는 에러와 비트에러율을 이용한 간단한 에러 모델을 통해 CAN 상의 메시지 응답시간 영향을 분석하였다. 또한 [11]에서 제안한 방법을 노이즈 모델링과 여러 개의 노이즈를 동시에 고려할 수 있는 방법으로 확장시켜 메시지의 응답시간 영향을 분석한 연구[9]가 있다. 그러나 이러한 연구들은 단순히 통신에러로 인한 메시지 응답시간에만 초점을 두어 분석하였으며 태스크와 메시지 간의 선행 관계와 통신에러로 인해 발생하는 태스크의 수행시

간 지연 영향이나 센서에서 최종 구동기까지의 양극단 응답 시간 지연 영향에 대해서는 고려하지 않았다.

본 논문에서는 노이즈 환경 하에서 운영되는 제어시스템을 위한 태스크와 메시지 스케줄링 방법을 제안하고 예제 시스템에 적용하여 그 유효성을 검증한다. 제안된 방법은 [8]에서 제안된 태스크와 메시지 스케줄링 방법을 노이즈로 인한 통신 에러를 고려하도록 확장한다. 이를 위해 여러 개의 노이즈 소스로 구성되는 노이즈 모델링 방법을 사용하고, CAN에서 이러한 노이즈에 대한 응답시간 분석방법에 대하여 서술한다. 2 개의 노이즈 모델에 대해 CAN으로 구성된 분산 제어시스템의 양극단 응답시간에 미치는 영향을 분석하여 통해 노이즈로 인한 전송지연이 시스템 스케줄링에 큰 영향을 미치는 것을 서술한다. 이를 통해 노이즈 조건에 따라 네트웍 속도와 같은 시스템 요구사양 조정이 필요함을 알 수 있으며 제안된 스케줄링 방법을 통해 시스템 설계자는 시스템 설치 및 운영 환경에 존재하는 노이즈 간섭 하에서도 요구되는 시스템 성능을 만족하는 시스템을 설계할 수 있게 된다.

2절에서는 노이즈 모델 및 CAN에서 노이즈로 인한 통신 지연시간 분석방법에 대해 서술하고 3절에서는 대상 시스템에 노이즈를 고려한 태스크와 메시지 스케줄링 방법을 적용한다. 끝으로 4절에서 결론을 맺는다.

II. 노이즈 모델 및 통신지연시간

1. 노이즈로 인한 문제

대부분의 제어시스템은 대용량 발전기 또는 대용량 모터 등이 동작하는 발전소나 공장환경에 설치되어 운용된다. 이러한 환경에서 발생하는 강한 전자계 간섭(EMI)이 통신 선로에 영향을 미쳐 통신 에러를 유발하고 노드간 통신을 지연시켜 시스템에 치명적인 오류를 야기시킬 수 있다. 이러한 영향이 그림 1에 잘 나타나있다. 그림 1은 센서노드 1, 센서노드 2, 제어기 1, 구동노드 1으로 구성되는 제어루프1과 센서노드 1, 센서노드3, 제어기 2, 구동노드 2로 구성되는 제어루프 2에 대한 타이밍도를 나타내고 있다. 센서노드, 구동노드, 제어기에는 각각 하나의 센서태스크, 구동태스크, 제어태스크가 수행된다. 그림 1의 모든 태스크는 동시에 기동이 되어 수행되며 제어태스크와 구동태스크는 태스크의 입력 데이터인 메시지가 수신될 때까지 기다 리게 된다. 이때 메시지의 우선순위는 센서태스크 1이 생성하는 메시지의 우선순위가 가장 높고 센서태스크 2,센서태스크 3, 제어태스크 1,제어태스크 2 순이다.

각 제어루프의 데드라인이 제어기의 제어태스크 주기와 같다고 가정하면, 노이즈가 없을 경우에는 제어루프 1과 제어루프 2 모두 센서 샘플링에서부터 구동기로 제어값 출력까지가 루프의 주기 이내에 완료되므로 데드라인을 만족시키게 된다. 그러나 센서태스크 1의 메시지 전송 시에 노이즈로 인한 통신에러가 발생한 경우에는 해당 메시지를 재전송하는 것과 같은 에러 복구 방법을 통해 에러가 발생한 메시지 전송을 완료하게 된다. 이러한 재전송 방법을 통한 통신 지연으로 인해 센서태스크 1의 메시지를 기다리고 있는 제어태스크1과 제어태스크 2의 수행 또한 지연되고 또 다시 구

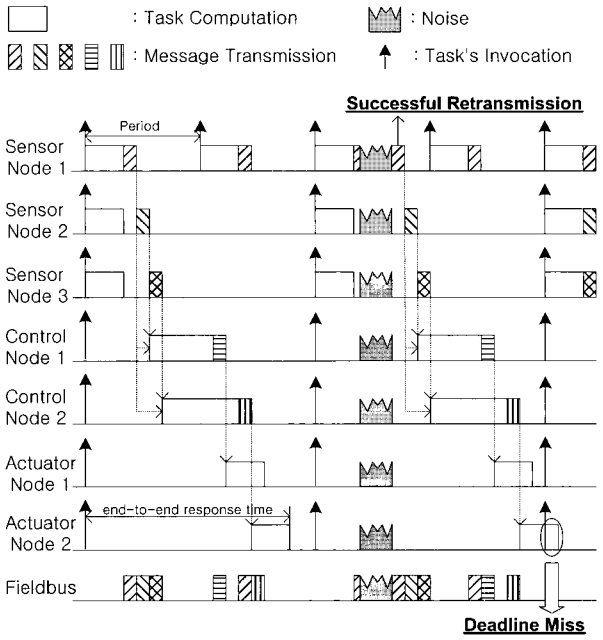

그림 1. 노이즈로 인한 양극단 응답시간 지연.
Fig. 1. End-to-end response time delay caused by noise.

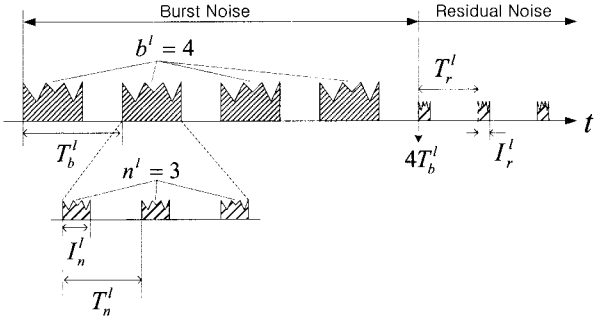

그림 2. 노이즈 모델.
Fig. 2. Noise model.

