이산사건 시스템 모델링과 구동 PLC 모델링을 결합한 통합 페트리네트

The Integrated Petri Net combining the discrete event system modeling with the operating PLC modeling

이 기 범*, 이 진 수**

*포항산업과학연구원, **포항공과대학교

1. 서론

이산 사건 시스템에서 가장 많이 쓰이고 있는 제어장치로는 PLC이다[3][21][22]. PLC 시스템의 초기 모델은 단순 릴레이반을 대처하기 위한 것이었다. 그러나 마이크로 프로세서 기술이 발전하면서 PLC 기능도 점차 고기능화 되어, 디지털 제어 뿐만 아니라 아날로그 제어도 가능하게 되었다. 한편 PLC의 신뢰도를 높이기 위하여 Redundancy 시스템[23][24]을 적용하고 있다. 복잡한 PID 제어가 가능할 뿐만 아니라 퍼지제어 기능을 갖은 고기능 PLC[17]도 출현하고 있다.

PLC 시스템의 제어언어로는 IL(Instruction List), LD (Ladder Diagram), FBD(Function Block Diagram), SFC(Sequential Function Chart), ST (Structured Text)가 사용되고 있다[1][4]. LD는 현장의 릴레이반을 대체시킨 기호로 출발하여 가장 오랜 동안 사용되고 있으며, 제어언어도 발전하여 IL 및 FBD로 사용 범위가 넓어졌다. 특히 Grafcet[2][15]을 기본으로 하여 발전한 SFC[9] 제어언어는 1995년에 이르러 IEC 1131-3 표준 규격[4]으로 최종 확정되었다.

플랜트를 제어하기 위하여 엔지니어는 플랜트의 순차 제어 방법을 정하고, 각 공정의 동시성과 비동기적 성질을 고려하여 LD 프로그램을 작성한다. 이러한 방법은 직관적이며 경험적인 방법으로서 많은 시행착오를 거쳐 최종 프로그램이 완성되게 된다. 한편 LD 제어언어는 현장의 작업 상황을 시퀀셜하게 작성한 심볼들의 집합으로서 순차적 표현으로는 적합하지만, 시스템 모델링 능력이나 분석 능력은 불가능하다. 따라서 보다 체계적으로 플랜트를 해석하고 분석하기 위하여 페트리네트 표현기법을 적용하고 있다[12][26]. 또한 LD 제어언어를 페트리네트로 도식화하여 모델링하고, 페트리네트의 분석 기법을 적용하여 보다 체계적으로 해석하고 있다[5][20].

페트리네트[14][18]는 이산사건 시스템의 표현에 적합한 도구로서 시각적 표현과 수학적 해석이 가능하여 현대 작업공정 및 생산 관리 시스템에 많이 적용되고 있다. 또한 제어분야로도 적용범위가 점차 확대되고 있으며, 퍼지 기법과 접목한 퍼지 페트리네트[17], 뉴럴네트와 퍼지함수를 접목한 뉴로퍼지 페트리네트[10][16]가 있다.

본 논문에서는 LD 제어언어를 페트리네트로 표현하는데 있어서 PLC 동작 특성을 고려한, 더미 플레이스와 보존아크를 새롭게 정의한다. 그리고 LD 제어언어의 페트리네트 변환 테이블를 만든다.

Modified 페트리네트를 사용하여 동시적, 비동기적 그리고 병렬적인 해석 기법을 제철소 원료 수송카 공정에 적용하여 페트리네트를 모델링한다. 그리고 모델링 된 제어대상 플랜트를 공정 흐름에 따라 해석하기 위한 상태방정식을 제시한다.

원료 수송카 작업 공정에 대한 페트리네트 모델링으로부터 상태표를 만들고, 그것으로부터 LD 제어언어를 만들어 낸다. 이렇게 만들어진 LD 제어언어를 페트리네트로 변환하기 위하여, 앞서 제시된 변환 테이블를 사용하여 페트리네트로 변환한다

페트리네트 모델링 적용 대상를 보면, 일반적인 공정이나 또는 시스템만을 각각 모델링하고 있다 그러나 본 논문에서는 대상 공정 뿐만 아니라 제어 시스템도 함께 모델링하여, 새로운 통합 페트리네트를 제시하고자 한다. 즉 원료 수송카를 제어하는 PLC 프로그램부(LD 제어언어)의 페트리네트와 원료 수송카의 작업 공정에 대한 페트리네트를 결합한 통합 페트리네트를 구현한다.

