PC 기반 장비 컨트롤러의 직렬통신 부하에 관한 연구

A study of serial communication load for PC-based equipment controllers

류 광 열, 정 무 영
산업공학과/제품생산기술연구소
기계산업공학부, 포항공과대학교

Abstract

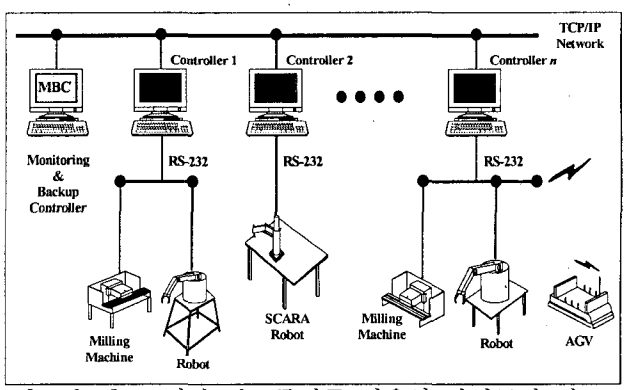
급변하는 소비자의 욕구를 만족시키고 이에 신속하게 대처하기 위해 Agile manufacturing 이나 Flexible manufacturing system 에 관한 연구가 활발히 전개되어 왔다. 또한 최근에 급속한 컴퓨터 관련 하드웨어의 발달로 인해 시스템 내부에 사용되는 장비 제어를 PC based controller 로 바꾸려는 움직임이 일고 있다. 이러한 PC based controller 가 전용 컨트롤러의 모든 기능을 수행하기란 아직은 한계가 있으나 점차 그 사용이 보편화될 것이며, 더욱이 컴퓨터 하드웨어의 놀라운 기술 발전으로 인해 앞으로는 여러 대의 장비를 한 대의 컨트롤러로 제어할 수도 있을 것이다.

본 연구에서는 이러한 환경에 민첩하게 대응하기 위해서 우선 각 장비별로 컨트롤러와의 통신을 위한 message 를 종합하고 이를 체계적으로 분류하도록 하며, 이렇게 분류된 각 message 들이 PC 기반의 컨트롤러에 줄 수 있는 load 를 계산하는 방법을 제시한다. 본 연구를 통해 컨트롤로의 각 장비를 제어함에 있어서 걸리는 작업 부하를 수치적으로 계산해 냄으로써 앞으로 제어할 장비를 무리 없이 컨트롤하기 위한 최적의 컨트롤러를 구성하는데 도움이 될 것이다.

1. 서론

급속도로 발전하는 여러 기술 요소로부터 생산 현장의 자동화를 실현하기 위해서 시스템의 architecture 에 관한 연구 및 이들의 실현 방안에 대한 연구는 꾸준히 이루어져 오고 있다. 전용 machine 을 포함한 shop 의 자동화 문제에 있어서 가장 걸림돌이 되는 것이 바로 이러한 machine 을 다루는 작업자의 개입 문제이다. 어쩔 수 없이 작업자의 개입이 필요한 작업이 발생할 가능성이 있을수록 완전 자동화는 그만큼 어려워지게 된다. 최근의 컴퓨터 관련 하드웨어의 눈부신 발전에 힘입어 이러한 문제는 각 machine 의 controller 를 PC 로 대체함으로써 어느 정도 이러한 단점을 보완할 수 있다. 즉, GUI 를 통해 사용자의 편의를 도모하고, 작업자만의 intelligence 를 controller 에 이식시킬 수도 있으므로 기존의 closed-architecture 를 갖는 controller 를 open-architecture 로 변경할 수 있는 기회가 마련되었다. 또한, computer network 관련 기술을 shop 내부에 존재하는 controller 간의 communication 을 위한 제반 기술로 사용할 수 있어 시스템 통합 및 재구성이 더욱 쉽게 이루어질 수 있게 되었다. 이렇게 machine 을 제어하는 controller 가 PC-based controller 로 바뀌었을 경우 생각해 볼 수 있는 또 다른 장점은 바로 최근의 PC 운영체제가 다중작업(multi-tasking)을 지원한다는 사실에서 비롯될 수 있다. 즉, 한 대의 controller 를 이용하여 여러 대의 장비를 동시에 제어하는 것이 쉬워졌다는 것이다. [그림 1]은 이러한 환경 하에서의 FMS 내부의 장비 구성의 예를 보여주고 있다.