동태스크 1과 구동태스크 2의 수행 지연을 야기하게 된다. 결과적을 제어루프 2의 경우 양극단 응답시간이 루프의 주기 또는 데드라인을 만족시키지 못하게 되는 치명적인 오류를 야기하게 된다. 그러므로 노이즈 환경에서 운용되는 분산 제어시스템 설계 시에는 실시간 성능을 보장하기 위해 노이즈로 인한 통신 에러를 고려하여야만 한다.

2. 노이즈 모델 및 지연시간[9]

분산 제어시스템 설계 시에 노이즈를 고려하기 위해서는 노이즈 모델링이 선행되어야 하며 모델링된 노이즈를 통해 해당 시스템의 네트웍에 노이즈가 어떤 영향을 미치는지에 대한 분석이 가능해 진다. 본 논문에서는 노이즈에 대한 모델로서 시스템에 미치는 여러 개의 노이즈 소스를 동시에 표현할 수 있는 [9]에서 제안한 모델을 사용한다.

[9]에서 제안된 노이즈 모델이 그림2에 나타나 있으며 하나의 노이즈 소스 l 을 나타내고 있다. 노이즈 소스 l은 최대 T_b^l의 주기로 발생하는 b^l개의 집중(burst) 노이즈 그룹과 최대 T_r^l의 주기를 갖으며 발생하는 잔여(residual) 노이즈로 이루어

진다. 다시 집중 노이즈의 각 그룹은 최대 T_n^l주기마다 I_n^l의 길이를 갖는 n^l개의 노이즈로 구성된다. 잔여 노이즈는 집중 노이즈 후에 T_r^l주기마다 I_r^l 길이를 갖는 노이즈들로 이루어 진다. 이와 같은 노이즈 모델링 방법과 네트웍의 에러 검출 및 복구기능에 의해 노이즈로 인한 통신지연시간을 유도할 수 있다.

그림 2에 나타난 모델과 같은 노이즈가 발생한다면 시간 t 동안에 발생하는 노이즈의 개수를 집중 노이즈 구간에서의 노이즈 수, $Bu^l(t)$ 와 잔여 노이즈 구간에서의 노이즈 수, $Re^l(t)$ 로 분리하여 구할 수 있다. 만약 시간 t가 잔여 노이즈 구간을 포함한다면 $Bu^l(t)$ 는 $n^l * b^l$이 되며, 집중 노이즈 구간만을 포함한다면 완전히 포함된 집중 노이즈 그룹 내의 노이즈 수, $\left\lfloor t/T_b^l \right\rfloor \times n^l$ 와 일부분만 포함되어 있는 집중 노이즈 그룹 내의 해당 노이즈 수, $\left\lceil t \bmod T_b^l / T_n^l \right\rceil$ 를 통해 $Bu^l(t)$ 를 계산할 수 있다. 따라서 (1)과 같이 $Bu^l(t)$ 를 얻을 수 있다.

$$Bu^l(t) = \min\left(n^l \times b^l, \left\lfloor \frac{t}{T_b^l} \right\rfloor \times n^l + \min\left(n^l, \left\lceil \frac{t \bmod T_b^l}{T_n^l} \right\rceil\right)\right) \quad (1)$$

시간 t 동안에 발생하는 잔여 노이즈 구간에서의 노이즈 수, $Re^l(t)$ 는 만약 시간 t 가 집중 노이즈 구간만을 포함한다면 0이 되며 잔여 노이즈 구간을 포함한다면 집중 노이즈 구간을 제외한 시간을 잔여 노이즈 발생 주기로 나눔으로써 $Re^l(t)$ 를 구할 수 있다. 즉, (2)와 같이 $Re^l(t)$ 를 구할 수 있다.

$$\mathrm{Re}^l(t) = \max\left(0, \left\lceil \frac{t - T_b^l \times b^l}{T_r^l} \right\rceil\right) \quad (2)$$

시간 t 동안의 노이즈로 인한 메시지 i의 전송 지연시간을 $E_i(t)$ 라고 하고 시간 t 동안의 노이즈 소스 l로 인한 메시지 i의 전송 지연시간을 $E_i^l(t)$ 라고 하면 $E_i(t)$ 는 하나 이상의 노이즈 소스로부터 영향을 받을 수 있으므로 여러 개의 노이즈 소스에 대한 합으로 나타낼 수 있다. 가령 k개의 노이즈 소스가 존재한다면 이로 인한 메시지 i의 전송지연 $E_i(t)$ 는 (3)으로 나타낼 수 있다.

$$E_i(t) = E_i^1(t) + E_i^2(t) + \ldots + E_i^k(t) \quad (3)$$

위에서 언급한 노이즈 모델과 노이즈 수 계산 방법을 통해 노이즈가 CAN 네트웍에 발생하였을 경우 CAN 메시지의 최악응답시간을 계산할 수 있다. CAN에서 하나의 에러 비트가 초래하는 오버헤드 또는 전송지연, O_i 는 최대 31 비트의 에러 복구 비트와 에러로 인한 메시지의 재전송으로 구성되며[6,7] (4)와 같이 나타난다. (4)에서 τ_{bit}는 1비트를 전송하는데 걸리는 시간이며 $hp(i)$ 는 메시지i 보다 우선순위가 높은 메시지 집합을 의미한다. 메시지의 재전송 시간을 메시지 i 보다 우선순위가 높은 메시지들과 자신을 포함한 모든 메시지 중에 가장 긴 전송시간을 사용한 이유는 최악의 경우를 고려하기 위해서 이다.

$$O_i = 31 \times \tau_{bit} + \max_{k \in hp(i) \cup \{i\}} (C_k) \quad (4)$$

(4)를 이용하여 노이즈 소스l에 대한 메시지 i의 전송지연 $E_i^l(t)$ 은 (5)와 같이 집중 노이즈 구간에서 발생한 노이즈와 잔여 노이즈 구간에서 발생한 노이즈로 인한 에러 비트 당 오버헤드의 곱으로 나타낼 수 있다. 또한 집중 노이즈 구간에서의 노이즈 길이 I_n^l 과 잔여 노이즈 구간에서의 노이즈 길이 I_r^l 가 1비트 전송시간보다 길 경우 이를 보상해주어야 한다.

$$\begin{aligned} E_i^l(t) = {} & Bu^l(t) \times (O_i + \max(0, I_b^l - \tau_{bit})) \\ & + \mathrm{Re}^l(t) \times (O_i + \max(0, I_r^l - \tau_{bit})) \end{aligned} \quad (5)$$

