Proceedings of the 14th
KACC, Octorber 1999

Supervisory Control Theory의 적용을위한 DES Testbed의 구현

Implementation of DES Testbed to investigate Supervisory Control Theory

°손 형 일*, 이 석**

* 부산대학교 대학원 지능기계공학과(Tel : 82-051-510-3091; E-mail : chakurt@nownuri.net)
** 부산대학교 기계공학부(Tel : 82-051-510-2320; E-mail : slee@hyowon.pusan.ac.kr)

Abstract : A discrete event systems (DES) is a physical system that is discrete in time and state space, asynchronous (event rather than clock-driven), and in some sense generative(or nondeterministic). In this paper we construct an experimental CIM system to investigate supervisory control theory. We designed 15 modular supervisors for the testbed. Thess supervisors are nonblocking, controllable and nonconflicting. After verification of the supervisors by simulation, the supervisors for AGV system have been implementd to demonstrate their efficacy.

Keywords : Discrete event systems, Supervisory control theory, Modular supervisor, Manufacturing cells

1. 서론

Discrete Event Systems (DES)에 대한 모델링과 제어기의 설계는 미분방정식과 같은 전통적 제어이론으로는 많은 어려움이 있다. 그래서 이런 DES를 제어하기 위한 새로운 방법론이 필요하게 되었고, 근래에 새로운 모델링 기법과 제어 이론들이 많이 소개되고 있다. Petri net, formal language, controlled automata, min-max algebra, temporal logic 등이 그 대표적인 예라고 할 수가 있다. 그렇지만 이런 이론들은 각각 별개의 것이 아니라, logic, language, automata 이론을 바탕으로 서로 관련되어 있다[3]. 이런 여러 가지 방법론들중에서 automata와 lattice 이론을 기반으로 하는 Ramadge, Wonham 교수의 supervisory control theory가 플랜트의 구속조건을 만족시키고 실제 일어날 수 있는 이벤트들을 최대한 허용시킴으로써 많은 관심을 받고 있다[3].

본 논문에서는 supervisory control theory를 실제 시스템에 적용하기 위해 CIM system을 모사하는 supervisory control testbed를 제작하였다. Supervisory control testbed는 3대의 로봇, 2대의 AGV, NC 공작기계, 컨베이어 벨트 등으로 구성되어 져있다[8]. 이렇게 제작된 각각의 구성요소들을 플랜트로 두고 deterministic automaton으로 모델링하였고, CIM system의 운용규칙을 구속조건(behavior specification 또는 legal language)으로 하여 슈퍼바이져를 설계하였다. 그리고 슈퍼바이져의 controllability, nonblockingness를 검사하였다. 또 슈퍼바이져를 modular로 설계하여 modular 슈퍼바이져들이 nonconflictness를 만족하는지 검사하였다[2][6].

마지막으로 각각의 modular 슈퍼바이져들을 구현을 위해서 Clocked Moore Synchronous State Mchine(CMSSM)으로 변환하였다[2][5]. 그리고 CMSSM을 회로설계 및 분석 프로그램인 PSpice로 시뮬레이션하여 legal language에 맞게 플랜트가 작동하는지 검사하였다. 특히 AGV 충돌방지 슈퍼바이져는 실제 제작하여 AGV 시스템을 원활하게 제어하는 것을 확인하였다.

2. Supervisory control theory

2.1 Generator

DES를 모델링하기 위한 automaton은 다음과 같은 5개의 구성요소를 가지고 있다.

$$G=\{Q,\ \Sigma,\ \delta,\ q_0,\ Q_m\} \tag{1}$$

여기서 Q는 상태들의 집합, Σ는 이벤트의 집합, δ는 $\delta : Q\times\Sigma^*\rightarrow Q$ 인 천이함수, q_0는 초기 상태, Q_m은 작업의 완료를 나타내는 marked 상태들의 집합이다. 그리고 Σ는 controllable 이벤트 집합 Σ_c와 uncontrollable 이벤트 집합 Σ_u로 나눌 수 있다.

