

Field Bus로서의 BITBUS Network에 대한 성능 연구

(A Study on the Performance of BITBUS Network as a Field Bus)

成 百 文*, 林 東 民**, 李 愷 洙**, 殷 鍾 官**

(Baik Moon Sung, Dong Min Lim, Hwang Soo Lee, and Chong Kwan Un)

要 約

공정 제어나 제조 자동화에 있어서 현장의 각 sensor나 actuator와 controller간의 연결에 있어서 종래에는 point to point 연결로 인해 cabling 문제가 점점 복잡해졌는데 이 문제를 해결하기 위해 한 개의 연결로 대체하고 각 sensor나 actuator가 time multiplexing에 의해 한 통신선을 공유하는 Field Bus를 도입한다.

본 논문에서는 Field Bus로 구현된 BITBUS network에서 BITBUS protocol상에는 정의되지 않은 cyclic service strategy로써 모든 node에 같은 priority를 주는 균등 priority 방식이나 특정한 node에 priority를 주는 차등 priority 방식을 제안하여 Kuehn과 Boxma의 해석법을 이용하여 cyclic service 형태의 model을 해석하고 simulation 결과와 비교하였다. 또한 해석이 가능하지 않은 조건에서는 simulation을 행하여 network의 성능을 waiting time의 관점에서 분석하였다. 결과로부터 IEC(International Electrotechnical Commission)에서 규정한 제조 자동화에서 요구하는 임계 응답시간 5msec내에 이르기 위해서는 전송률이 2.4Mbps인 상황에서 각 node의 평균 message 도착률이 5,000 messages/sec라 하면 10개의 node까지 연결할 수 있음을 알 수 있었다.

Abstract

With the increasing complexity of cabling at sensory level in process control and manufacturing automation, field buses were introduced to replace the traditional point to point links from each sensor or actuator to its controlling equipments by a single link on which all information is transmitted serially and multiplexed in time.

In this paper, we introduce the BITBUS network as a field bus. For the service discipline of the BITBUS network, two service strategies are proposed to obtain the performance of the network. They are the equal priority cyclic service strategy and the non-equal priority cyclic service strategy. The former assigns equal priority to each node for polling and the latter assumes non-equal priority. The BITBUS network was modeled as a cyclic queueing model and it is analyzed by two methods: the Kuehn's and the Boxma's. Computer simulation was also done for the cyclic queueing model and simulation results were compared with those. Under mathematically non-analyzable environment, only the computer simulation was done. From the simulation result, in order to meet the response time requirement of 5 msec imposed by International Electrotechnical Commission when each node has the average traffic of 5000 messages/sec in manufacturing automation the number of slave nodes should be smaller than 10 at the transmission rate of 2.4 Mbps.

*正會員, 現代電子産業株式會社

(Hyundai Elec. Industries Co., Ltd.)

**正會員, 韓國科學技術院 電氣 및 電子工學科

(Dept. of Elec. Eng., KAIST)

接受日字: 1989年 8月 22日

(본 연구는 한국과학재단 목적기초 연구비 지원에 의한 것임.)

I. 서 론

미래의 공장 자동화는 무인화 공장 또는 CIM (Computer Integrated Manufacturing)으로 일컬어지게 될 것이다.¹⁾ 이 말은 상품의 구상에서 설계, 제조판매, 적하에 걸친 전 과정에서 사이에 사람의 개입없이 자동화한다는 뜻이다. 이런 추세에서 환경 변화에 유

연하게 대처할 수 있는 통신망의 요구가 증대되어 왔고 MAP (Manufacturing Automation Protocol)이 출현하게 되었다. MAP은 수요자 입장인 General Motors사에 의해 자사내의 필요에 의해 개발되기 시작한 것인데 국제 표준화 기구의 OSI (Open Systems Interconnection) 7계층 프로토콜에 바탕을 두었기에 General Motors사와 같은 공장 자동화용 장비의 수요자들 뿐만 아니라 장비의 제작자들에게도 수용되기 시작하였다!^{2, 3}

한편 제어기와 actuator 그리고 robotics등의 실체들이 산재해 있는 공장 현장에서는 대부분 실시간 동작을하므로 각 실체들 간의 통신이나 사람과 기계들 간의 interface에서 발생하는 정보는 길이가 짧고 발생률이 높은 특징이 있다!⁴ 또한 현장의 실체들은 제조환경의 변화에 쉽게 대처할 수 있어야 하므로 설치가격이 저렴해야 하고 실시간 동작을 위해 프로토콜을 단순하게 하여 정보의 전송시간을 단축시키는 것이 요청된다. 이런 공장 현장수준의 network 프로토콜로서 MAP은 지나친 overhead가 된다. 그러므로 CIM의 큰 흐름속에 계층 구조상으로 기초가 되는 공장 현장의 통신에서 field bus의 도입은 증대되는 추세에 있다!^{4, 5, 6}

Field bus는 아직 표준화가 이뤄지지 않은 채 field bus의 개념에 맞는 여러가지 방식들이 제안되어 사용되고 있다!⁴ 본 논문에서는 지금까지 가장 일반적으로 알려진 Intel사의 BITBUS방식을 택하기로 한다. BITBUS는 IBM의 SDLC 프로토콜에 기초하고 있다. 64Kbps, 375Kbps, 2.4Mbps의 표준 전송률을 가지고 bus 위에 serial로 정보를 전송한다. RS-485 전기적인 사양을 따르고 최대 250 node가 bus위에 연결될 수 있는 특징을 가지고 있다.