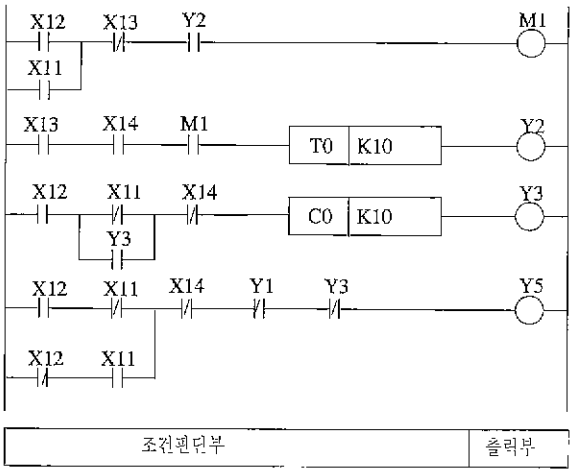


그림 1. 일반 LLD 제어언어 구성도

2. LD 제어언어와 페트리네트

2.1 LD 제어언어

래더 언어는 그림 1에서 보여주는 바와 같이 두 가지 스테이트로 이루어져 있다(조건 판단부, 출력부)[11]. 조건 판단부에서 각 접점의 스테이트를 읽어들여 참 혹은 거짓을 판단한 후 출력부에 있는 접점으로 On 또는 Off 값을 내보낸다. 조건 판단부에는 상시 On인 a 접점과 상시 Off인 b 접점으로 구성되어 있다. 출력부에는 직접 출력되어 외부 Actuator와 연결되는 출력 코일(Y_i)과 내부 메모리로 기록되는 내부 코일(M_j)로 구분할 수 있다. 그리고 입력 조건이 On이면 지정된 시간만큼 시간 지연한 후에 출력이 발생되는 타임머가 있고, 입력 조건이 지정된 횟수 만큼 On, Off를 반복하였는가를 적산하여 출력이 발생되는 카운터가 있다[9].

2.2 페트리네트 구조

페트리네트는 Discrete event 시스템에 적용할 수 있는 시각적으로 표현되는 그레프형의 모델링 기법으로서 수학적 분석이 가능한 모델링 도구이다[19] 페트리네트는 동시적(concurrent), 비동기적(asyn chronous), 분산적(distributed), 병렬적(parallel), 비확정적(nondeterministic), 그리고 확률적(stochastic)인 현상을 갖는 정보처리, 작업공정 시스템을 묘사하고 분석, 연구하는데 있어 유익한 도구이다[18]. 특히 페트리네트는 Discrete event 조건의 모델링 능력이 우수하여 생산공정, 작업공정, 그리고 통신망 모델링[14]에 많이 적용되고 있다.

일반적인 페트리네트는 플레이스와 트랜지션으로 구성되어 있으며 그림 3과 같이 표현된다 플레이스는 원으로 된 노드(Node)이며 토큰을 가질 수 있다. 트랜지션은 수직으로 된 직선으로서 발화 조건에

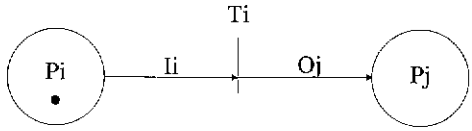


그림 2. 타임드 페트리네트 구조

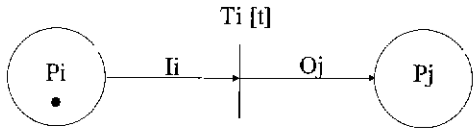


그림 3. 기본 페트리네트 구조

따라서 토큰의 이동를 제어한다. 그리고 플레이스와 트랜지션을 연결하는 직선으로 된 아크(Arc)는 토큰의 흐름을 화살표로 표시하고 있다[2].

일반적인 페트리네트는 다음과 같이 5개의 구성원소로 이루어져 있다.

$$PN = P, T, I, O, M0 \quad (1)$$

P : 플레이스의 집합
T : 트랜지션의 집합
I : $P \rightarrow T$, 입력함수
O : $T \rightarrow P$, 출력함수
M_0 : 페트리네트의 초기 마킹

페트리네트의 발화 조건은 플레이스 후단의 트랜지션와 연결되어 있는 아크 수와 플레이스의 토큰 수를 비교하여 같거나 클 때 발생한다. 이러한 발화 조건이 성립하면 플레이스(Pi)에 있는 토큰은 플레이스(Pj)로 즉시 이동하게 된다.

발화 조건이 성립되어도 토큰의 흐름에 지연 시간을 주기 위하여, 그림 2와 같이 트랜지션에 지연 시간 값을 갖는 Timed Petri net가 있다[2][10].

복잡한 페트리네트를 간결하게 표현하기 위해, 각각의 토큰에 특성를 지정하여 플레이스 수를 줄이고, 트랜지션의 발화조건에 함수를 추가하여 토큰을 수행시키는 Coloured Petri net가 있다[6][7].

3. Modified 페트리네트와 LD 제어언어

3.1 Modified 페트리네트

그림 1에서 보여주는 LD 제어언어의 동작 상황을 보면 입력 접점이 On됨과 동시에 출력 접점이 On되도록 되어 있다. 그리고 입력 접점이 Off되면 출력 접점도 Off 된다. 한편 입력접점의 상태는 외부에서 들어오는 신호로서 출력 접점이 On이 되어도 Off되지 않는다.

이것은 LD를 페트리네트로 대응시킬 때 매우 중요한 사항으로서, 일반 페트리네트를 그대로 LD에 적용시키면 안되는 요인이다. 따라서 일반 페트리네트에 다음과 같은 규칙을 추가 적용한다.

규칙 1: 파이어링 규칙

플레이스(Pi)에 있는 토큰이 트랜지션(Ti)와 연결된 아크수(Ai)보다 같거나 많으면 파이어링 조건이 된다. 이때 플레이스(Pi)에 있는 토큰은 변화가 없으며, 다음 플레이스(Pj)에 아크수(Aj)와 같은 토큰의 개수가 생성된다.

규칙 2: 플레이스 규칙

페트리네트가 가지는 상태 플레이스와 형태는 같으나, 기능적으로 보면 플레이스내의 토큰의 변화는 연결된 아크와 트랜지션의 파이어링 조건에 의하여, 토큰이 다음 플레이스로 이동할 것인가 또는 유지하면서 전달만 할 것인가 결정한다.