이렇게 구성된 automaton이 생성해내는 언어는 $L(G)$로 나타내며 다음과 같이 정의된다.

$$s\in L(G) \Leftrightarrow s\in\Sigma^*,\ \delta(q_0,\ s)! \tag{2}$$

여기서 Σ^*는 이벤트의 sequence(string)를 말하며, $\delta(q_0,\ s)!$는 q_0에서 스트링 s가 일어난 다음의 상태가 정의됨을 나타낸다.

$L(G)$의 prefix closure를 $\overline{L(G)}$로 나타내고 정의는 다음과같다..

$$\overline{L(G)}=\{t\in\Sigma^* \mid t\leq s \text{ for } \textit{some } s\in L(G)\} \tag{3}$$

그리고 automaton G의 marked 언어를 $L_m(G)$로 나타내고

$$s\in L_m(G) \Leftrightarrow \delta(q_0,\ s)!\in Q_m\ ,\ L_m(G)\subseteq L(G) \tag{4}$$

와같이 정의된다. 이때 G가 $\overline{L_m}(G)=L(G)$를 만족하면 $L(G)$를 nonblocking하다고 말한다. 즉 G의 모든 상태에서 임의의 스트링이 일어난 후 marked 상태로 도달할 수 있음을 의미한다. 그리고 이는 supervisory control theory에서 proper 슈퍼바이져가 되기위한 중요한 필요조건이 된다[6].

2.2 Supervisor

슈퍼바이져 역시 다음과 같은 5개의 구성요소를 가진 automaton으로 표현된진다.

$$S=\{X,\ \Sigma,\ \xi,\ x_0,\ X_m\} \tag{5}$$

G를 주어진 플랜트라고 하면 슈퍼바이져 S아래에서의 G의 거동은 다음과 같이 나타난다.

$$S/G=\{X\times Q,\ \Sigma,\ \xi\times\delta,\ (x_0, q_0),\ X_m\times Q_m\} \tag{6}$$

그리고 G에 대한 $L(S)$의 controllability는 다음조건으로 검사될 수 있다.

$$(\forall s,\ \sigma)s\in\overline{L}(S),\ \sigma\in\Sigma_u,\ s\sigma\in L(G) \Rightarrow s\sigma\in\overline{L}(S) \tag{7}$$

즉 슈퍼바이저 S가 플랜트 G에 대해서 controllable하다는 것은 다음과 같이 설명할 수 있다. 슈퍼바이저 S에서 허용되는 임의의 스트링 s가 있고 플랜트 G에서 발생할 수 있는 어떤 uncontrollable 이벤트 σ가 있다고 했을 때 스트링 $s\sigma$가 플랜트 G 에서 발생했을 때 슈퍼바이저 S도 스트링 $s\sigma$가 일어날 수 있도록 허용한다면 슈퍼바이저 S는 플랜트 G에 대해서 controllable하다고 한다. 그리고 이렇게 controllable한 여러 슈퍼바이저중에서 supremal language가 optimal 슈퍼바이저가 된다[5][6]. 그림 1에 supervisory control system의 구조를 간략하게 나타내었다[3].

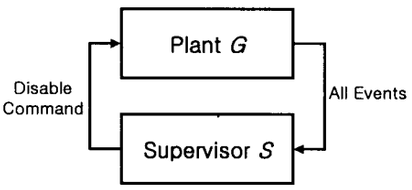

그림 1. 감독 제어 시스템

Fig 1. Supervisory control system

2.3 Modular Supervisor

슈퍼바이저 S를 $S_i, i=1,2\ldots$로 설계할 때 이런 S_i를 modular 슈퍼바이저라한다. 그리고 modular 슈퍼바이저는 다음과같은 nonconflictness 조건을 만족하여야한다[6].

$$\overline{S}=\overline{S_1}\wedge\overline{S_2}\wedge\ldots \tag{8}$$

Modular 슈퍼바이저가 nonconflictness를 만족하면, 모든 modular 슈퍼바이저들이 동시에 플랜트를 supervisory control하여 centralized 슈퍼바이저와 같은 작용을 할 수 있다. 만약 각 슈퍼바이저들이 conflicting하다면, modular 슈퍼바이저들은 centralized 슈퍼바이저가 허용하지않는 스트링을 허용하게 되어 legal language를 만족시키지 못하게된다.