노이즈로 인한 메시지 i의 전송지연은 메시지 i가 CAN의 전송 큐 상에 대기함으로써 미디어를 점유하기 위해 경쟁하기 시작하는 시점부터 전송이 완료되는 시점까지 발생하는 모든 노이즈에 영향을 받는다. 그러므로 노이즈로 인한 메시지 i의 전송지연은 큐잉 지연시간 q_i^E 에 포함되며 (6)의 $E_i(q_i^E+C_i)$ 로 나타내어 지며 (3)에서 처럼 여러 개의 노이즈 소스가 존재할 시에는 각각의 노이즈 소스에 대한 지연시간의 합을 계산함으로써 $E_i(q_i^E+C_i)$ 을 얻을 수 있다.

$$q_i^E = B_i + \sum_{j \in hp(i)} \left(\left\lceil \frac{q_i^E + \tau_{bit} + J_j}{T_j} \right\rceil \times C_j \right) + E_i(q_i^E + C_i) \quad (6)$$

여기서 B_i 는 메시지 i보다 낮은 우선순위를 갖는 메시지로 인해 지연될 수 있는 최대 시간이며, J_j 는 메시지 j가 CAN의 전송 큐로 삽입할 때 발생하는 지터이며, T_j 는 메시지 j의 주기, C_j 는 메시지 j의 순수 전송시간이다. CAN 상에서 모든 노이즈를 고려한 메시지 i의 최악 응답시간, R_i^E - 메시지를 전송 큐에 삽입한 시점부터 실제 메시지가 목적지로 전송이 완료되는 시점 - 는 해당 메시지가 CAN의 전송 큐로 삽입하는데 발생하는 지터 J_j 와 전송 큐에 삽입된 후 메시지를 전송하기 전까지 걸리는 큐잉 지연시간 q_i^E 와 실제 메시지를 전송하는 시간 C_i로 구성되며[14] (7)로 표현된다.

$$R_i^E = J_i + q_i^E + C_i \quad (7)$$

(1) 에서 부터 (7)을 이용하여 얻어진 노이즈 조건 하에서의 메시지 i의 최악응답시간 R_i^E 는 해당 메시지의 데드라인 Ed_i 와 $R_i^E \le Ed_i$ 의 관계가 성립되므로 노이즈 조건 하에서의 예상 가능한 메시지의 데드라인으로 사용될 수 있다.

III. 노이즈 환경 하에서 태스크와 메시지 스케줄링

1. 태스크와 메시지 스케줄링

분산 제어시스템에서 태스크와 메시지의 우선순위와 주기는 시스템의 실시간 특성에 매우 중요한 영향을 미치므로 시스템 설계 시 실시간 특성을 만족시킬 수 있도록 태스크와 메시지의 우선순위와 주기를 설정하는 스케줄링이 필요하다. 이러한 스케줄링의 절차로서 [8]에서 제안한 방법은 다음과 같다. 시스템의 요구사항이 주어지면 이를 만족하는 태스크 그래프를 설계하고 모든 태스크를 물리적인 각 노드에 할당한 후 태스크간 메시지간 제약들을 식으로 유도한다. 제약 유도 후제안된 알고리즘을 통해 모든 태스크와 메시지에 우선순위와 주기를 할당한 후 태스크와 메시지의 최악응답시간

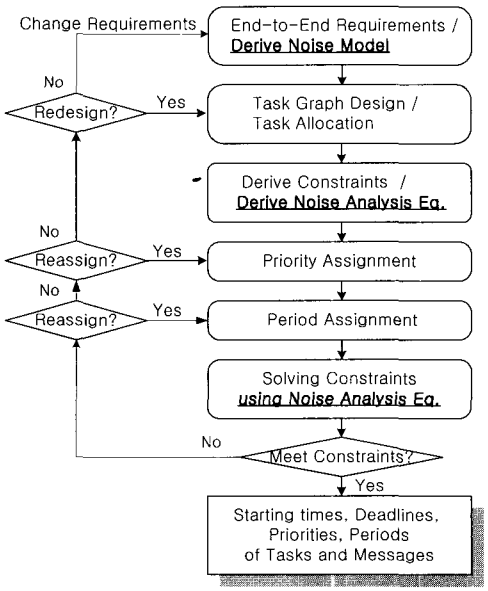

그림 3. 노이즈 환경 하에서 태스크와 메시지 스케줄링.
Fig. 3. Tasks and messages scheduling under noise environment.

을 구하여 이전에 유도된 제약식에 대입하여 식을 풀게 된다. 제약식의 결과가 시스템의 양극단 시간 제약을 만족시키지 못한다면 바로 전 단계인 태스크와 메시지에 주기를 다시 할당하고 제약식을 다시 풀게 된다. 이러한 과정은 제어루프의 양극단 시간제약을 만족하면서 설정 가능한 가장 짧은 주기가 될 때까지 반복하게 된다. 반복적인 주기 조정을 통해서도 원하는 결과를 얻지 못한다면 이전 단계인 우선순위를 조정하던가 태스크 그래프 재설계 또는 시스템 요구사양을 변경하게 된다.

노이즈를 고려하기 위해 기존 연구에서 확장된 부분은 그림 3에서 보여지는 바와 같이, 시스템 요구사양 도출 시에 현장 환경에 대한 노이즈 모델을 유도하고 시스템의 제약 유도 시에 사용된 네트웍에 따라 모델링된 노이즈 영향을 분석할 수 있는 메시지 응답시간 분석방법을 유도한다. 마지막으로 우선순위와 주기할당 후 유도된 제약식들을 풀 때 유도된 응답시간 분석 방법을 사용하게 된다. 이러한 방법을 통해 노이즈로 인해 발생하는 지연시간을 제어루프의 센서에서 구동기까지의 양극단 관점에서 보상할 수 있게 된다.

또한 분산 제어시스템을 구성하는 노드 또는 스테이션들은 서로 분산되어 있는데 이때 각 노드가 설치되어 있는 장소에 따라 노이즈 특성이 다르게 나타날 수 있다. 예를 들어, 대용량 모터 주변에 위치한 노드 또는 통신 선로 경우는 높은 에러율을 나타낼 것이며 이러한 간섭이 없는 곳에 설치된 노드 또는 통신 선로에는 에러율이 매우 낮을 것이다. 그러므로 분산 제어시스템 내부에서도 장소에 따라 노이즈 조건을 따로 적용한다면 시스템의 설계가 더 유연하게 될 수 있는 장점을 갖게 된다. 그러나 본 논문은 이러한 연구를 추후 과제로 하고 시스템 내부에서는 모든 노이즈 특성이 같다라는 가정 하에서 서술한다.

2. 시스템 모델과 태스크 그래프

본 논문에서 제안된 태스크와 메시지 스케줄링 방법을

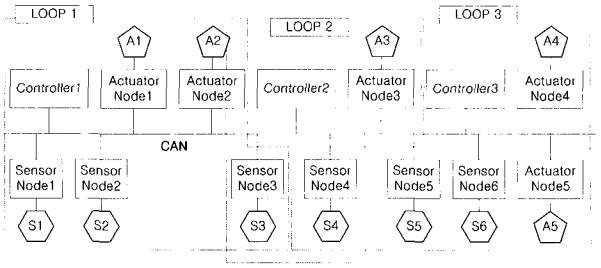

그림 4. 대상 시스템.
Fig. 4. Target System.