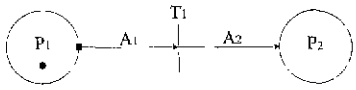
이와 같은 규칙에 의거하여 수행되는 페트리네트를 만들기 위해, 다음과 같은 플레이스와 아크를 정의한다.

정의 1: 보존아크

상태 플레이스에 토큰이 생성되어 파이어링 조건이 만족되면, 생성된 토큰은 일반 아크(⟶)에 의하여 다음 플레이스로 이동한다. 그러나 파이어링 조건이 만족되었을 때 보존아크 (●⟶)를 만나면, 입력 플레이스의 토큰은 이동되지 않는다. 다만 출력 플레이스에 아크수에 해당하는 수만큼 토큰이 전달되게 된다.

그림 4의 예에서 보면, 플레이스(P_1)에 한개의 토큰이 있으므로 트랜지션(T_1)의 파이어링 조건는 만족된다. 따라서 플레이스(P_1)의 토큰는 플레이스(P_2)로 이동이 가능하다. 그러나 플레이스(P_1)와 연결된 아크(A_1)가 보존아크이므로 플래이스(P_1)의 토큰이 제거되지 않는 상태에서 플레이스(P_2)에 토큰이 전달된다.

a1. 트랜지션의 발화 전



a2. 트랜지션의 발화 후

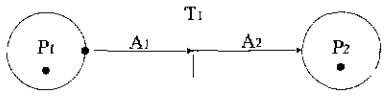


그림 4. 보존 아크의 토큰 전이 과정

정의 2: 더미 플레이스

상태 플레이스는 토큰을 보유할 수 있는 원으로서 일반적으로는 트랜지션의 파이어링으로 플레이스에 토큰이 전달되도록 되어 있다. 그러나 이와 같은 상태 플레이스만으로는 LD 언어를 표현하는데 한계가 있다. 따라서 다음과 같은 더미 플래이스를 정의한다.

그림 5. 더미 플레이스

더미 플레이스 :

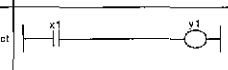
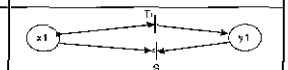
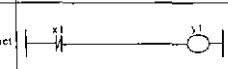
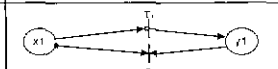
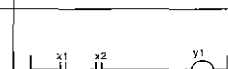
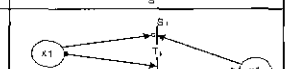
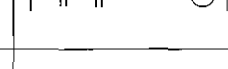
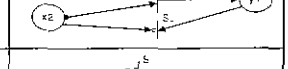
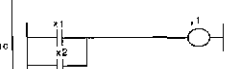
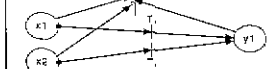
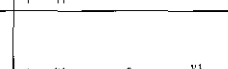
- 일반 상태 플레이스의 크기에서 내부에 D의 기호를 갖는 그림 5와 같은 심볼로 표현한다.
- 더미 플레이스의 기능은 상태 플레이스가 가지는 기능과 같다. 다만, 모델링 과정에서 시스템의 디바이스와 상관없이 추가되는 플레이스이다.

예를 들면, LD 구조에서 OR 블럭이 AND 조건과 연결될 경우 더미 플레이스를 사용하지 않으면, 순차적 논리 수식에 부적합하게 된다. 따라서, 표 1에서 보는 바와 같이 LD 모델링시 논리적 해석과 일치하도록 더미 플레이스를 사용한다.

3.2 LD 제어언어의 페트리네트 대응 변환

앞에서 정의한 더미 플레이스와 보존아크를 사용하여 LD 제어언어를 페트리네트로 변환해 보면 표 1과 같다.

표 1. LD 제어언어의 페트리네트로의 변환

Function	Ladder Diagram	Petri net
a contact		
b contact		
AND logic		
OR logic		
OR-AND logic		
Timer		

a 접점을 페트리네트로 표현하기 위하여 보존아크와 금지아크를 사용한다. 즉 플레이스(X_1)에 토큰이 입력되면 플레이스(Y_1)에 토큰이 생성된다. 반면에 플레이스(X_1)에 토큰이 제거되면 금지아크와 연결되어있는 흡수 트랜지션(S)이 발화되어 플레이스(Y_1)의 토큰은 제거된다.

b 접점에서는 플레이스(X_1)에 토큰이 없으면 플레이스(Y_1)에 금지아크에 의하여 토큰이 생성된다. 반면에 플레이스(X_1)에 토큰이 있으면 흡수 트랜지션(S)이 발화되어 플레이스(Y_1)로부터 토큰을 제거한다.

OR-AND 결합 로직에서는 더미 플레이스를 사용하여 페트리네트 로직를 표현한다. X_1 또는 X_2 접점이 On이고 X_3 접점이 On이면 Y_1이 출력된다. 이와 같은 표현을 페트리네트로 표현하려면 더미 플레이스(D)를 사용하여 표현하여야 한다. 반면에 플레이스(X_1)와 플레이스(X_2)의 토큰이 둘다 제거되면, 흡수 트랜지션(S_1)이 발화되어 플레이스(Y_1)의 토큰이 제거되고, 플레이스(X_3)의 토큰이 제거되면 흡수 트랜지션(S_2)이 발화되어 플레이스(Y_1)의 토큰이 역시 제거된다.