3. Supervisory Control Testbed

3.1 시스템 구성

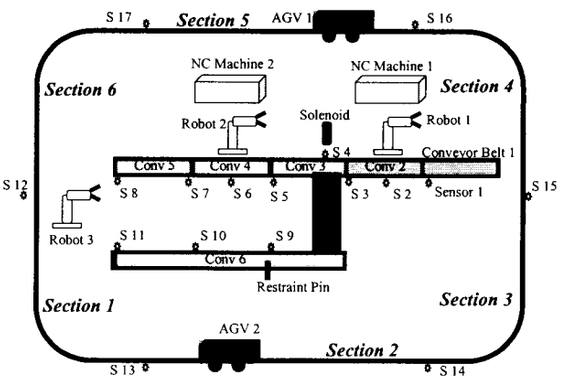

그림 2. 실험 시스템 구성

Fig 2. Supervisory control testbed

본 논문에서는 supervisory control testbed로써 NC 공작기계 2대, 로봇 3대, 컨베이어 벨트, AGV 등으로 구성된 모형 CIM 시스템을 제작하였다. 모형 CIM 시스템은 두 종류의 제품을 생산한다고 가정하고 누적형과 비누적형의 두 개의 생산라인을 가지게 설계하였다. 전체 시스템 구성은 그림 2와 같다.

3.2 Plant 모델링

전체 플랜트를 모델링한 automaton은 NC 기계, 로봇 등과 같은 각각의 시스템 구성요소들을 automaton으로 모델링한 후 각각의 automaton을 synchronous product하면 된다. 본 논문에서는 플랜트를 모델링할 때 상태와 이벤트를 최소화하였다. 즉 슈퍼바이저가 관측할 필요가 없거나 관측할 수 없는 이벤트와 legal language에 특별한 영향을 주지않는 이벤트들을 ε 이벤트로 projection시켰다. 예를들어 AGV를 모델링할 때 AGV의 속도가 변하는 것은 AGV automaton에 나타내지 않았다[6].

AGV 2대, 로봇 3대, NC 기계 2대, 컨베이어 벨트 5개, 센서 17개, 구속핀, 솔레노이드를 스테이트가 2개인 automaton으로 모델링하였다[7]. 그리고 전체 플랜트 automaton은 다음과같이 TCT[6]를 사용하여 구할 수 있다.

$$CIMPlant = SYNC(AGVs,\ Robots,\ NC\ Machines,\ Conveyor\ Belts,\ Sensors,\ Restraint\ Pin,\ Solenoid) \tag{9}$$

3.3 Supervisor 설계

슈퍼바이저는 legal language를 세워, 플랜트에 대한 supremal controllable sublanguage를 구한 결과이다. 이렇게 얻어진 슈퍼바이저는 다음을 만족시킨다.

$$L(S/G)=L(S) \tag{10}$$

그렇지만 이렇게 설계된 슈퍼바이저는 플랜트에서 일어나는 이벤트들에 대해서 legal language보다 훨씬 많은 스테이트를 갖고 있어 매우 복잡하게된다. 그래서 실제 구현시 어려움이 생기기 때문에 다음을 만족하는 또다른 슈퍼바이저를 구하는게 슈퍼바이저의 구현시 간편하다.

$$L(S'/G)=L(S) \tag{10}$$

즉, 슈퍼바이저 감독아래에서의 플랜트 거동 language는 S 감독아래에서의 플랜트 거동 language와 같지만 automaton은 S보다 간단한 S'을 설계할 수 있다. 이때 legal language를 K라고 하면, S는 최대 $K\wedge G$와 같은 스테이트 수를 가지지만, S'는 최대 K와 같은 스테이트 수를 가진다.