본 논문에서는 BITBUS network이 실시간 환경에 얼마만큼 유용할 수 있는가를 message의 waiting time 관점에서 수학적 해석과 simulation을 행하기로 한다.

II. 제안된 BITBUS Network와 Model

BITBUS protocol은 IBM SDLC에 기초하고 있다. NRZI (Not Return to Zero Inverting) coding을 사용하며 64Kbps, 375Kbps, 2.4Mbps의 표준 전송률을 가지고 Bus상에 serial로 정보를 전송한다. BITBUS는 전기적으로 RS-485사양을 따르고 한 쌍 (2.4Mbps) 혹은 두 쌍 (375Kbps, 64Kbps)의 twisted pair 선로를 통해 전송이 이뤄진다.

Master station이 모든 message 전송의 시발점이 된다. 즉 slave로 polling frame이나 data frame을 보내

야만 slave에서 비로소 응답이나 data frame을 보낼 수 있다. Cross communication과 broadcast frame 등은 허용되지 않는다. Frame 들은 표준 SDLC frame 들이다. 응용계층에는 다양한 primitive들이 제공되지만 cyclic service strategy에 관해 정해진 것은 없다. 그림1에 BITBUS의 계층구조를 보여주고 있다.

BITBUS protocol에 따라 그림 2와 같이 한개의 master에 multidrop형태로 slave가 연결되어 network을 구성하였다.

각 station을 순환하면서 polling하는 방법은 BITBUS 사양에는 정해져 있지 않고 사용자가 응용에 따라 임의로 정할 수 있게 되어 있다. 제시된 network에선 N_0 개의 slave node를 가정한다. 이들 slave들을 polling하는 형태에 관해 고려해 보면 제안하는 network에선 각 slave node들은 master로부터 polling을 받기전

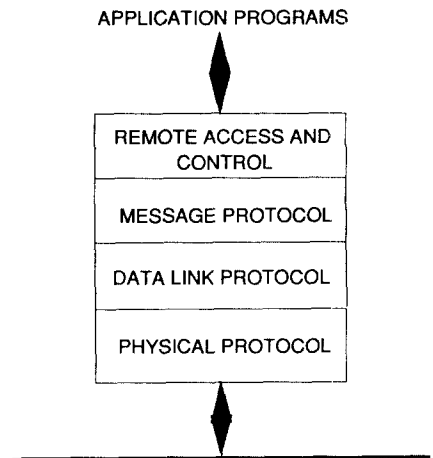


그림 1. BITBUS 사양의 계층구조
Fig. 1. The layer structure of the BITBUS specification.

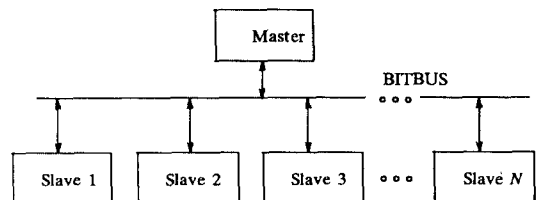


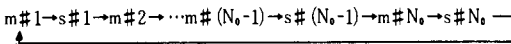
그림 2. BITBUS network 구성도
Fig. 2. The skeleton of BITBUS network.

에 message를 먼저 보낼 수 있는 권한이 부여되지 않으므로 contention문제는 생기지 않는다.

Master의 polling strategy는 사용자가 응용에 따라 임의로 정할 수 있다. 우선 random하게 polling 하는 경우를 생각할 수도 있다. 이 경우는 확률적으로 slave가 선정되므로 어느 slave는 전혀 polling을 받지 못하는 최악의 상황이 있을 수도 있게 된다. 그런데 BITBUS 응용이 대개 빠른 응답시간을 필요로 하고 어느 한도 내에는 반드시 전달되어야만 하는 time critical한 환경에서 쓰이고 있으므로 random하게 slave를 선정하는 방법은 택할 수 없다. 그러면 polling strategy를 달리해서 어느 주기 내에선 반드시 모든 slave node들이 적어도 한 번은 polling을 당하는 경우를 고려한다. 이 경우에 다시 구분지어 한주기 안에 모든 slave를 똑같은 횟수만큼 polling하는 균등 priority polling방식과 임의의 slave node는 priority를 다른 slave보다 높게 준다는 의미로 한 주기 내에 다른 slave보다 polling을 더 해 주는 방식으로 나누어 볼 수 있다. 본 논문에서는 simulation의 경우 두가지 (priority, non-priority) 경우를 모두 다루게 되지만 analysis에선 해석의 편의를 위해 모든 slave에 균등하게 priority를 주어 polling하는 형태를 택하기로 한다.