Timer에서는 타임드 페트리네트를 사용하여 페트리네트로 변환한다. 플레이스(X_1)와 플레이스(X_2)에 토큰이 동시에 입력되면 타임드 트랜지션(Ti[10])이 발화된다. 즉 10msec 지연 후에 플레이스(Y_1)에 토큰이 생성된다. 반면에 플레이스(X_1) 또는 플레이스(X_2)에 토큰이 제거되면 플레이스(Y_1)에 있는 토큰는 자동으로 제거된다.

4. 대상 플랜트의 페트리네트 모델링

대상 플랜트는 원료의 수송작업을 반복하는 원료 수송카로서 구성도는 그림 2와 같다. 여기서 원료 수송카는 정해진 철로 위를 반복적으로 이동하면서 원료를 수송하고 있다.

4.1 플랜트 구동 사항

원료 수송카를 포함한 전 공정는 PLC 시스템에 의하여 자동으로 구동되고 있으며 전체 진행 공정은 다음과 같다.

1) 초기상태로서 원료 수송카가 S_2 지점에 있고, 원료 탱크에 원료의 높이가 L_1이하에 있고, V_2 벨브가 닫혀있고, 그리고 K_1 브레이크가 잡혀있으며, V_1 벨브가 열리면서 원료가 원료탱크에 주입되게 된다.
2) 일정시간이 지나 원료의 높이가 L_2 높이가 되면 V_1 벨브는 닫힌다. 그리고 원료 수송카의 전진 스위치를 On하여 S_1 지점으로 이동하게 한다.
3) 일정시간이 지나 원료 수송카가 S_1 지점에 도착하면 K_2 브레이크가 작동하여 원료 수송카는 멈추게 된다.
4) 원료 수송카가 멈춘 뒤 V_2 벨브를 열어 원료 저장탱크에 원료를 방출하게 된다. 일정시간이 지난 뒤 원료가 줄어들어 L_1 레벨 이하가 되면 V_2 벨브를 닫는다.
5) 원료 수송카의 후진 스위치를 On하여 S_2 지점으로 이동하게 한다.
6) 일정시간이 지나 원료 수송카가 S_2 지점에 도착하면 K_1 브레이크가 작동하여 원료 수송카는 멈추게 된다
7) 1)-6)의 과정를 반복 작업하게 된다.

4.2 제어 대상에 대한 상태 플레이스 선정

원료 수송카의 반복된 운전 상황를 페트리네트의 상태 플레이스로 표현한다.

P_1 : 원료 수송카가 S_2지점에 도착하여 정지한 상태

P_2 : 원료를 수송 탱크에 싣고 있는 상태

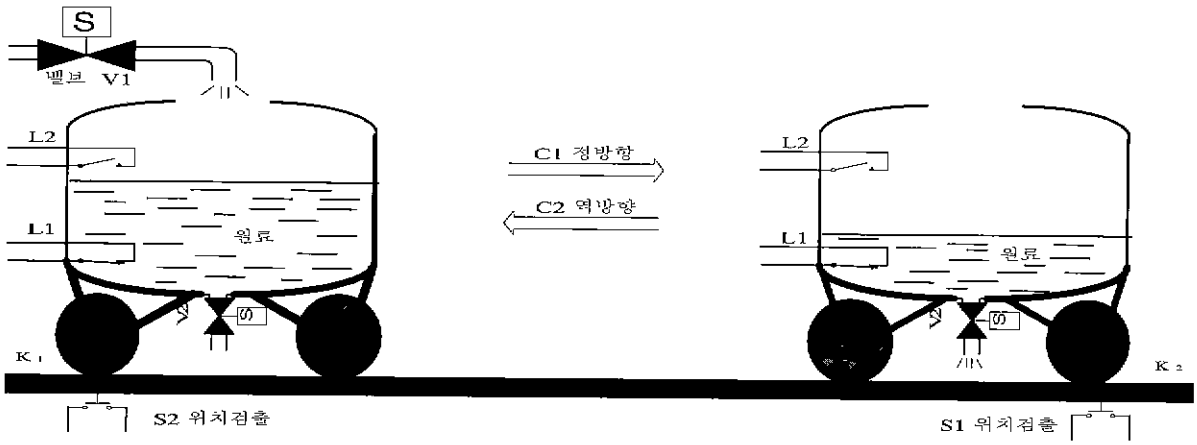


그림 6. 원료 수송카 작업 공정

P_3 : 원료 수송카가 S_2지점에서 S_1지점으로 이동 상태
P_4 : 원료 수송카가 S_1지점에 도착하여 정지한 상태
P_5 : 원료를 수송 탱크에서 저장 탱크로 방출하고 있는 상태
P_6 : 원료 수송카가 S_1지점에서 S_2지점으로 이동상태

4.3 제어 장치에 대한 상태 플레이스 선정

제어 장치는 원료 수송카가 움직이는데 필요한 조작 기기들로서 위치 센서 신호(S_{1i},S_{2i}), 레벨 센서 신호(L_{1i},L_{2i}) 그리고 PLC에 의해 출력되는 출력신호(Y_1 - Y_6)가 있다. 페트리네트에 의한 각 디바이스의 표시상태는 다음과 같다.