[그림 1]과 같이 한 대의 controller 로 여러 대의 장비를 제어함에 있어서 안정적인 시스템 운영을 위해서는 일단 각 equipment 를 controller 에 할당함에 있어 일관된 정책이 필요하게 된다. 가령 한 대의 controller 에 너무 많은 장비를 할당하게 된다면 전체 시스템을 구성하기 위한 비용은 줄일 수 있으나 시스템 운영이 불안정해지고 예상치 못한 치명적인 상황이 전개될 수도 있을 것이다. 또한 불균형한 장비의 할당으로 controller 의 utilization 이 떨어져 결국 시스템 performance 를 감소시키는 원인이 될 수도 있다. 결국 이러한 상황에서는 할당된 장비로 인한 controller 의 load balance 를 맞추기 위한 작업이 필요하게 된다.



[그림 1] PC 기반 컨트롤러를 이용한 장비구성 예

Network load 의 balance 를 균형 있게 하기 위해 load-sharing 이나 load balancing 이라 일컫는 연구가 지금까지 많은 학자들에 의해 연구되어 왔다[1-3]. 그러나 이들이 말하는 load 란 주로 TCP/IP protocol 이나 범용적으로 쓰이는 communication protocol 을 이용한 controller 간의 communication load 에 대해 다루고 있다. 즉, controller 의 serial port 를 이용한 equipment 와의 통신으로 인해 발생하는 load 를 고려하지는 않았다는 것이다. 각 장비가 특정 controller 에 할당되어 제어될 때 controller 에 주는 load 를 예측하여 이를 기준으로 장비를 할당하는 것은 [그림 1]과 같은 장비구성을 갖는 시스템을 구성하기 위해 반드시 필요한 작업이라 하겠다. 따라서 본 연구에서는 예제를 통해 각 장비별로 controller 와 주고 받는 control message 를 분류하고, 이를 이용하여 각 장비로부터 controller 에 발생하는 load 를 실제로 계산해 보기로 한다.

2. 장비 제어를 위한 메시지

한 대의 장비를 자동화 하는 것은 그다지 어려운 일은 아니다. 그러나, 공장 전체를 효과적으로 자동화 하는 문제는 시스템 통합이라는 커다란 걸림돌이 존재하게 된다. 특히 이 부분에 있어서 서로 규격이 다른 회사에서 생산된 장비들을 통합한다는 것은 더더욱 어려운 일이 될 것이다. 이러한 문제를 해결하기 위해 MMS (Manufacturing Message Specification)라는 상이한 환경에서 제작된 각 장비들 사이에 주고 받는 메시지의 규격이 MAP (Manufacturing Automation Protocol)의 일환으로 제정되어 국제 규격으로 승인되었다[4]. MMS 규격은 생산 장비들의 외부로의 기능을 소프트웨어적인 측면에서 표현한 것으로써[5], 주요 기능을 key-attribute 로 기타 vendor name, model name, revision number 등을 attribute 로 갖게 된다.

그러나, 이러한 규격에 맞추어진 장비들을 운영함에 있어 PC-based controller 를 사용함으로 인해 얻을 수 있는 장점 중의 하나는 여러 명령이 모여 하나의 macro 한 작업을 수행하는 control message 를 정의할 수 있다는 것이다. 생산 시스템 내부에 존재할 수 있는 메시지는 primitive 스타일(예를 들어, *load*, *unload*, *pick* 등)과 macro 스타일(예를 들어, *move*, *refixture* 등)의 두 가지가 존재한다[6]. 본 연구에서도 이러한 두 가지 스타일에 따라 메시지 타입에 따라 발생하는 로드를 달리 계산하기로 한다.