균등 priority polling방식에 따르면 미리 정해진 순서에 따라 slave node를 차례로 polling하고 다시 순환하여 같은 순서로 polling을 하게 된다. 이때 BITBUS protocol에 따르면 polling을 할 때 master는 unnumbered frame으로 할 수도 있고 polling하고자 하는 slave station으로 가고자 하는 message가 있는 경우 message를 piggyback 시켜서 동시에 polling을 행할 수도 있다. 따라서 network의 효율적인 점을 고려할 때 가능한 한 piggyback시키지 않고 polling하는 경우를 감소시켜야 한다. 그러므로 master쪽에 queue를 한개만 만들지 않고 slave의 수만큼 만들기로 한다. 이유는 한 개의 queue만 만들 경우 목표 slave가 도착하는 message에 따라 임의로 선정되므로 미리 정해진 polling 순서에 따라 piggyback을 1/전체 slave수 (N₀)의 확률만큼 행할 수 있다. 따라서 이 점을 고려하여 master쪽에 각 slave node마다 자기 대응된 queue를 따로 만들어 slave의 수만큼 master쪽에 queue를 만들기로 한다. 이상을 정리하여 구성한 queueing model이 그림 3에 나타나 있다. 이때 각 master 쪽과 slave쪽의 message 도착은 poisson arrival을 따른다고 가정한다.

전체적으로 cyclic polling순서는 다음과 같다.



여기서 m#i는 master쪽의 queue중 i(i=1,2,...N₀)번째 queue를 뜻한다. s#i는 slave쪽에서 i(i=1,2,...N₀)번째 slave를 지칭한다. 따라서 그림3에 나타난 queueing model은 그림4에서와 같이 ring topology로 나타낼 수 있다.

Ring topology로 해석하기 위해서 본래 network의 실체들이 어떻게 해석에 필요한 요소로 변형될 수 있는지 알아본다.

1) Switchover Time

Master나 slave쪽의 queue를 찾아서 보내고자 하는 message가 없는 경우 unnumbered polling frame이나 혹은 응답 frame만을 전송하게 된다. 이 unnumbered frame이 전송되는 시간을 switchover time으로 대비시킨다. 따라서 piggyback시키는 경우 switchover time이 없게 된다.

2) Service시간

Message로부터 frame의 생성과 frame의 전송에 소요되는 시간을 말한다. 따라서 service시간을 알려면 message 전송 경로를 상세히 추적하여 각각의 과정에 소요되는 시간을 추정해야 한다. Message가 어떻게 만들어지고, 만들어진 Message가 어떤 경로를 거쳐 전송 되는지를 알아보는 것은 service 시간으로 정의한 전송시간을 추정하는데 기준이 될 수 있다. 따라서 전송경로를 도식적으로 나타내면 그림 5와 같다. 상세한 설명을 덧붙이면 우선 master extension으로 동작하는 IBM-PC에서 돌아가는 message 전송을 위한 application program은 20 instruction으로 이뤄졌다고 가정하고 한 instruction은 10개의 clock으로 구성되고 IBM-PC의 CPU clock은 10 MHz로 가정하여 application program에서 소요되는 시간을 계산한다.

그리고 iDCX 51 Executive가 Bus Arbiter로 동작하여 전송 task와 수신 task간의 message 전송이 이뤄지게 하는 RQSENDMESSAGE system call 수행 시간은 106 microseconds로 알려져 있다.⁷⁾

Message 길이의 경우 information field를 포함하지 않은 ack.나 polling만을 위한 frame의 경우 5 byte로 가정하고 information이 포함된 frame인 경우 18 byte로 가정했다. 더불어 2.4Mbps, 375Kbps, 64Kbps 각각의 전송률에 대해 전송시간을 계산했다.

각 message 길이와 전송 경로에 따라 다음과 같이 4가지 유형으로 message를 구분하여 각각의 경로에 대해 전송 시간을 구하였다.

가) Master node에서 임의의 slave node로 가는 message (Piggyback없이 polling의 목적으로만 쓰인 unnumbered frame).

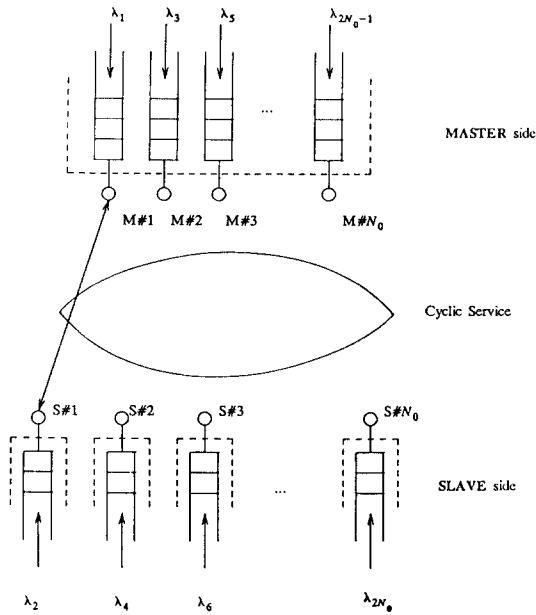


그림 3. BITBUS network의 queuing model
Fig. 3. The queuing model of BITBUS network.