0 원료 수송카 표시상태
C_{10} : 원료 수송카의 전진 스위치가 Off 상태
C_{11} : 원료 수송카의 전진 스위치가 On 상태
C_{20} : 원료 수송카의 후진 스위치가 Off 상태
C_{21} : 원료 수송카의 후진 스위치가 On 상태

0 벨브 표시상태
V_{10} : 벨브 V_1이 Off 상태
V_{11} : 벨브 V_1이 On 상태
V_{20} : 벨브 V_2가 Off 상태
V_{21} : 벨브 V_2가 On 상태

0 원료탱크의 원료 높이 검출센서 표시상태
L_{10} : 원료탱크의 높이 검출센서 L_1이 Off 상태 (b 접점)
L_{11} : 원료탱크의 높이 검출센서 L_1이 On 상태 (b 접점)
L_{20} : 원료탱크의 높이 검출센서 L_2가 Off 상태 (a 접점)
L_{21} : 원료탱크의 높이 검출센서 L_2가 On 상태 (a 접점)

0 원료 수송카의 위치검출 표시상태
S_{10} : 원료 수송카의 위치검출 센서 S_1이 Off 상태
S_{11} : 원료 수송카의 위치검출 센서 S_1이 On 상태
S_{20} : 원료 수송카의 위치검출 센서 S_2가 Off 상태
S_{21} : 원료 수송카의 위치검출 센서 S_2가 On 상태

0 위치센서 검출 후 브레이크 지시 상태
K_{10} : 원료 수송카를 멈추기 위한 S_1지점의 브레이크 Off 상태
K_{11} : 원료 수송카를 멈추기 위한 S_1지점의 브레이크 On 상태
K_{20} : 원료 수송카를 멈추기 위한 S_2지점의 브레이크 Off 상태
K_{21} : 원료 수송카를 멈추기 위한 S_2지점의 브레이크 On 상태

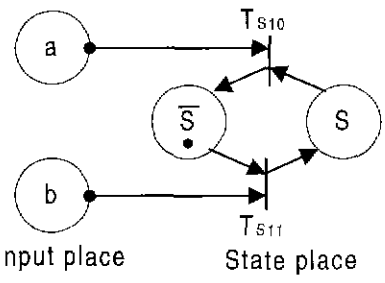


그림 7. 토큰 입력에 따른 On, Off 상태 페트리네트

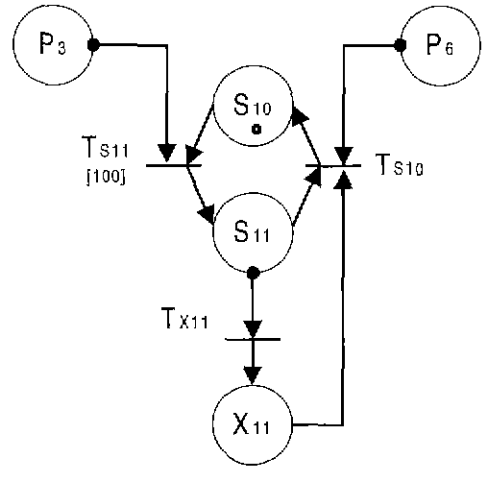


그림 8. 입력단자 플레이스의 On, Off 상태 표시

4.4 대상 플랜트의 페트리네트 모델링

4.4.1 On, Off 상태 플레이스

입력 디바이스와 출력 디바이스의 상태를 나타내기 위하여 그림 7과 같은 페트리네트 그래프로 표현한다.

플레이스(a)에 토큰이 입력되면 트랜지션(T_{S10})이 발화되어 플레이스(S)로 토큰이 이동된다. 그러나 그림 9에서 보여준 바와 같이 보존아크에 의하여 플레이스(a)의 토큰은 제거되지 않고 그대로 유지된다. 플레이스(S)로 토큰이 이동된 후 플레이스(b)에 토큰이 입력되면 같은 방법으로 트랜지션(T_{S11})이 발화되어 플레이스(S)의 토큰은 플레이스(S)로 이동된다. 따라서 플레이스(a)는 플레이스(S)의 Off 상태를 표현하여 주고, 플레이스(b)는 플레이스(S)의 On 상태를 표현하여 준다.

4.4.2 입력단자 플레이스 토큰 On, Off

입력단자 플레이스 토큰의 입.출력를 구현하기 위하여 그림 8과 같은 페트리네트 그래프를 구성할 수 있다.

입력단자 플레이스(X_{11})에 토큰을 입력하기 위하

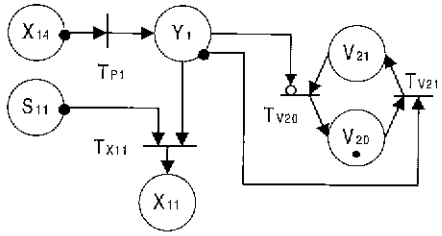


그림 9. 출력단자 플레이스의 On, Off 상태 표시

여 플레이스(P_3)에 토큰이 주입되면 된다. 한편 입력단자 플레이스(X_{11})에서 토큰을 제거하기 위하여 플레이스(P_6)에 토큰이 입력되면 된다.

4.4.3 출력단자 플레이스 토큰 On, Off

출력단자 플레이스 토큰의 입.출력를 구현하기 위하여 그림 9와 같은 페트리네트 그래프를 구성할 수 있다.

출력단자 플레이스(Y_1)에 토큰를 생성하기 위하여 입력단자 플레이스(X_{14})에 토큰이 입력되면 된다. 한편 출력단자 플레이스(Y_1)의 토큰를 제거하기 위하여 플레이스(S_{11})에 토큰이 주입되면 된다. 이때 플레이스(Y_1)의 토큰은 플레이스(X_{11})로 빠져 나간다. 플레이스(V_{20})와 플레이스(V_{21})는 출력단자 플레이스의 토큰 유,무를 표시해주는 출력 디바이스 플레이스이다.