장비로부터 controller 에 발생하는 load 는 크게 장비와 컨트롤러 사이의 메시지와 컨트롤러 간의 메시지로 나눌 수 있으며, 장비와 컨트롤러 사이의 메시지는 다시 컨트롤러로부터 장비로 보내지는 control message 와 장비로부터 컨트롤러로 보내어지는 information message 로 나눌 수 있다. 또한 각 메시지는 그 안에 파일을 포함하고 있는지의 여부도 load 에 영향을 미치게 된다. 이러한 요소를 고려하여 컨트롤러와 장비 사이에 존재하는 메시지를 크게 3 가지의 type 으로 나눌 수 있다.

- MC2E(F) : macro message from controller to equipment (including files)
- PC2E(F) : primitive message from controller to equipment (including files)
- E2C(F) : message from equipment to controller (including files)

Controller 에 발생하는 communication load 는 장비와의 통신 과정에서 발생하는 메시지의 length 나 작업처리 요청시간(response time)과도 관련이 있다. 여기서 말하는 메시지의 length 는 그 메시지가 포함하고 있는 parameter 의 수를 나타낸다. 특히 MC2E(F) 타입의 메시지에는 다수의 primitive 메시지가 포함될 수 있으므로 각 primitive 메시지의 parameter 를 모두 더해야만 length 를 알 수 있다. 또한 MC2E(F) 타입의 메시지는 여러 메시지가 모여 하나의 작업이 이루어지는 만큼 각 primitive 메시지를 처리하는 동안 외부의 개입으로 인한 작업의 중단이 이루어져서는 안되므로 다른 primitive 메시지와 구별하여 더 높은 priority 를 가져야만 한다. 특히 시스템 에러가 발생하여 모든 가공을 중단할 경우에 대한 일처리는 상당히 시급하므로 가장 높은 priority 를 가져야만 한다.

3. 직렬 통신 로드

Shop floor 에 배치된 controller 에 발생하는 communication load 는 비록 같은 장비의 조합으로 이루어졌다고 해도 가공을 위한 process plan 에 따라 달라질 수 있다. 복잡하고 많은 수의 장비가 동원되는 part 의 가공은 장비와 controller 사이에 더욱 빈번한 메시지의 교환이 이루어질 것이기 때문이다. 제 2 장에서 message 와 관련된 load factor 는 식 (1)과 같이 계산될 수 있다.

$$ML_i = \sum_{\substack{msg.in \\ process\ plan}} P \times (L + F) \qquad (1)$$

ML_i 는 장비 i 의 message 로 인해 발생되는 load 를 말한다. 하나의 메시지로부터 발생되는 load 는 L (message length)과 F (file factor)의 합으로 표현되며 이 값은 P (priority factor)에 따라 load 의 양이 달라지게 된다. P 값은 1, 2, 3 으로 나누어지며 macro 스타일의 메시지 내부에 존재하는 각 primitive 메시지는 2 의 값을, 시스템 내부에 발생하는 에러 메시지는 3 의 값을, 그 밖의 모든 메시지는 1 의 값을 갖는다. F 값은 다루는 파일의 크기에 비례하여 그 값을 부여한다.

[그림 1]과 같은 환경을 만들기 위해서는 컨트롤러 사이의 communication 에 의해 발생되는 load 도 고려를 해야 한다. 그러나 이러한 문제는 본 연구의 범위를 벗어나므로 본 연구에서는 장비 i 를 control 하기 위해 다른 컨트롤러와의 communication 으로 인해 발생되는 load 값을 CL_i (controller load)라 하여 이 값은 ML_i 와 비례하는 값으로 취급하여 고려하였다.