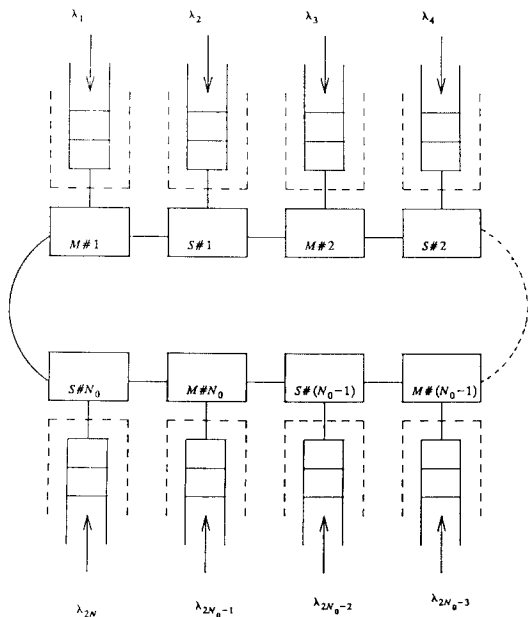


그림 4. Queuing model로부터 변형된 ring topology
Fig. 4. The ring topology transformed from queuing model.

- 나) Master쪽 queue에 있는 message 임의의 slave로 가게 된다.
- 다) Slave 쪽 message piggyback 없이 응답만 보내는 unnumbered frame.
- 라) Slave쪽 queue에 있는 message - Master로 가는 message. - 임의의 다른 slave로 가는 message

이 경우는 직접 message를 보내지 못하고 master를 경유하여 master에서 목표하는 slave로 보내주게 한다. 이 때는 응답을 받을 때 piggyback을 시키지 않는 것으로 한다. 이유는 계속 slave간에 전송만 담당하게 되는 최악의 상태에서 순환 순서가 진행되지 못하기 때문이다.

Ⅲ. Kuehn^[8]의 방법을 이용한 해석

그림4에서 보인 바와 같이 N_0 개의 slave가 $2N_0$ 개의 queue로 modeling되었다. 각 queue에 도착하는 message는 p.d.f가 $A_j(t) = P\{T_{A_j} < t\}$, $j=1, 2, \dots, 2N_0$ 인 General Independent Arrival Process를 따른다고 가정한다. 여기서 T_{A_j} 는 queue j에서의 임의의 message 도착시간 간격을 뜻하는 random variable이다. 따라서 도착률 λ_j 는 $\lambda_j = 1/ET_{A_j}$ 가 된다. 여기서 E는 평균을 뜻한다. 임의의 queue j에 있는 message가 받는 random service 시간을 T_{W_j} 라면 T_{W_j} 의 p.d.f $H_j(t)$ 는 $H_j(t) = P\{T_{W_j} < t\}$ 에 따른다고 한다. Service 방식은 한 queue에 message가 있는 경우 한 message만 service 하고 다음 queue로 넘겨주는 nonexhaustive service 방식을 취한다. 이때 queue j에서 다음 queue(j+1)로 service 받을 권한을 넘겨주는 switchover 시간을 T_{W_j} 로 정한다. T_{W_j} 의 p.d.f는 $U_j(t) = P\{T_{W_j} < t\}$ 이고 평균 $w_j = ET_{W_j}$, $j=1, 2, \dots, 2N_0$ 이다.

T_c 는 random cycle 시간이라 하고 c 는 평균 cycle 시간이라 한다. 즉 $c = ET_c$. 그리고 $c_0 = u_1 + u_2 + \dots + u_{2N_0}$ 으로 표시하면 한 cycle동안 service받는 message가 하나도 없다는 조건아래 평균 cycle 시간을 뜻한다. 이와같은 가정 아래 Kuehn의 해석법^[13]을 이용하여 임의의 queue j에서 message의 waiting 시간을 구하는 상세한 과정과 Model에의 적용은 참고문헌^[15]에 제시하고 있으며 여기서는 결과식만 제시하기로 한다.

$$w_j = \frac{c_j^{(2)}}{2c_j} + \frac{\lambda_j c_j^{(2)}}{2(1-\lambda_j c_j)} \quad (1)$$

여기서 w_j 는 queue j에서의 waiting시간을 의미하며 (2)는 2nd moment를 뜻한다. (1)식의 물리적인 의

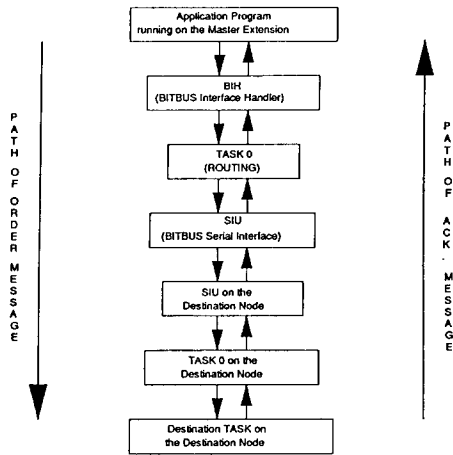


그림 5. BITBUS message 전송 경로
 Fig. 5. The transmission path of BITBUS message.