4.4.4 원료 수송카의 페트리네트 모델링

원료 수송카의 전체 공정에 따른 대상 플랜트를 페트리네트로 모델링하기 위해 전체 시스템을 크게 4부분으로 나눈다.

- 0 제어대상부: 원료 수송카를 중심으로 이루어지는 기본 공정
- 0 입력 디바이스부 : 전체 공정의 신호 검출부
- 0 출력 디바이스부 : 전체 공정의 신호 출력부
- 0 PLC 프로그램부 : 입력 디바이스로부터 신호을 받아 해당 연산을 수행한 후 출력 디바이스로 신호를 출력하는 부분

이와 같은 구조에서 각각의 순차 공정에 의거하여 부분별 페트리네트로 구성하고, 상호 연관 관계를 고려하여 전체 페트리네트 모델링을 그림 10과

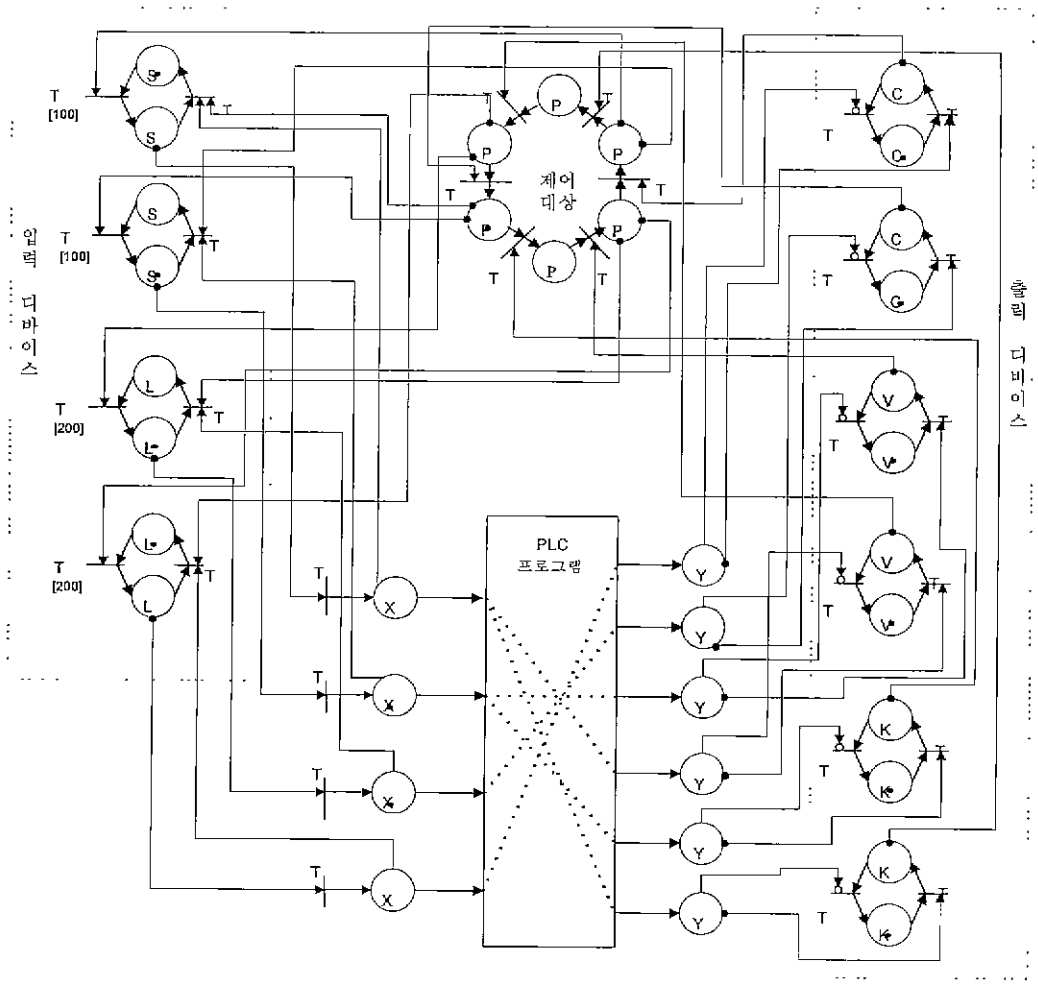


그림 10. 원료 수송카 작업 플랜트에 대한 페트리네트 그래프

같이 구할 수 있다. 여기서 원료 수송카의 기본 순환 구조를 제어 대상으로 표현하였고, 레벨 검출기와 위치 검출기는 입력 디바이스 상태로 표현하였다. 그리고 원료 수송카 방향 스위치, 벨브 구동 스위치. 그리고 브레이크 스위치는 출력 디바이스 상태로 표현하였다.

원료 수송카 공정의 모델링 순서는 다음과 같다.