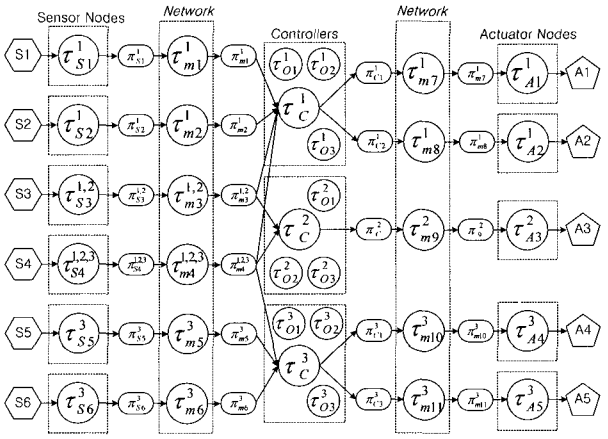

그림 5. 대상 시스템의 태스크 그래프.
Fig. 5. Task graph of target system.

적용할 대상 시스템과 해당 태스크 그래프가 각각 그림4와 그림 5에 나타나 있다. 6개의 센서 노드, 3개의 제어기, 5개의 구동노드로 구성되어 있으며 총 3개의 제어루프를 형성하게 된다. 첫 번째 제어루프는 센서노드 1, 2, 3, 4, 제어기 1,구동노드 1, 2로 구성되며, 두번째 제어루프는 센서노드 3, 4, 제어기 2, 구동노드 3으로 구성된다. 마지막 제어루프는 센서노드 4, 5, 6, 제어노드 3, 구동노드 4, 5로 구성된다. 노드 간 연결은 CAN을 통해 이루어져 있다.

제안된 방법을 통해 시스템을 스케줄링하기 전에 미리 주어지는 시스템 요구사양 설정치들이 표 1에 나타나 있다. 세 개의 제어루프에 양극단 데드라인과 각 태스크들의 순수 수행시간, 메시지 전송시간을 계산해 내기위한 각 메시지들 크기, 제어루프에 포함되지는 않지만 각 제어기 노드에서 수행되고 있는 기타 태스크들에 대한 정보가 나타나 있다.

표 2에는 대상 시스템에 적용할 두 가지 노이즈 모델 CASE1과 CASE2가 2절에 서술된 방법으로 나타나 있다. CASE2 노이즈 모델은 2개의 노이즈 소스가 동시에 존재하는 모델이다. 두 가지 노이즈 환경 하에서 대상 시스템을 운영 특성이 어떤 차이가 있는지 비교 평가하고 제안된 방법을 통해 두 가지 노이즈 환경에서 시스템 스케줄링이 가능함을 보인다. 또한 노이즈가 없는 환경의 스케줄링 결과와 비교함으로써 제어시스템에서 노이즈가 미치는 영향이 매우 중요함을 알 수 있게 된다.

3. 제약식 유도

시스템 요구사양 및 노이즈 모델을 도출한 후 대상 시스템의

표 1. 시스템 특성 값.

Table 1. System parameters.

MADT, [ms]	$MADT^1=60,\ MADT^2=80,\ MADT^3=100$
Execution Time of Tasks, [ms]	$e^1_{S1}=1,\ e^1_{S2}=1,\ e^{1,2}_{S3}=1,\ e^{1,2,3}_{S4}=1,\ e^3_{S5}=1,\ e^3_{S6}=1,$ $e^1_C=7,\ e^1_{O1}=4,\ e^1_{O2}=6,\ e^1_{O3}=8$ $e^2_C=9,\ e^2_{O1}=4,\ e^2_{O2}=6,\ e^2_{O3}=8,$ $e^3_C=11,\ e^3_{O1}=4,\ e^3_{O2}=6,\ e^3_{O3}=8,$ $e^1_{A1}=1,\ e^1_{A2}=1,\ e^2_{A3}=1,\ e^3_{A4}=1,\ e^3_{A5}=1$
Message Length, [Byte]	$\tau^1_{m1}=2,\ \tau^1_{m2}=2,\ \tau^{1,2}_{m3}=8,\ \tau^{1,2,3}_{m4}=8,\ \tau^3_{m5}=2,\ \tau^3_{m6}=2,$ $\tau^1_{m7}=2,\ \tau^1_{m8}=2,\ \tau^2_{m9}=2,\ \tau^3_{m10}=2,\ \tau^3_{m11}=2$
Period of Tasks, [ms]	$T^1_{O1}=20,\ T^1_{O2}=50,\ T^1_{O3}=60$ $T^2_{O1}=20,\ T^2_{O2}=50,\ T^2_{O3}=60,$ $T^3_{O1}=20,\ T^3_{O2}=50,\ T^3_{O3}=60$

표 2. 에러 모델.

Table 2. Error model.

	Noise Source	b^l	n^l	I^l	I^l_r	T^l_f	t^l_f	r^l_f
CASE1	1	1	4	0.5	0.5	4	0.01	10
CASE2	1	4	2	0.1	0.1	1	0.001	10
	2	2	2	0.1	0.1	1	0.001	20

태스크 그래프 유도와 각 태스크의 노드 할당이 이루어지면 이러한 시스템의 특성과 태스크와 메시지 간의 관계를 나타낼 수 있는 식을 유도한다. 식으로 유도된 시스템 특성 및 태스크와 메시지 간의 관계는 시스템 스케줄링 시에 제약으로 사용됨으로써 시스템의 요구사양을 만족하는 스케줄링이 가능하게 된다. 이러한 제약들의 유도는 시스템의 태스크 그래프를 통해 이루어진다. 시스템 요구사항 및 설계 구조에 의해서 유도되는 제약들은 제어루프의 양극단 응답시간에 대한 제약과 태스크 간 주기에 대한 제약, 태스크간 선행제약으로 이루어진다. 각 제어루프에 대한 제약식은 다음과 같다.