그리고 본 논문에서는 legal language를 $K_1, K_2, \ldots$로 설계하고 각각의 legal language를 만족시키는 슈퍼바이저를 $S_1, S_2, \ldots$으로 설계하였다. 이런 슈퍼바이저를 modular 슈퍼바이저라고 한다. 그리고 이렇게 설계된 슈퍼바이저가 optimal proper 슈퍼바이저인지는 다음의 조건을 만족하는지 검사함으로써 알수가 있다.

정리 1 : Optimal proper 슈퍼바이저[6]

1) 슈퍼바이저 S_i가 플랜트 G에 대해서 controllable해야 한다.
2) $\overline{L_m(S_i)}=L(S_i)$
3) SS_i를 K_i에 대한 supremal controllable sublanguage라 할 때, $L(S_i/G)=L(SS_i)$를 만족해야한다.

마지막으로 modular 슈퍼바이저가 플랜트에 대해서 centralized 슈퍼바이저와 같은 supervisory control을 하는지 검사하여야한다. 이는 S_i들이 서로 nonconflict한지 검사함으로써 알수가있다.

본 논문에서 설계한 플랜트 supervisory control testbed에는 아래와 같은 구속 조건을 주어 15개의 modular 슈퍼바이저를 만들어냈다.

1) 컨베이어 벨트 2, 4, 6의 버퍼 크기를 2로 한다.
2) 로봇 1이 작업물을 집어 NC 기계 1에 옮긴다음 NC 기계 1

이 가공을 완료하면 로봇 1이 작업물을 집어 컨베이어 벨트 위에 올려놓는다.

3) 로봇 2가 작업물을 집어 NC 기계 2에 옮긴다음 NC 기계 2가 가공을 완료하면 로봇 2가 작업물을 집어 컨베이어 벨트 위에 올려놓는다.
4) 두 생산라인에서 가공완료된 제품을 로봇 3이 두 대의 AGV에 구분하여 싣는다.
5) 솔레노이드가 작업물을 분류시킨다.
6) 두 대의 AGV는 각각의 위치에 제품을 하역한다.
7) AGV는 서로 충돌하지 말아야한다.

3.3.1 생산라인 슈퍼바이저

위의 구속조건 1)~5)에 맞는 legal language를 만들어 8개의 modular 슈퍼바이져를 만들었다[7]. 그림 3에 버퍼크기 슈퍼바이져 1을 나타내었다.

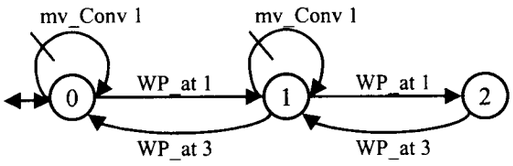

그림 3. 버퍼크기 슈퍼바이져 1
Fig 3. Buffer size Supervisoer 1

버퍼크기 슈퍼바이져의 제어 데이터는 초기 상태와 상태 1에서는 모든 이벤트를 enable하다가 상태 2에서 이벤트 mv_Conv1을 disable시키는 것이다. 즉 스트링 ε^*mv_Conv1* ε^*WP_at1 ε^*mv_Conv1* ε^*WP_at1 ε^*이 일어나면 버퍼크기 슈퍼바이져는 이벤트 mv_Conv1이 일어나지 못하게한다.

3.3.2 AGV 슈퍼바이저

구속조건 6),7)에 맞는 7개의 modular 슈퍼바이져를 설계하였다[7]. 구속조건 7)의 legal language는 AGV 라인을 그림 2와 같이 6개의 구간으로 나누어서 6개로 설계하였다. 그리고 각 구간에 대해서 슈퍼바이져를 만들었다. 그림 4에 구간 1에 대한 충돌방지 슈퍼바이져를 나타내었다. 다른 구간에 대한 충돌방지 슈퍼바이져는 상태 천이 이벤트만 각 구간의 센서 신호에 맞게 바꾸면 된다.