1) 원료 수송카의 기본 공정를 중심으로 선정된 플레이스(P_1 - P_6)까지의 진행 순서를 순차적으로 나열하고 각 공정간의 진행을 트랜지션과 아크로 연결한다.
2) 입력 디바이스 상태와 제어 대상 공정간의 상관관계를 순차적으로 고려하며 트랜지션과 아크로 연결한다.
3) 출력 디바이스 상태와 제어 대상 공정간의 상관관계를 트랜지션의 발화 조건과 연관지어 순차적으로 트랜지션 및 아크를 연결하여 나간다.
4) 입력 디바이스 상태 플레이스는 PLC의 입력 단자 플레이스와 연관지어 연결하고 출력 디바이스 플레이스의 트랜지션과는 PLC의 출력 단자 플레이스와 연관지어 연결한다.
5) 원료 수송카의 초기 조건를 선정하고 해당 플레이스에 초기화 토큰를 부여한다. 초기상태 : 원료 수송카가 S1지점에서 원료를 방출하고 나서 V_1 벨브를 잠근 뒤에 S_2 지점으로 접근하여 위치 검출 센서가 On된 상태를 초기 상태로 선정한다.

$$L_{11} = \mathrm{On}\ ,\ X_{13} = \mathrm{On}$$
$$S_2 = \mathrm{On}\ ;\ X_{12} = \mathrm{On}$$

6) PLC 내부 프로그램을 고려하여 전체 공정에 대한 순차 흐름도에 따라서 토큰을 이동시키면서 모델링된 페트리네트 그래프가 정확한가를 검증한다.

여기서 플레이스(S_{10},S_{20})에 대한 타임드 트랜지션은 100초 동안 이동 중임을 나타내고 있으며, 플레이스(L_{10},L_{20})에 대한 타임드 트랜지션는 200초 동안 원료를 원료 수송 탱크에 입력 또는 방출하고 있음을 나타낸다.

5. 제어대상부의 토큰 해석

5.1 제어대상부의 상태 방정식

페트리네트 모델링에서 원료 수송카의 제어대상 공정을 고려하여 정리하면 그림 11과 같이 표현할 수 있다.

여기서 초기상태는 원료 수송카가 원료를 싣기 위하여 플레이스(P_1)에 정지하고 있는 상태이다. 원

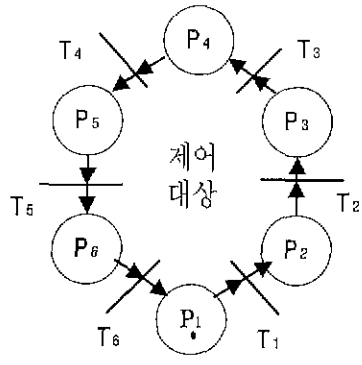


그림 11. 제어 대상부의 페트리네트

료 수송카의 현재 공정상태를 토큰으로 표현하고 있으므로 실제 작업사항과 토큰의 이동 사항을 수학적으로 해석하기 위하여 상태방정식을 도입한다.

시간 kT 일 때 시스템 상태는 상태벡터 P(k)로 표시된다. 이때 발화에 대한 토큰 수 증감은 $J \bullet T(k)$로 표시된다. 그래서 시간 (k+1)T에서 시스템 상태 방정식 P(k+1)는 (2)식과 같이 표현된다[2].

$$P(k+1) = P(k) + J \bullet T(k) \qquad (2)$$
여기서 : $J = O - I$
$T(k)$ = 트랜지션 벡터

그림 11로부터 접속행렬(J)을 구하면 다음과 같다.

입력행렬 :

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

출력행렬 :

$$O = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

접속행렬 :

$$J = O - I = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \qquad (3)$$

따라서 상태 방정식은 다음과 같이 구할 수 있다

$$\begin{bmatrix} P_1(k+1) \\ P_2(k+1) \\ P_3(k+1) \\ P_4(k+1) \\ P_5(k+1) \\ P_6(k+1) \end{bmatrix} = \begin{bmatrix} P_1(k) \\ P_2(k) \\ P_3(k) \\ P_4(k) \\ P_5(k) \\ P_6(k) \end{bmatrix} +$$

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} T_1(k) \\ T_2(k) \\ T_3(k) \\ T_4(k) \\ T_5(k) \\ T_6(k) \end{bmatrix} \quad (4)$$

5.2 점화 횟수 구하는 방법

토큰의 초기상태에서 원하는 위치까지의 트랜지션 점화 횟수를 구하기 위하여 (2)식으로부터 상태 방정식을 정리하면 다음과 같다.

$$J \bullet T(k) = P(k+1) - P(k)$$

$$T(k) = J^{-1}\{P(k+1) - P(k)\} \quad (5)$$

(5)식으로부터 트랜지션 점화 벡터계열을 구할 수 있다. 상태가 $P_0, P_1, P_2, P_3 \Lambda$ 로 전이하는 경우 점화 벡터계열은

$$T(0) = J^{-1}\{P(1) - P(0)\}$$
$$T(2) = J^{-1}\{P(2) - P(1)\} \quad (6)$$
$$---$$

으로 된다. 따라서 초기상태 *P(0)*부터 최종상태 *P(N)*까지 발화 벡터계열은 다음과 같이 구할 수 있다[19].

$$\sum_{k=0}^{N-1} T(k) = \sum_{k=1}^{N} J^{-1}\{P(k) - P(k-1)$$
$$= J^{-1}\{P(N) - P(0)\} \quad (7)$$

6. LD 제어언어의 페트리네트 변환

원료카의 작업 공정 모델링으로부터 PLC의 LD 제어 언어를 구하기 위하여 공정 상태표를 구한다. 공정 상태표로부터 상태 논리식을 구한다. 그리고 상태 논리식으로부터 LD 제어 언어를 구하고 그것를 근거로 하여 PLC 페트리네트 모델링을 구한다.

6.1 LD 제어언어 표현

원료 수송카의 전체 공정을 페트리네트로 표현한 그래프가 그림 10이다. 따라서 그림 10에 나타나 있는 원료 수송카의 순차 공정을 기본으로 하여 입력변수와 출력변수를 선정한다. 원료 수송카의 현재 시간 kT에서의 작업공정을 기준으로 하고, 다음시간 (k+1)T으로의 작업 공정을 순차적으로 나열하여 상태표로 정리하면 표 2와 같다. 여기서 X는 don't care 상태이다.