그 밖에 직렬통신으로 인해 컨트롤러에 발생하는 load 에 중요하게 영향을 미치는 요소가 바로 컨트롤러의 CPU 속도와 연결되는 장비의 수, 즉 사용되는 직렬포트의 개수이다. 컴퓨터의 처리 속도에는 물론 CPU 의 속도 이외에 사용하는 메모리의 양이라든지 I/O resource 등도 영향을 줄 수가 있지만 이들은 대부분 CPU 의 속도에 맞춰 선택되므

로[7] 본 연구에서는 CPU 의 속도만을 고려하도록 한다. 즉, CPU 의 속도가 느릴수록 같은 작업에 대해 더 큰 communication load 가 발생된다는 것이다. 또한 컨트롤러의 사용 가능한 직렬포트 개수가 많을수록 연결할 수 있는 장비의 수가 많아지므로 컨트롤러에 발생하는 load 가 커짐은 당연히 짐작될 수 있다.

이상을 종합해 보면 한 대의 PC-based controller 에 발생하는 communication load 는 식(2)와 같이 정리할 수 있다.

$$ControllerLoad = \frac{\alpha}{CPU} \times \left(COM \times \sum_i ML_i + \sum_i CL_i \right) \quad (2)$$

식(2)를 보면 결국 한 대의 controller 에 발생하는 communication load 에 영향을 미치는 요소로는 연결된 모든 장비와의 통신 과정에서 발생하는 메시지와 컨트롤러 사이의 메시지, 컨트롤러의 CPU 속도와 사용 가능한 직렬포트의 개수로 요약될 수 있다. 계산되는 communication load 값은 사용되는 CPU 의 속도에 반비례하고 COM 포트(serial port)의 수에 비례하게 된다. 사용되는 직렬 포트에 영향을 받는 요소는 장비와의 통신 과정에서 발생하는 메시지로 인한 load 이며, 컨트롤러 간의 통신에는 영향을 미치지 않는다. 여기서 α값은 시스템 내부의 stability 를 나타내는 factor 로써, 불안정한 시스템에 사용되는 컨트롤러에는 보통 이 값을 1 보다 큰 값을 사용한다. 이렇게 하면 연결된 장비로부터 발생하는 load 가 커지므로 결국 컨트롤러에 더 적은 load 를 발생시키는 다른 장비로 대체시키거나 아예 연결된 장비 수를 줄이게 되어 원활한 시스템 운영을 할 수 있게 한다.

4. 적용 예제

본 장에서는 앞에서 언급된 직렬통신 로드 계산 방법을 실제 적용해 보도록 한다. Part 의 가공을 위한 FMS 에 존재하는 메시지를 타입별로 분류하고 이를 기초로 각 컨트롤러에 장비와의 통신으로 인해 발생되는 load 를 계산해 본다.

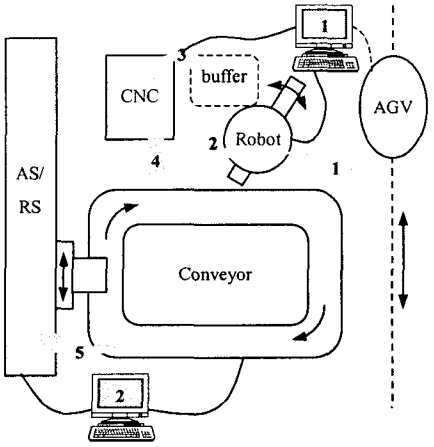
4.1 System Environment

[그림 2]와 같은 구성을 갖는 FMS 에는 총 5 대의 장비가 있으며 이는 2 대의 컨트롤러에 의해 제어된다고 한다. Controller1 은 AGV, CNC machine, robot 을, controller 2 는 AS/RS 와 conveyor 를 각각 동시에 제어하고 있다.

Part 의 가공을 위한 시나리오는 다음과 같다.