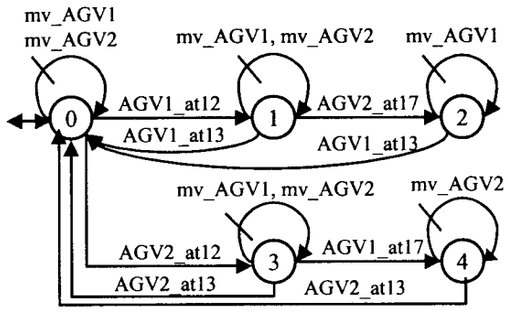

그림 4. 충돌방지 슈퍼바이져 1
Fig 4. Collision Protection Supervisor 1

충돌방지 슈퍼바이져 1은 상태 2에서 이벤트 mv_AGV 2를 disable시키고, 상태 4에서 mv_AGV 1을 disable시키는 제어 데이터를 가지고 있다. 즉, 스트링 ε^*(mv_AGV1 + mv_AGV2)* ε^*AGV1_at12 ε^*(mv_AGV1 + mv_AGV2)* ε^*AGV2_at17 ε^* mv_AGV1* ε^*이 발생하면 이벤트 mv_AGV 2를 disable시키고, 스트링 ε^*(mv_AGV1+mv_AGV2)* ε^*AGV2_at12 ε^*(mv_AGV1 +mv_AGV2)* ε^*AGV1_at17 ε^*mv_AGV1* ε^*이 발생하면 이벤트 mv_AGV1을 disable시킨다.

4. Implmentation

4.1 CMSSM 변환

본 논문에서는 설계된 modular 슈퍼바이져를 실제구현하기 위해서 CMSSM으로 변환하였다. CMSSM이란 현재의 상태, 입력 그리고 clock에 대해서 특정한 출력값을 가지는 machine을 말한다[4]. CMSSM으로 변환된 슈퍼바이져는 PLC나 디지털 회로로 쉽게 구현될 수 있다[1][2]. 그림 5.에 버퍼크기 슈퍼바이져 1의 CMSSM을 나타내었다.

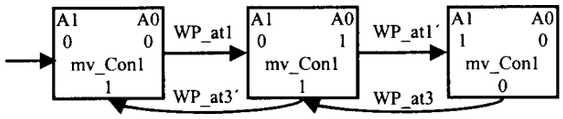

그림 5. 버퍼크기 슈퍼바이져 1의 CMSSM
Fig 5. CMSSM for Buffer size Supervisor 1

여기서 슈퍼바이져를 CMSSM으로 변환시킬 때 고려되어야할 중요한 점은 다음과 같다.

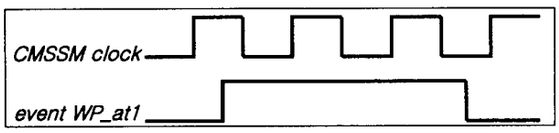

그림 6. 이벤트 발생 신호
Fig 6. Event occurrence signal

버퍼크기 슈퍼바이져의 초기 상태에서 이벤트 WP_at1이 발생하면 두번째 상태로 이동하게되고 또다시 WP_at1이 발생하면 세번째 상태로 바뀌게된다. 그런데 CMSSM에서는 그림 6과 같이 WP_at1의 발생시간이 CMSSM의 clock 주기보다 길게되면 CMSSM에서는 WP_at1이 여러번 발생한 것으로 인식하고 WP_at1이 한번 발생하더라도 초기상태에서 세 번째 상태로 이동하게 된다. 그러므로 슈퍼바이져를 CMSSM으로 변환시킬 때 같은 이벤트가 상태를 연속으로 바꿀때는 그런 이벤트들을 구분해야된다. 즉, 여기서는 초기 상태에서 일어나는 WP_at1와 첫번째 상태에서 일어나는 WP_at1을 구분해서 CMSSM을 만들어야된다. 버퍼크기 슈퍼바이져 CMSSM에서는 후자의 WP_at1을 WP_at1'으로 나타내었다. 그리고 이는 실제 구현시 또다른 센서가 추가되어야함을 의미한다.