□ 제어루프 1

✓ 양극단 응답시간 제약

$\max(\phi^1_{A1}+d^1_{A1},\phi^1_{A2}+d^1_{A2})\le MADT^1$

✓ 주기에 관한 제약

$T^1_{S1}=T^1_{m1},\ T^1_{S2}=T^1_{m2},\ T^{1,2}_{S3}=T^{1,2}_{m3},\ T^{1,2,3}_{S4}=T^{1,2,3}_{m4},$

$T^1_{m1}\,|\,T^1_C,\ T^1_{m2}\,|\,T^1_C,\ T^{1,2}_{m3}\,|\,T^1_C,\ T^{1,2,3}_{m4}\,|\,T^1_C,\ T^1_C=T^1_{m7}=T^1_{A1},$

$T^1_C=T^1_{m8}=T^1_{A2}$

✓ 태스크간 선행제약

$\phi^1_{S1}+d^1_{S1}\le\phi^1_{m1},\ \phi^1_{S2}+d^1_{S2}\le\phi^1_{m2},\ \phi^{1,2}_{S3}+d^{1,2}_{S3}\le\phi^{1,2}_{m3},$

$\phi^{1,2,3}_{S4}+d^{1,2,3}_{S4}\le\phi^{1,2,3}_{m4},$

$\max(\phi^1_{m1}+Ed^1_{m1},\phi^1_{m2}+Ed^1_{m2},\phi^{1,2}_{m3}+Ed^{1,2}_{m3})\le\phi^1_C,$

$\phi^1_{C1}+d^1_{C1}\le\phi^1_{m7},\ \phi^1_{C1}+d^1_{C1}\le\phi^1_{m8},\ \phi^1_{m7}+Ed^1_{m7}\le\phi^1_{A1},$

$\phi^1_{m8}+Ed^1_{m8}\le\phi^1_{A2}$

□ 제어루프 2

✓ 양극단 응답시간 제약

$\phi^2_{A3}+d^2_{A3}\le MADT^2$

✓ 주기에 관한 제약

$T^{1,2}_{S3}=T^{1,2}_{m3},\ T^{1,2,3}_{S4}=T^{1,2,3}_{m4},\ T^{1,2}_{m3}\,|\,T^2_C,\ T^{1,2,3}_{m4}\,|\,T^2_C,$

$T^2_C=T^2_{m9}=T^2_{A3}$

✓ 태스크간 선행제약

$\phi^{1,2}_{S3}+d^{1,2}_{S3}\le\phi^{1,2}_{m3},\ \phi^{1,2,3}_{S4}+d^{1,2,3}_{S4}\le\phi^{1,2,3}_{m4}$

$\max(\phi^{1,2}_{m3}+Ed^{1,2}_{m3},\phi^{1,2,3}_{m4}+Ed^{1,2,3}_{m4})\le\phi^2_C,\ \phi^2_{C2}+d^2_{C2}\le\phi^2_{m9},$

$\phi^2_{m9}+Ed^2_{m8}\le\phi^2_{A3}$

□ 제어루프3

✓ 양극단 응답시간 제약

$\max(\phi^3_{A4}+d^3_{A4},\phi^3_{A5}+d^3_{A5})\le MADT^3$

✓ 주기에 관한 제약

$T^{1,2,3}_{S4}=T^{1,2,3}_{m4},\ T^3_{S5}=T^3_{m5},\ T^3_{S6}=T^3_{m6},\ T^{1,2,3}_{m4}\,|\,T^3_C,$

$T^3_{m5}\,|\,T^3_C,\ T^3_{m6}\,|\,T^3_C,\ T^3_C=T^3_{m10}=T^3_{A4},\ T^3_C=T^3_{m11}=T^3_{A5}$

✓ 태스크간 선행제약

$\phi^{1,2,3}_{S4}+d^{1,2,3}_{S4}\le\phi^{1,2,3}_{m4},\ \phi^3_{S5}+d^3_{S5}\le\phi^3_{m5},\ \phi^3_{S6}+d^3_{S6}\le\phi^3_{m6},$

$\max(\phi^{1,2,3}_{m4}+Ed^{1,2,3}_{m4},\phi^3_{m5}+Ed^3_{m5},\phi^3_{m6}+Ed^3_{m6})\le\phi^3_C,$

$\phi^3_{C3}+d^3_{C3}\le\phi^3_{m10},\ \phi^3_{C3}+d^3_{C3}\le\phi^3_{m11},\ \phi^3_{m10}+Ed^3_{m10}\le\phi^3_{A4},$

$\phi^3_{m11}+Ed^3_{m11}\le\phi^3_{A5}$

4. 스케줄링 결과

3.3절에서 유도된 제약식을 이용하여 시스템을 스케줄하기 위해서는 태스크와 메시지에 우선순위와 주기가 할당되어야 한다. 유도된 제약식 중 주기에 관한 제약식을 이용하는 [8]에서 제안한 태스크와 메시지의 우선순위 및 주기 할당 알고리즘을 이용하여 태스크와 메시지에 우선순위와 주기를 할당한다. 우선순위와 주기 할당이 이루어지면 태스크와 메시지의 최악응답시간을 계산한다. 메시지의 최악 응답시간은 2.2절에 제시되어 있는 분석 방법을 사용함으로써 노이즈 조건 하에서의 통신에러로 인한 전송지연을 고려하여 시스템을 스케줄링하는 것이 가능해 진다. 얻어진 태스크와 메시지의 최악응답시간을 유도된 제약식의 데드라인으로 대입하여 태스크간 선행 제약식들을 풀어나간다. 선행 제약식들을 모두 계산한 결과가 양극단 시간제약을 만족한다면 이때의 태스크와 메시지의 주기, 우선순위, 초기시작시간이 노이즈 환경 하에서 시스템의 양극단 요구사양을 만족하는 태스크와 메시지의 스케줄링 결과가 된다.

표 2에서 정의한 두 가지 노이즈 모델과 노이즈가 존재하지 않는 세 가지 경우에 대한 대상 시스템의 태스크와 메시지 스케줄링 결과가 그림 6에서 부터 그림 11까지 나타나 있다. 그림의 X 축은 [8]에서 제안한 알고리즘의 반복적인 계산 횟수를 의미하며 Y축은 [ms] 단위의 각 제어루프의 양극단 최악응답시간과 주기를 나타낸다. 그림 6에서 그림 10까지는 제안된 알고리즘을 이용하여 얻어지는 각 제어루프의 주기와 이때의 양극단 최악응답시간 결과가 나타나 있다. [8]에서 제안된 알고리즘은 이분법 알고리즘에 기반한 반복적 계산을 통해 시스템이 스케줄링 가능한 한도 내에서 제어루

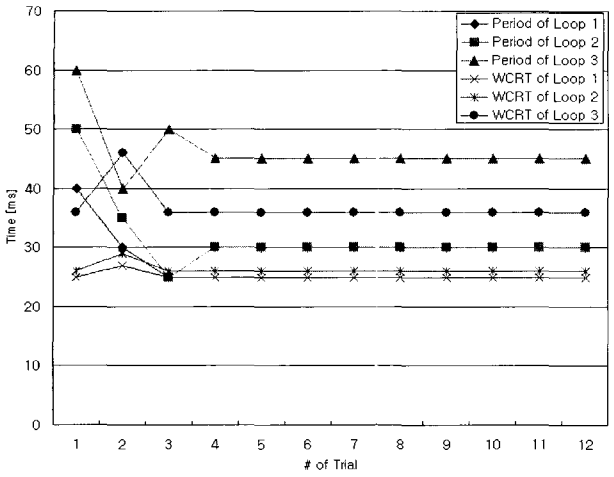

그림 6. 100kbps의 CAN 상에 노이즈가 없는 경우.
Fig. 6. CAN: 100kbps under no noise condition.