- AGV 를 통해 운반된 part 는 robot 에 의해 buffer 에 옮겨진다. ([그림 2]의 1, 2 과정)
- CNC machine 이 idle 상태이면 robot 이 buffer 에 있는 part 를 CNC machine 으로 옮겨 가공을 시작한다. ([그림 2]의 3 과정)
- CNC 에서 가공이 끝난 경우 robot 이 idle 하면 가공된 part 를 conveyor 로 옮기고 그렇지 않으면 buffer 에 옮겨 robot 이 idle 상태일 때까지 기다린다. ([그림 2]의 4 과정)
- Conveyor 에 옮겨진 part 는 AS/RS 의 지정된 bay 에 저장된다. ([그림 2]의 5 과정)

각 컨트롤러는 200Mhz 의 CPU 사양을 갖고 있으며 최대 3 개의 COM port 를 사용할 수 있다고 가정한다. 모든 장비는 controller 와 RS-232/422 protocol 을 이용하여 직렬통신을 하며 각 컨트롤러 사이의 통신은 TCP/IP protocol 을 사용한다.



[그림 2] PC-based 컨트롤러를 이용한 FMS 의 예

4.2 메시지의 분류

Part 의 machining 을 위한 장비 및 material handling 을 담당하는 장비와 controller 사이에 존재하는 메시지는 [표 1]과 같이 간략히 분류될 수 있다.

[표 1] 적용 예제의 메시지 분류

Equipment	Message	Type	L	P
AGV	*Send_status*	PC2E	2	1
	Send_status_done	E2C	1	1
	Stop_to_unload	PC2E	1	1
	Stop_to_unload_done	E2C	1	1
	Unload_part	MC2E	4	1
	Unload_part_done	E2C	1	1
	Wait_unload	PC2E	1	1
	Move_position	PC2E	1	1
	Reset	MC2EF	5	1
	Error	E2C	1	3
Robot	*Send_status*	PC2E	2	1
	Send_status_done	E2C	1	1
	Move_part	MC2E	10	1
	Move_part_done	E2C	1	1
	Reset	MC2EF	4	1
	Error	E2C	1	3
CNC	*Send_status*	PC2E	2	1
	Send_status_done	E2C	1	1
	Load_part	MC2E	5	1
	Load_part_done	E2C	1	1
	Unload_part	MC2E	4	1
	Unload_part_done	E2C	1	1
	Machine_part	MC2EF	80	1
	Machine_part_done	E2C	1	1
	Reset	MC2EF	10	1
	Error	E2C	1	3

Equipment	Message	Type	L	P
Conveyor	*Up_pallet*	MC2E	4	1
	Up_pallet_done	E2C	1	1
	Down_pallet	MC2E	4	1
	Down_pallet_done	E2C	1	1
	Reset	MC2EF	3	1
	Error	E2C	1	3
AS/RS	*Store_part*	MC2E	15	1
	Store_part_done	E2C	1	1
	Reset	MC2EF	5	1
	Error	E2C	1	3

[표 1]에서 계산된 각 메시지의 length (L) 값은 메시지에 포함되는 parameter 의 개수로 정함을 원칙으로 하나 parameter 가 없는 메시지, 예를 들어 *Send_status_done*, *Move_part_done* 과 같은 메시지들은 length 를 1 로 계산한다. 또한 macro 스타일의 메시지는 priority 값이 1 이지만 그 안에 포함된 메시지는 priority 값을 2 로 하여 load 를 계산하도록 한다. 가령, AS/RS 의 *Store_part* 라는 MC2E type 의 메시지에는 *Move_to_pickup_point*, *Open_gripper*, *Index_down*, *Close_gripper*, *Delay_for_grip_orientation*, *Index_up*, *Move_through_hover_point*, *Move_to_the_drop_point*, *Move_carrage_to_bay*, *Move_carrage_out_of_bay* 와 같은 priority 2 를 갖는 메시지가 중복되어 포함되어 있다.