이렇게 CMSSM이 만들어지면 CMSSM의 입력, 출력에 대해서 logic을 만들수가 있다. 버퍼크기 슈퍼바이져 CMSSM에 대해서는 센서 신호 WP_at1, WP_at1', WP_at3, WP_at3'가 입력이 되고, 컨베이어 벨트1의 제어 신호 mv_Conv1이 출력이 된다.

4.2 Simulation

앞에서 설계된 CMSSM이 원하는 제어신호를 생성하는지를 검증하기위해 회로설계 및 분석 프로그램인 PSpice로 CMSSM을 시뮬레이션하였다. 시뮬레이션은 각각의 CMSSM의 입력신호를 임의로 주고, 출력신호를 검사하였다. AGV 라인 구간 1에대한 충돌방지 슈퍼바이져의 CMSSM에 대한 PSpice 시뮬레이션 결과를 그림 7에 나타내었다.

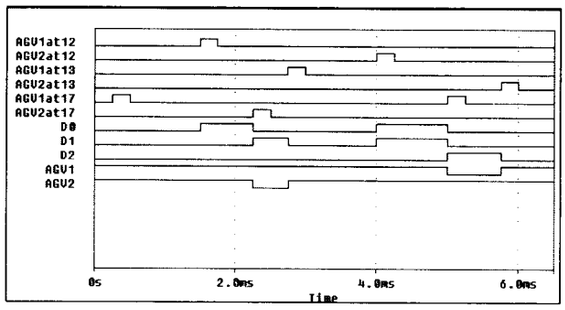

그림 7. 충돌방지 슈퍼바이져1의 시뮬레이션 결과

Fig 7. Simulation Result for Collision Protection Supervisor 1

그림 7에서 AGV1at12, AGV2at12, AGV1at13, AGV2at13, AGV1at17, AGV2at17이 입력으로 쓰인 센서신호이고, AGV1, AGV2가 출력을 나타내는 AGV 제어 신호이다. 그리고 D0 ~ D2가 CMSSM의 상태를 나타낸다. 시뮬레이션 결과를 분석해 보면 먼저 D0 ~ D2가 모두 0으로 초기상태로 되어있다. 이때 AGV1at17 이벤트가 발생하게 된다. 그렇지만 CMSSM의 상태는 바뀌는 않는다. 왜냐하면 AGV1at17은 초기상태에서 self-loop 이벤트이기 때문이다. 그리고 AGV1at12가 발생하면 CMSSM은 상태 1로 변화하고 이때 AGV2at17이 발생하게되면 AGV2를 disable시키게된다. 그리고 AGV1at13이 발생하면 다시 AGV1을 enable시킨다. 즉, 먼저 어떤 AGV가 구간 1로 들어오고 그 AGV가 그 구간을 빠져나가기전에 다른 AGV가 그 구간으로 들어온다면 나중의 AGV를 첫 번째 AGV가 그 구간을 빠져나가기전까지 정지시키게 되는것이다.

4.3 슈퍼바이저 제작

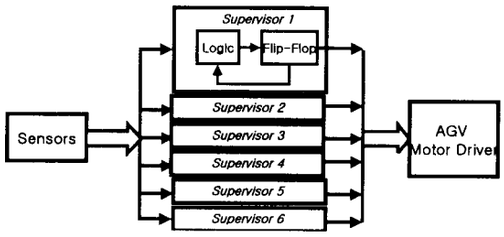

그림 8. AGV 충돌방지 시스템의 블록 다이아그램

Fig 8. Block Diagram of AGV Collision Protection System

그림 9는 실제 제작된 AGV 충돌방지 시스템이다.