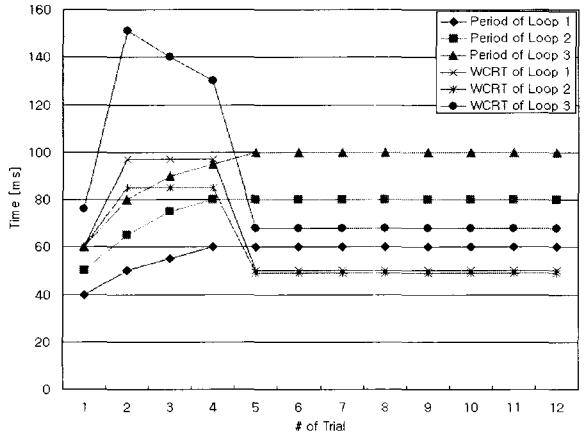

그림 7. 100kbps의 CAN 상에 CASE1 노이즈가 있는 경우.
Fig. 7. CAN: 100kbps under CASE1 noise condition.

프에 적용 가능한 가장 짧은 주기를 얻어내는 알고리즘이다. 즉, 얻어낸 제어루프의 주기가 양극단 응답시간보다 길어야만 시스템 스케줄링이 가능하게 된다.

그림 6에는 노이즈가 없는 환경에서 100kbps의 CAN을 사용하는 대상시스템의 각 제어루프의 주기설정 값 및 해당 주기에서의 양극단 최악응답시간에 대한 결과가 나타나 있다. 그림 6에 나타난 바와 같이 노이즈가 없는 환경에서 대상 시스템을 구성하는 3개의 제어루프는 각각 25ms, 26ms, 36ms의 양극단 최악응답시간을 갖는 30ms, 30ms, 45ms의 주기로 설정되었다. 이렇게 설정된 주기는 60ms, 80ms, 100ms의 각 제어루프의 데드라인을 만족하는 결과이며 노이즈가 없는 환경에서 100kbps의 CAN을 사용하는 시스템이 요 구사항을 만족하도록 스케줄 가능함을 알 수 있다.

그림 7에는 표 2의 CASE 1 모델의 노이즈 환경에서 100kbsp 속도의 CAN을 사용할 경우에 각 제어루프의 주기 설정 값 및 해당 주기에서의 양극단 최악응답시간을 나타내고 있다. CASE 1의 노이즈 조건에서는 통신에러가 발생하므로 그림 6보다는 상당히 길어진 60ms, 80ms,100ms로 각각의 제어루프 주기가 설정되었으며 이때 양극단 응답시간은 50ms, 50ms, 65ms가 되었다. 이렇게 얻어진 결과 또한 각 제어루프의 데드라인을 만족하는 결과이며 CASE 1 모델의 노이즈 환경에서도 100kbps의 CAN을 사용하는 시스템이 스케줄 가능함을 알 수 있다.

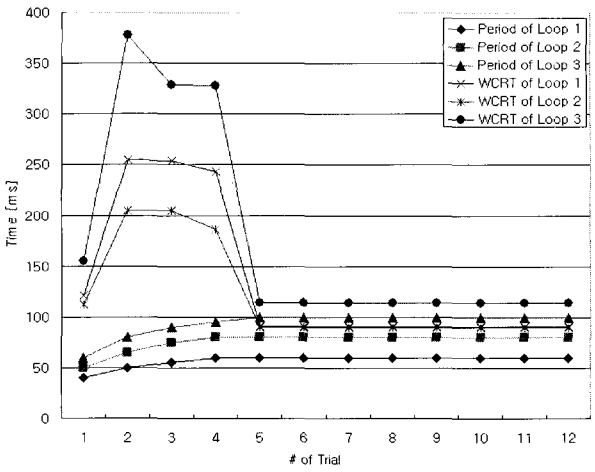

그림 8. 100kbps의 CAN 상에 CASE2 노이즈가 있는 경우.
Fig. 8. CAN: 100kbps under CASE2 noise condition.

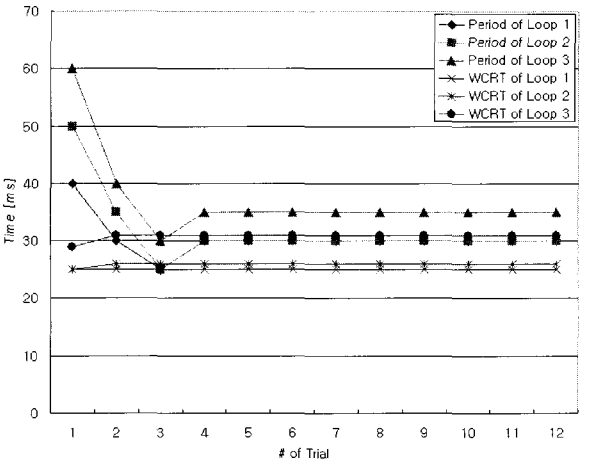

그림 9. 500kbps의 CAN 상에 CASE1 노이즈가 있을 경우.
Fig. 9. CAN: 500kbps under CASE1 noise condition.

그림 8에는 두 개의 노이즈 소스를 갖는 표 2의 CASE 2 모델의 노이즈 조건 하에서 100kbps의 CAN을 사용하는 시스템을 스케줄링한 결과가 나타나 있다. CASE 2의 노이즈 조건에서는 많은 양의 에러로 인해 메시지 전송지연이 상당히 길어졌으며 이로 인해 각 제어루프의 양극단 응답시간이 루프의 데드라인을 만족시킬 수 없었다. 그림 8에 나타난 바와 같이 노이즈로 인한 통신지연이 시스템을 스케줄링하지 못할 정도로 길기 때문에 이러한 문제를 해결하기 위해서 시스템 요구사양인 네트웍 속도를 500kbps 로 수정하였다. 이에 대한 결과가 그림 9와 그림 10에 나타나 있다.