미를 알아보면 첫째 항 $c_i^{(2)}/2c_i$ 은 M/G/1 system 에서 평균 residual cycle 시간을 나타내고 두번째 항은 평균 queue 길이 곱 평균 cycle 시간을 나타낸 값이다.

IV. Boxma⁽⁹⁾의 방법을 이용한 해석

상세한 수학적 해석과 model에의 적용은 참고문헌⁽¹⁵⁾에 제시하고 있다. 앞 절 III의 Kuehn 방식과 구별되는 차이는 Kuehn은 조건 cycle 시간을 도입한 반면 Boxma와 Meister는 pseudo-conservation law⁽¹⁰⁾를 이용하여 해석한다는 점이다. 결과식으로 임의의 queue에 있는 message의 평균 waiting 시간 EW_i 는

$$EW_i = \frac{1-\rho+\rho_i}{1-\rho-\lambda_i s} \cdot \frac{1-\rho}{(1-\rho) \cdot \rho + \sum_{i=1}^{2N_0} \rho_i^2} \times \left[\frac{\rho}{2(1-\rho)} \sum_{i=1}^{2N_0} \lambda_i \beta_i^{(2)} + \frac{\rho}{2S} \sum_{i=1}^{2N_0} \psi_i^2 + \frac{s}{2(1-\rho)} \sum_{i=1}^{2N_0} \rho_i (1+\rho_i) \right], \quad i=1,2,\dots,2N_0 \quad (2)$$

여기서 ρ_i 와 ρ 는 queue i에서와 전체 utilization을 각각 나타내고 λ_i 는 queue i에서의 도착률을 뜻하고 s 는 전체 switchover 시간을 뜻한다. ψ_i^2 은 queue i에서 (i+1)까지의 switchover시간의 분산 값이다. 즉 $\psi_i^2 = E s_i^2 - (s_i)^2$.

V. Analysis와 Simulation 결과 비교

각 master와 slave에 도착하는 message는 Poisson arrival process를 따른다고 가정하였고 simulation slave수는 10개로 정했다.

결과 그림을 논의하기로 한다. 그림 6에서 볼 수 있는 세계의 Plot중에서 s라 표시된 것은비대칭적인 message 분포 상황에서 어떤 응답시간을 보여주는가를 알아보기 위해 simulation해 본 결과로서 simulation 환경은 master쪽 queue에서의 message 도착률은 일정하게 유지시키고 slave쪽 queue들 모두에 Poisson 도착률을 증가시킴에 따른 slave쪽 임의의 한 node에서 평균 waiting시간을 구해본 것이다. 이와는 달리 그림 7의 s라 표시된 것은 대칭인 message 분포상황에서 어떤 응답시간을 보여주는가를 알아본 것으로 simulation 환경은 master쪽과 slave쪽 모두에 똑같이 Poisson분포를 따르는 message의 도착률을 증가시키면서 slave쪽 임의의 한 node에서 message의 평균 waiting 시간을 보여준 것이다. 또한 그림6과 그림7에서 K와 B로 표시한 것은 각각 Kuehn과 Boxma의 analysis 결과를 나타내고 있다. Simulation과 비교해서 알 수 있듯이 high traffic에서 simulation 결과와 차이를 나타내고 있다. 이것은 Boxma와 Kuehn의 analysis과 정에서 각 queue들의 상호작용을 근사화시킨 것이 실

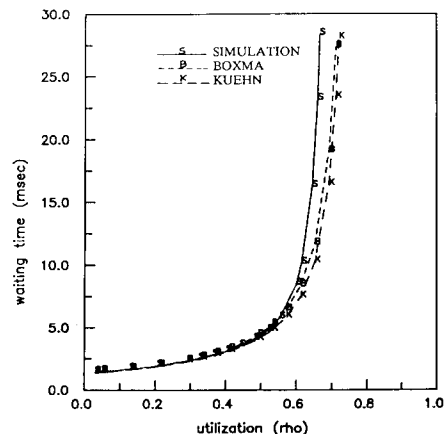


그림 6. Slave node의 traffic증가가 slave node의 평균 message waiting time에 미치는 영향 (Poisson arrival, 일정한 master node traffic, 같은 조건의 slave node)

Fig. 6. The effect of the slave node traffic intensity on the average message waiting time at a slave node. (Poisson arrival, fixed master node traffic and symmetric slave nodes).