[그림 2]에서 사용된 각 컨트롤러의 장비와의 직렬통신으로 인한 load 는 식(1)과 (2)에 의해 다음과 같이 계산될 수 있다. 이때 사용된 CPU speed 에 대한 factor 는 2 로, α 값으로 1.5 를 사용하였다.

$$Controller1\_load = \frac{1.5}{2}(3 \times 281 + 281) = 843$$

$$Controller2\_load = \frac{1.5}{2}(2 \times 49 + 49) = 110.25$$

위의 결과에서도 알 수 있듯이 controller 1 과 controller 2 의 communication load 는 큰 차이를 보여 장비의 할당이 제대로 이루어지지 않았음을 알 수 있다. 따라서, controller 1 에 AGV, robot, AS/RS 를 재배치하고, controller 2 에 CNC machine 과 conveyor 를 할당했을 경우의 각 컨트롤러에 발생하는 load 값은 마찬가지 방법으로 계산했을 경우 각각 384 와 454.5 의 값을 나타내게 된다. 재배치가 이루어지기 전에 비해서 두 컨트롤러 사이의 load 를 어느 정도 균형있게 배분했으므로 더욱 원활한 시스템 운영이 예상된다.

5. 결론 및 추후 연구과제

본 연구에서는 한 대의 컨트롤러로 여러 대의 장비를 제어할 수 있는 구조를 갖는 FMS 내에서 각 컨트롤러에 할당된 장비와의 직렬통신으로 인해 발생하는 communication load 에 대해 알아보았다. Part 의 가공을 위한 process plan 을 이용하여 시스템 내부에 존재할 수 있는 메시지를 정의, 이를 분류하였으며, 이를 바탕으로 각 장비별로 컨트롤러에 줄 수 있는 load 를 계산하는 방법을 도출하였다. 직렬통신으로 인해 발생하는 load 에 영향을 미치는 요인으로는 시스템 내부에 존재하는 메시지의 양과 각 메시지의 priority, 메시지에 딸린 file 의 존재 여부를 들 수 있으며, 이러한 값들은 장비가 할당되는 컨트롤러의 CPU speed 와 사용되는 직렬포트의 수, 그리고 시스템의 안정성을 나타내는 factor 에 영향을 받는다.

그러나, CPU speed 에 대한 factor 를 결정하는 기준이라든지 시스템의 stability 를 나타내는 α값을 정하는 문제 등은 앞으로 반복적인 실험을 통해 그 값을 얻어내야 할 것이다.

본 연구를 통해, [그림 1]과 같은 환경을 구축하려 할 경우 직렬통신으로 인해 컨트롤러에 발생하는 communication load 를 미리 계산해 봄으로써 장비의 할당 문제라든지 최적의 컨트롤러를 구성함에 있어 도움을 얻을 수 있을 것으로 기대된다

Acknowledgement

본 논문은 1995 - 1998 년도 한국학술진흥재단 연구비의 일부지원으로 연구되었습니다. 지원에 감사드립니다.

References

[1] D.L. Eager, *et al.*," Adaptive load sharing in homogeneous distributed systems" , *IEEE Transactions on Software Engineer*, SE-12(5), pp.662-675, 1986

[2] P. Krueger and M. Livny, " The diverse objectives of distributed scheduling policies" , *Proc. Of the 7th International Conference on Distributed Computing Systems*, pp.242-249, 1987

[3] S. Zhou, and D. Ferrari, " A measurement study of load balancing performance" , *Proc. Of the 7th International Conference on Distributed Computing Systems*, pp.242-249, 1987

[4] S. G. Shanmugham, *et al.*," Manufacturing communication: a review of the MMS approach" , *Computers ind. Engng*, Vol. 28, No. 1, pp.1-21, 1995

[5] J. Caven and J. Jackman, " Icon based approach system control development" , *IEEE Trans. Ind. Electron.*, 37, pp.259-264 1990

[6] H. Cho, " Petri net models for message manipulation and event monitoring in an FMS cell" , *Int. J. Prod. Res.*, Vol.36, No.1, pp.231-250, 1998

[7] Robert Leslie and Sati McKenzie, " Evaluation of loadsharing algorithms for heterogeneous distributed systems" , *Computer Communications*, 22, pp.376-389, 1999