그림 9와 그림 10에는 각각 표 2의 CASE 1과 CASE 2의 노이즈 조건 하에서 500kbps의 CAN을 사용했을 때 제어루프의 주기 설정 값과 양극단 응답시간이 나타나 있다. CASE 1 노이즈 조건 하에서 각 제어루프는 25ms, 26ms, 31ms의 응답

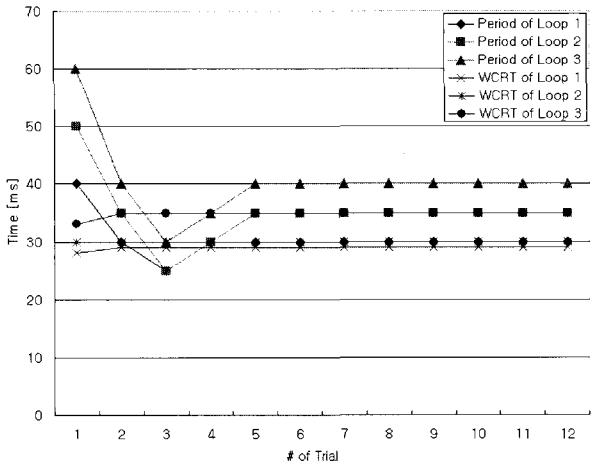

그림 10. 500kbps의 CAN상에 CASE2 노이즈가 있을 경우.
Fig. 10. CAN: 500kbps under CASE2 noise condition.

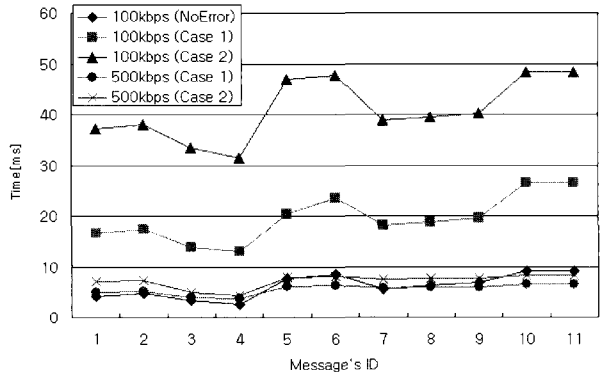

그림 11. 노이즈 조건에 따른 메시지 최악응답시간.
Fig. 11. Worst-case response times of all messages.

시간을 갖는 30ms, 30ms, 35ms의 주기로 스케줄 되었으며 CASE 2 노이즈 조건 하에서도 29ms, 30ms, 35ms의 극단 응답시간을 갖는 30ms, 35ms, 40ms의 주기로 각각의 제어루프가 스케줄 되었다. 100kbps의 CAN을 사용한 경우에는 CASE 2의 노이즈 조건 하에서 시스템을 스케줄 할 수 없었지만 CAN의 속도를 500kbps로 높여주면 CASE 2의 노이즈 조건 하에서도 시스템이 스케줄 가능했다. 이러한 결과를 통해 시스템 설계 시에 노이즈 조건에 따라 시스템의 요구사양인 네트웍 속도 선정이 가능함을 알 수 있다.

그림 11에는 네트웍 속도와 노이즈 모델에 따른 각 메시지의 최악응답시간이 나타나 있다. 네트웍 속도가 100kbps인 경우는 세 가지 노이즈 조건에 따라 각 메시지의 응답시간이 크게 변화함을 알 수 있다. 이는 저속도의 네트웍이기 때문에 에러 복구에 걸리는 시간이 상대적으로 길다는 것을 의미한다. 500kbps인 경우에는 노이즈 조건에 따라 100kbps에 비해 상대적으로 작게 변화함을 알 수 있다. 에러로 인한 통신지연은 각 메시지의 입장에서는 상대적으로 작은 지연일 수도 있지만 제어루프의 양극단에 미치는 지연시간은 해당하는 모든 메시지의 통신지연의 합이므로 시스템에 매우 큰 영향을 미치게 된다. 또한 500kbps 사용 시 CASE 2의 노이즈 조건에서도 메시지의 최악응답시간은 노이즈가 없는 100kbps

표 3. 500kbps의 CAN 상에 CASE2 노이즈가 있는 경우.
Table 3. CAN: 500kbps under CASE2 condition.

Loop 1 (MADT: 60)									Loop 2 (MADT: 80)									Loop 3 (MADT: 100)								
Task	Priority		e	T	R	d	ϕ	Task	Priority		e	T	R	d	ϕ	Task	Priority		e	T	R	d	ϕ
	T	M							T	M							T	M					
τ^1_{S1}	1		1	30	1	1	0	$\tau^{1,2}_{S2}$	1		1	5	1	1	0	$\tau^{1,2,3}_{S4}$	1		1	5	1	1	0
τ^1_{m1}		3	0.1	30	7.2	8	1	$\tau^{1,2}_{m3}$		2	0.3	5	4.8	5	1	$\tau^{1,2,3}_{m4}$		1	0.3	5	4.3	5	1
τ^1_{S2}	1		1	30	1	1	0	$\tau^{1,2,3}_{S4}$	1		1	5	1	1	0	τ^3_{S7}	1		1	40	1	1	0
τ^1_{m2}		4	0.1	30	7.3	8	1	$\tau^{1,2,3}_{m4}$		1	0.3	5	4.3	5	1	τ^3_{m5}		8	0.1	40	7.9	8	1
$\tau^{1,2}_{S3}$	1		1	5	1	1	0	τ^2_C	2		9	35	15	15	6	τ^3_{S8}	1		1	40	1	1	0
$\tau^{1,2}_{m3}$		2	0.3	5	4.8	5	1	τ^2_{O1}	1		4	20	4	4	0	τ^3_{m6}		9	0.1	40	8.1	9	1
$\tau^{1,2,3}_{S4}$	1		1	5	1	1	0	τ^2_{O2}	3		6	50	19	19	0	τ^3_C	2		11	40	15	15	10
$\tau^{1,2,3}_{m4}$		1	0.3	5	4.3	5	1	τ^2_{O3}	4		8	60	31	31	0	τ^3_{O1}	1		4	20	4	4	0
τ^1_C	2		7	30	11	11	9	τ^2_{m9}		7	0.1	35	7.8	8	21	τ^3_{O2}	3		6	50	25	25	0
τ^1_{O1}	1		4	20	4	4	0	τ^2_{A3}	1		1	35	1	1	29	τ^3_{O3}	4		8	60	33	33	0
τ^1_{O2}	3		6	50	17	17	0									τ^3_{m10}		10	0.1	40	8.2	9	25
τ^1_{O3}	4		8	[illegible]	29	29	0									τ^3_{A4}	1		1	40	1	1	34
τ^1_{m7}		5	0.1	30	7.5	8	20									τ^3_{m11}		11	0.1	40	8.2	9	25
τ^1_{A5}	1		1	30	1	1	28									τ^3_{A5}	1		1	40	1	1	34
τ^1_{m8}		6	0.1	30	7.6	8	20																
τ^1_{A6}	1		1	30	1	1	28																

경우의 메시지 응답시간과 비슷한 값을 나타내고 있다. 이를 통해 노이즈 조건 하에서 시스템 성능을 만족하기 위해서는 적절한 네트웍 속도 선정이 필요함을 알 수 있다.