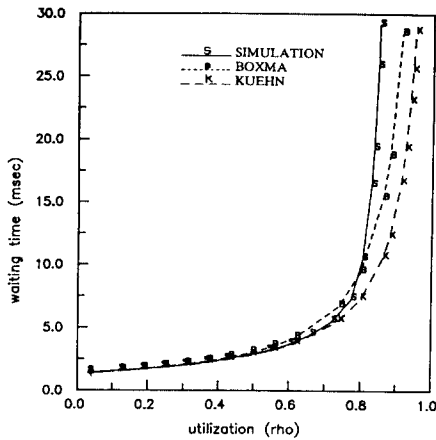


그림 7. Traffic증가가 평균 message waiting time에 미치는 영향(Poisson arrival, 같은 조건의 slave와 master node)

Fig. 7. The effect of the traffic intensity on the average message waiting time (Poisson arrival, symmetric slave and master node).

제 상황에서선 무리가 있는 것임을 보여주고 있다.

Boxma방식이 Kuehn방식에 비해 비대칭적으로 slave한 쪽만 변화시킨 그림 6보다 대칭적으로 slave, master쪽 모든 queue에 대해 같이 변화시킨 그림 7이 더 simulation 결과에 근사함을 보여주고 있다. 이것은 Boxma의 방식을 유도하는 과정에서 모든 queue의 message load가 대칭이란 가정을 도입하여 유도했으므로 대칭성이 커질수록 analysis 결과에 근사함을 알 수 있다.

VI. SIMULATION 결과 및 토의

지금까지의 알려진 방법으로는 analysis가 불가능 하지만 실제 상황에 근접하는 상황을 만들어 이 환경에서 BITBUS network의 성능이 어느 정도인가를 message의 waiting시간을 성능의 척도로 삼아 simulation을 행하였다. 우선 보다 실제 환경에 가깝게 하기 위해 각 master queue나 slave에 주기적으로 도착하는 message를 고려하였다. 이유는 실제 BITBUS의 응용면에서 실시간 message가 많고 data acquisition을 생각할 때 미리 정해진 program에 의해 일정한 시간 간격을 두고 계속해서 상태를 update시켜야 하는 경우가 많기 때문이다. 또한 cyclic-service-strategy에 있어서 모든 master쪽의 queue와 slave 쪽 queue에 동일한 priority를 주는 뜻으로 차례대로 한 cycle 동안 같은 횟수로 polling하는 경우와 어느 특정

한 slave에 message가 편중한 경우를 고려하여 한 cycle동안 특정 slave에 대해 polling을 다른 slave에 비해 한번더 해줌으로써 특정한 slave에 priority를 주는 경우를 고려했다.

Simulation 결과 그림을 논의해 보기로 한다. 두 cyclic-service-strategy를 비교하기 용이하게 같은 지면에 나타내었다. 이때 점선은 동일한 priority(균등이라 표시)를 준 경우를 뜻하며 실선은 특정한 queue에 priority를 준 경우(차등이라 표시)를 나타내기로 한다.

그림8과 그림9는 주기적인 message는 일정하게 영향을 미치는 환경에서, slave쪽 각각에 대해서만 Poisson arrival rate를 증가시킴에 따른 추이를 본 것이다. 주의해서 볼것은 IEC에서 제조자동화에 쓰이는 경우 허용하는 최대응답시간 5 msec를 최대 waiting time⁽¹⁾으로 지정했으므로 이때의 traffic intensity를 보고 network의 효용성을 추산할 수 있겠다.

Cyclic service strategy에 따른 비교로 traffic이 균등한 환경에선 당연히 priority를 주는 형태가 좋지 않음을 보여주고 있다. 하지만 그림9의 경우 차등 priority strategy에서 우선권을 가지는 특정한 station에 대응된 slave쪽의 queue에만 Poisson arrival rate를 변화시키고 다른 모든 queue는 고정시킨 상태에서의 결과이다. 여기서 보듯이 traffic이 편중한 경우 차등 priority를 주는 service strategy가 좋은 성능향

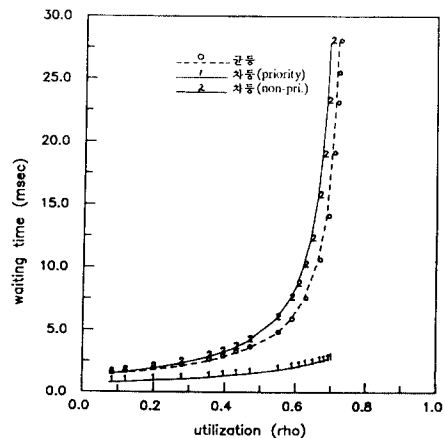


그림 8. Priority strategy와 non-priority strategy의 비교(Poisson arrival, 같은 조건의 slave node)

Fig. 8. The comparison of priority strategy and non-priority strategy. (Poisson arrival and symmetric slave nodes).