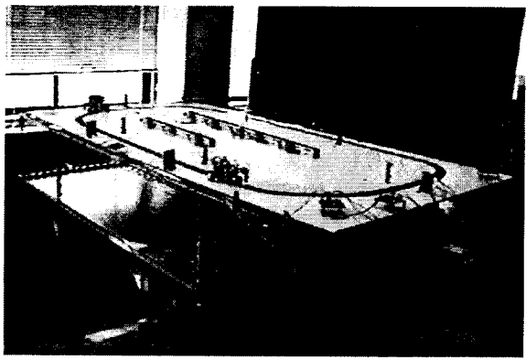

그림 9. AGV 충돌방지 시스템

Fig 9. AGV Collision Protection System

충돌 방지 슈퍼바이져를 그림 8과 같은 구조로 실제 제작하여 시험하였다. AGV 충돌방지 시스템은 다음과 같이 작동하게 된다. AGV 라인에 있는 각각의 센서들이 AGV 1, AGV 2를 감지하여 그 센서 신호를 6개의 충돌방지 modular 슈퍼바이져로 보내게된다. 그리고 각 슈퍼바이져들은 그 센서신호를 입력으로 받아들여 미리 입력된 논리회로에 따라 feedback 과정을 거쳐 AGV 모터 구동회로로 출력신호를 보내게 되는 것이다.

5. 결론

지금까지 supervisory control theory를 실제 시스템에 적용하기 위해 supervisory control testbed를 구성하고 modular 슈퍼바이져를 설계하였다. 슈퍼바이져를 modular로 설계함으로써 legal language가 바뀌거나 추가되어도 전체 슈퍼바이져를 바꾸지않아도 되므로 슈퍼바이져의 수정, 보완이 간편해진다. 또 controllability를 검사함에 있어서도 훨씬 적은 계산 복잡도을 요하게 된다. 그리고 이렇게 설계된 슈퍼바이져들을 CMSSM으로 변환하여 PSpice를 이용하여 시뮬레이션하였다. 시뮬레이션 결과 각각의 modular 슈퍼바이져들은 주어진 legal language대로 언어를 생성해내었다. 마지막으로 설계된 CMSSM을 실제 제작하여 AGV 시스템을 원활히 제어하였다.

그리고 본 논문에서는 partial observation 아래에서의 슈퍼바이져를 개발함으로써 많은 unobservable 이벤트들을 ε으로 고려하였다. 그렇지만 좀더 많은 이벤트를 고려하고 이를 hierarchical supervisor로 설계하면 좀더 실제시스템에 가깝게 시스템을 모델링할 수 있고, 더욱 정밀한 supervisory control을 할 수 있을 것이다. 또는 이벤트들의 발생 시간을 고려한 timed DES를 플랜트로 삼는 것도 보다 효과적인 감독 제어 시스템을 개발하는데 많은 도움을 줄것이다.

참고문헌

[1] B. A. Brandin "The Real-Time Supervisory Control of an Experimental Manufacturing Cell", Systems Control Group Report No. 9404, Department of Electrical and Computer Engineering University of Toronto, 1994

[2] R. J. Leduc, "PLC Implementation of a DES Supervisor for a Manufacturing Testbed", MACs. Thesis, Department of Electrical and Computer Engineering, University of Toronto, 1996

[3] P. J. Ramadge, W. M. Wonham, "The Control of Discrete Event Systems", *Proceedings of the IEEE*, Vol. 77, No. 1, January 1989, pp. 81-98

[4] J. Wakerley. *Digital Design Principles.* Prentice-Hall, Inc., 1990

[5] W. M. Wonham, P. J. Ramadge, "On the supremal controllable sublanguage of a given language", *SIAM J. Control and Optimization*, Vol. 25, No. 1, January 1987

[6] W. M. Wonham, "Notes on Control of Discrete-Event Systems", Department of Electrical and Computer Engineering, University of Toronto, 1998

[7] 손형일, 김철수, 이석, "제조셀의 제어를 위한 DES 슈퍼바이저의 설계", '99 춘계 정밀공학회 학술회의 논문집, pp. 721-724, 1999