제안된 태스크와 메시지 스케줄링 방법을 통해 네트웍 속도 500kbps와 표 2의 CASE 2 노이즈 조건에서 시스템을 스케줄링한 결과가 표 3에 상세하게 나타나 있다. 스케줄 결과로 얻어진 태스크와 메시지의 우선순위, 주기, 시작시간, 데드라인과 같은 파라미터는 시스템 구현 및 설치 시에 태스크와 메시지의 파라미터로 정적으로 설정되어 요구사양을 만족하며 시스템이 운영되는 것을 가능하게 한다.

제어시스템에 미치는 노이즈의 영향을 시스템을 구성하는 제어루프의 양극단에서 살펴 봄으로써 노이즈 환경 하에서 운용되는 분산 제어시스템 설계 시에 노이즈에 대한 고려가 매우 중요함을 알 수 있었다. 또한 제안된 스케줄링 방법을 통해 시스템 설계 및 시스템을 구성하는 태스크와 메시지를 스케줄하는 것이 가능하였고 시스템 설계자는 네트웍 속도와 같은 설계 요구사양을 조정할 수 있게 되었다.

VI. 결론

강한 EMI를 발생시키는 발전기나 대용량 모터 등과 같은 기기들이 동작하고 있는 산업현장에 설치되어 있는 제어 시스템은 이러한 환경적 요인으로 인해 통신선로 상에 다량의 노이즈 영향을 받게 된다. 이러한 노이즈로 인한 통신에러는 시스템의 응답시간을 지연시키고 시스템 성능에 큰 영향을 미치게 되므로 시스템 설계 시에 이를 반드시 고려해야 한다.

본 논문에서는 노이즈 환경 하의 제어시스템을 위한 태스크와 메시지 스케줄링 방법을 제안하고 예제 시스템에 제안된 방법을 적용하여 그 유효성을 검증하였다. 이를 위해 여러 개의 노이즈 소스를 나타낼 수 있는 노이즈 모델링 방법을 사용하였고 CAN에서 노이즈에 대한 응답시간 분석방법에 대하여 서술하였다. 2개의 노이즈 모델에 대해 CAN으로 구성된 분산 제어시스템의 양극단 응답시간에 미치는 영향 분석을 통해 노이즈로 인한 전송지연이 시스템 스케줄에 큰

영향을 미치는 것을 나타내었으며, 노이즈 조건에 따라 네트웍 속도와 같은 시스템 요구사양 조정이 필요함을 알 수 있었다. 제안된 스케줄링 방법을 통해 시스템 설계자는 시스템이 설치 및 운영되는 환경의 노이즈를 고려하여 시스템을 설계할 수 있게 된다.

향후 연구 과제로서 시스템 요구사양과 노이즈 모델과의 매핑 방법과 장소에 따른 노이즈 모델 적용방법 등이 있다.

참고문헌

[1] A. Burns, "Preemptive priority based scheduling: an appropriate engineering approach in Principles of real-time systems," *Prentice Hall,* 1994.

[2] J. Xu and D. Parnas, "Scheduling processes with release times, deadlines, precedence and exclusion relations," *IEEE Tr. on Software Engineering,* pp. 360-369, March, 1990.

[3] J. Y.-T Leung and J. Whitehead, "On complexity of fixed-priority scheduling of periodic real-time tasks," *Performance Evaluation,* 2(4), pp. 237-250, December, 1982.

[4] K. Tindell, "Holistic schedulability analysis for distributed hard real-time systems," *Technical Report,* YCS-197, Dept. of Computer Science Univ. of York, NOV., 1994.

[5] R. Gerber and S.S. Hong, "Guaranteeing real- time requirements with resource-based calibration of periodic processes," *IEEE Tr. on Software Engineering,* 21(7), July, 1995.

[6] J. W. Park, Y. S. Kim, S. S. Hong, M. Saksena, S. H. Noh and W. H. Kwon, "Network conscious of distributed real-time systems," *Journal of System Architecture,* pp. 131-156, 1998.

[7] S. Faucou, A.-M. Deplanche and J.-P. Beauvais, "Heuristic techniques for allocating and scheduling communicating," *WFCS-2000,* pp. 257-265, 2000.

[8] 김형육, 이철민, 박홍성 "분산 제어시스템에서의 태스크와 메시지 기반 스케줄링을 이용한 최적 주기와 우선순위 할당", 제어 · 자동화 · 시스템공학 논문지 제 8 권 제 6 호 pp. 506-513, 2002.6.

[9] S. Punnekkat, H. Hansson, and C. Norston, "Response time analysis under errors for CAN," *Proc. of the 6th Real-Time Technology and Applications Symposium,* pp.258-265, MAY, 2000.

[10] L. Cai and W. Zhang, "EMI in hydropower plant and EMC design for its computer monitoring and control system," *the 3rd International Symposium on Electromagnetic Compatibility,* pp.378 -381, 2002.

[11] K. Tindell and A. Burns, "Guaranteed message latencies for distributed safety critical hard real-time networks," *YCS 229, Dept. of Computer Science,* Univ. of York, June, 1994.

[12] CAN in Automation (CiA), CAN Specification 2.0 Part A and Part B. http://www.can-cia.de

[13] Intel, 82527 Serial Communications Controller Architectural Overview, Jan., 1996.

[14] K. Tindell, H. Hansson, and A. Wellings, "Analyzing real-time communications: controller area network," *IEEE Real-time Systems Symposium,* 1994.

김 형 육

1973년 7월 16일생. 1999년 강원대 제어계측공학과 졸업. 동 대학원 제어계측공학과 석사(2001). 2001년~현재 동 대학원 제어계측공학과 박사과정. 관심분야는 실시간 스케줄링, 필드버스, 무선 데이터 통신 등.

박 홍 성

1961년 3월 16일생. 1983년 서울대 제어계측공학과 졸업. 동 대학원 제어계측공학과 석사 (1986). 동 대학원 제어계측공학과 박사(1992). 1992년~현재 강원대 전기전자정보통신공학부 교수. 관심분야는 실시간 네트워크, 무선 네트워크 등.

윤 건

1977년 2월 13일생. 2001년 강원대 제어계측공학과 졸업. 동 대학원 제어계측공학과 석사(2003.8). 2003년~현재 LG산전 자동화 연구소 Advanced Platform 연구단. 관심분야는 분산 시스템, 임베디드 시스템 등.