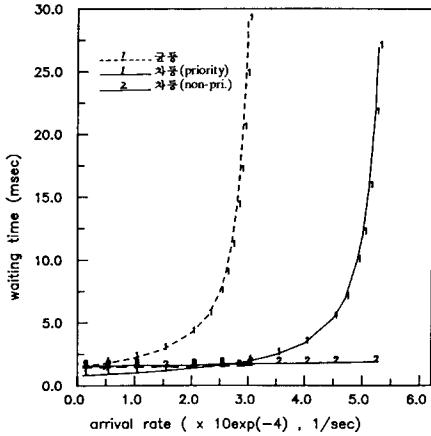


그림 9. Priority strategy와 non-priority strategy의 비교 (Poisson arrival, 우선순위를 갖는 한 slave node의 traffic 증가에 따라)

Fig. 9. The comparison of priority strategy and non-priority strategy. (Poisson arrival, according to increasing traffic intensity of one slave node with priority)

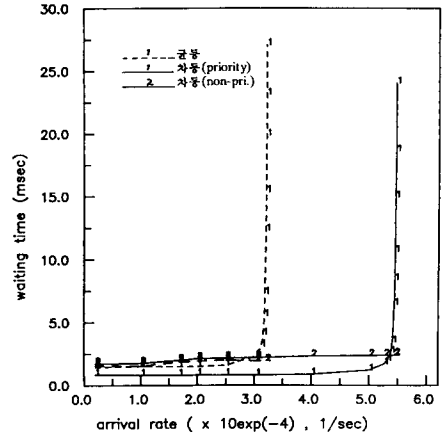


그림 11. Priority strategy와 non-priority strategy의 비교 (주기적인 traffic, 우선 순위를 갖는 한 slave node의 traffic 증가에 따라)

Fig. 11. The comparison of priority strategy and non-priority strategy. (Periodic traffic, according to increasing traffic intensity of one slave node with priority)

상을 보여주고 있다. 다음으로 그림 10과 그림 11의 경우는 Poisson 도착을 따르는 message는 일정하게 영향을 미치는 상태 아래 일정한 주기를 갖는 message의 도착주기를 변화시킴에 따른 그림이다. 변화 양상은 앞의 Poisson 도착을 따르는 message 경우와 비슷

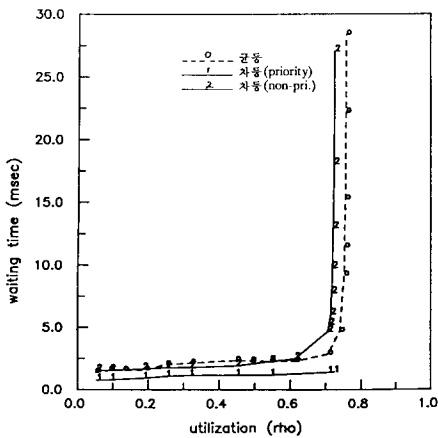


그림 10. Priority strategy와 non-priority strategy의 비교 (주기적인 traffic, 같은 조건의 slave node)

Fig. 10. The comparison of priority strategy and non-priority strategy. (Periodic traffic and symmetric slave nodes)

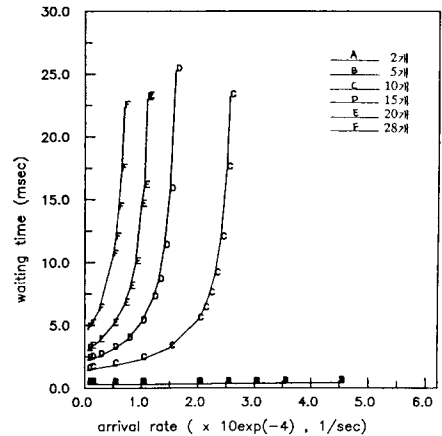


그림 12. Node수가 master node의 평균 message waiting time에 미치는 영향 (master node의 traffic 증가에 따라)

Fig. 12. The effect of the number of node on the mean message waiting time at master node. (According to increasing traffic intensity of the master node)

하므로 여기에선 언급을 피하고 Poisson을 따르는 그림 8과 그림 9 그리고, 주기적인 도착을 따르는 그림 10과 그림 11을 서로 비교하기로 한다.

주기적인 message의 특징적인 모습은 impulse 형태를 띠는 점이라 하겠다. 이것은 arrival process가 확률적인 것이 아니고 결정적인 형태이기에 나타난 결과이다. 즉 system의 허용 용량에 입력되는 message가 임계치에 도달하자마자 계속적으로 정해진 주기로 입력 message는 도착하므로 갑자기 폭주하는 형태가 나오는 것이다.

그림 12는 전송률을 2.4Mbps로 고정된 환경에서 slave의 수를 변화시키면서 waiting시간을 본 것이다. 결과에서 node가 10개까지는 제조자동화에 적합하고 그 이상은 제조자동화에는 곤란하고 공정자동화엔 쓸 수 있음을 알 수 있다.

Ⅶ. 결 론

공장현장에서 제조자동화나 공정자동화, 실시간을 중시하는 robotics등의 통신에 있어서 field bus의 일종인 BITBUS network이 어느 정도 유용한가를 message의 waiting 시간을 척도로 하여 해석과 simulation을 수행하였다.

먼저 Kuehn과 Boxma의 해석법을 이용하여 BITBUS queueing model을 해석하고 해석 환경과 같은 조건에서 simulation을 하여 서로의 결과를 비교하였다. 비교 결과에 의하면 low traffic인 경우 Kuehn 방식과 Boxma 방식 모두 simulation결과와 거의 일치함을 보여주고 있고 high traffic으로 갈수록 두 방식 모두 simulation결과와 차이가 남을 보여주고 있다. 또한 high traffic에서 각 station마다 traffic이 대칭으로 분포할 수록 Boxma방식이 Kuehn방식보다 더 simulation결과에 접근함을 알 수 있었다.

그리고 공장 현장에서 보다 더 가까운 환경으로 주기적인 message를 도입하고 cyclic service strategy를 달리하며 simulation을 하였다. simulation 결과에 의하면 message의 길이가 짧고 빈번히 발생하는 즉 주기적으로 발생하는 공장 현장은 D/D/1 queue로 model될 수 있음을 볼 수 있었고 따라서 임계 traffic 이전까지는 안정적인 waiting시간을 나타낸다. 또한 traffic이 특정 node에 편중된 경우 한 cycle동안 특정 node를 더 polling함으로 priority를 주는 차등 priority cyclic service strategy를 택하는 것이 유리함을 알 수 있었다. 그리고 IEC에서 규정한 제조 자동화에 요구되는 5msec응답시간에 부응하기 위해선 2.4Mbps의 전송률로 10개의 node까지 가능함을 알 수 있었다.

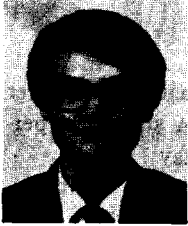
공장에서 field bus가 위치한 machine 계층 위의 cell 계층이나 그 위의 plant 계층의 통신형태인 Mini-MAP

이나 MAP과의 연결이 이루어져야 한다. 앞으로의 연구과제로 gateway의 실제적인 구현이 이루어져야 했고 구체적인 응용을 택하여 실제 hardware로 구현해 보는 것도 의미가 크다고 생각한다.

참 고 문 헌

- [1] A. Kusiak and S.S. Heragu, "Computer integrated manufacturing: A structural perspective," *IEEE Network*, May 1988.
- [2] J. Voelcker, "Helping computers communicate," *IEEE SPECTRUM*, Mar. 1986.
- [3] Special Edition, "Manufacturing automation protocol," *Control Engineering*, Oct. 1986.
- [4] P. Pleinevaux and J.D. Decotignie, "Time critical communication networks: field buses," *IEEE Network*, May 1988.
- [5] L. Sintonen and T. Virvalo, "The hierarchy of communication networks in the programmable assembly cell: an experimental framework," *IEEE Network*, May 1988.
- [6] M. Kazahaya, "Field bus: new standard?," *Control Engineering*, Oct. 1987.
- [7] Intel Corp., *IDCX 51 Distributed Control Executive User's Guide*, 1987.
- [8] P.J. Kuehn, "Multiqueue Systems with nonexhaustive cyclic service," *The Bell Sys. Tech. Journal*, Mar. 1979.
- [9] O.J. Boxma, B.W. Meister, "Waiting time approximations for cyclic-service systems with switchover times," *Performance evaluation* 7, 1987.
- [10] O.J. Boxma and W.P. Groenendijk, "Pseudo-conservation laws in cyclic-service systems," *Journ. of Appl. Prob.* 24, 1987.
- [11] L. Kleinrock, *Queueing Systems vol. I*, John Wiley & Sons, 1975.
- [12] R.W. Wolff, "Poisson arrivals see time averages," *Operat. Res.* 30, 1982.
- [13] IEEE Std 802.4-1985, *Local Area Network Standard-Token Passing Bus Access Method and Physical Layer Spec.*
- [14] G.M. Birtwistle, O-J Dahl, B. Myrhaug and K. Nygaard, *SIMULA BEGIN*, Studentlitterature, 1976.
- [15] 성백문, "Field Bus로서의 BITBUS Network에 대한 성능 연구", 석사학위논문, 한국과학기술원 전기 및 전자공학과, 1989.

著 者 紹 介



成 百 文 (正會員)

1963年 3月 13日生. 1987年 2月
서울대학교 제어계측공학과 졸업
공학사학위 취득. 1989年 2月
한국과학기술원 전기 및 전자공
학과 졸업 공학석사학위 취득.

1989年 3月~현재 현대전자산업
주식회사 산업전자영업본부 시스템 소프트웨어 개발
부. 주관심분야는 데이터통신 등임.



林 東 民 (正會員) 第26卷 第3號 參照

현재 한국과학기술원 전기및
전자공학과 박사과정 재학중

李 滉 洙 (正會員) 第26卷 第3號 參照

현재 한국과학기술원 전기및
전자공학과 조교수



殷 鍾 官 (正會員) 第26卷 第3號 參照

현재 한국과학기술원 전기및
전자공학과 교수. 디지콤 정
보통신연구소 고문