표 2. 원료 수송카의 작업 공정에 따른 상태표

상태	현재 시간 kT										다음 시간 (k+1)T						비 고
	위치 레벨				수송카 벨브 브레익						수송카 벨브 브레익						
	X_{11}	X_{12}	X_{13}	X_{14}	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	
	S_1	S_2	L_1	L_2	C_1	C_2	V_1	V_2	K_1	K_2	C_1	C_2	V_1	V_2	K_1	K_2	
P_1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0	수송카 정지 S_2
P_1	0	1	1	0	0	0	0	0	1	0	X	X	1	0	1	X	벨브 연다
P_2	0	1	0	0	X	X	1	0	1	X	X	X	1	0	1	X	원료 주입한다
P_2	0	1	X	1	X	X	1	0	1	X	0	0	0	0	X	X	벨브 닫는다
P_3	0	1	X	1	0	0	0	0	X	X	1	0	0	0	0	0	이동 지시
P_3	0	X	X	1	1	0	0	0	0	0	1	0	0	0	0	0	수송카 이동
P_4	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	수송카 정지 S_1
P_4	1	0	0	1	0	0	0	0	0	1	X	X	0	1	X	1	벨브 연다
P_5	1	0	0	0	X	X	0	1	X	1	X	X	0	1	X	1	원료 방출한다
P_5	1	0	1	X	X	X	0	1	X	1	0	0	0	0	X	X	벨브 닫는다
P_6	1	0	1	X	0	0	0	0	X	X	0	1	0	0	0	0	이동 지시
P_6	X	0	1	X	0	1	0	0	0	0	0	1	0	0	0	0	수송카 이동

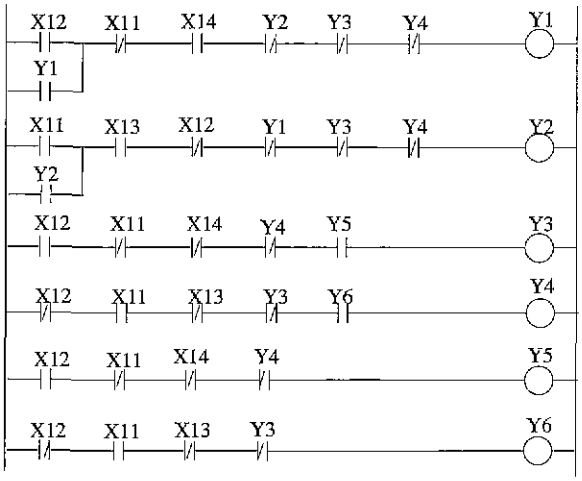


그림 12. 원료 수송카 제어를 위한 LD 제어언어

6.1.1 상태 논리식

원료 수송카의 작업 공정을 수행하기 위하여 출력제어 값이 '1'이 되어야 한다. 따라서 표 2에서 출력변수의 값이 '1'이 되는 출력 값을 기준으로 상태 논리식을 정리하면 다음과 같다.

$$
\begin{aligned}
Y_1 &= X_{12}\overline{X}_{11}X_{14}\overline{Y}_1\overline{Y}_2\overline{Y}_3\overline{Y}_4 + \overline{X}_{11}X_{14}Y_1\overline{Y}_2\overline{Y}_3\overline{Y}_4 \\
&= (X_{12}+1)\overline{X}_{11}X_{14}\overline{Y}_2\overline{Y}_3\overline{Y}_4 \\
Y_2 &= X_{11}X_{13}\overline{X}_{12}\overline{Y}_1\overline{Y}_2\overline{Y}_3\overline{Y}_4 + X_{13}\overline{X}_{12}\overline{Y}_1Y_2\overline{Y}_3\overline{Y}_4 \\
&= (X_{11}+1)X_{13}\overline{X}_{12}\overline{Y}_1\overline{Y}_3\overline{Y}_4 \qquad (8) \\
Y_3 &= X_{12}\overline{X}_{14}\overline{X}_{11}\overline{Y}_4Y_5 \\
Y_4 &= \overline{X}_{13}\overline{X}_{12}X_{11}\overline{Y}_3Y_6 \\
Y_5 &= \overline{X}_{11}X_{12}\overline{X}_{14}\overline{Y}_4 \\
Y_6 &= X_{11}\overline{X}_{12}\overline{X}_{13}\overline{Y}_3
\end{aligned}
$$

6.1.2 LD 제어언어

그림 10으로부터 상태표를 구하고 그 상태표로부터 상태 논리식를 구하였다. 따라서 상태 논리식으로부터 LD 제어언어를 구할 수 있다. 이때 논리식의 곱 연산은 LD 제어언어의 AND 로직으로 연결하고, 더하기 연산은 LD 제어언어의 OR 로직으로 연결한다. AND, OR 로직 연결은 조건 판단부에 위치시키며, 출력 로직(Y_1 - Y_6)은 LD의 출력부에 위치시킨다. 이와 같은 방법으로 구한 LD 제어언어는 그림 12와 같다.

6.2 LD 제어언어의 페트리네트 표현

표 1에서 정의한 LD 제어언어의 각 로직에 대한 대응관계를 그림 12에 적용하므로서, PLC 프로그램부의 LD 제어언어에 관한 페트리네트 그래프를 그림 13과 같이 구할 수 있다.

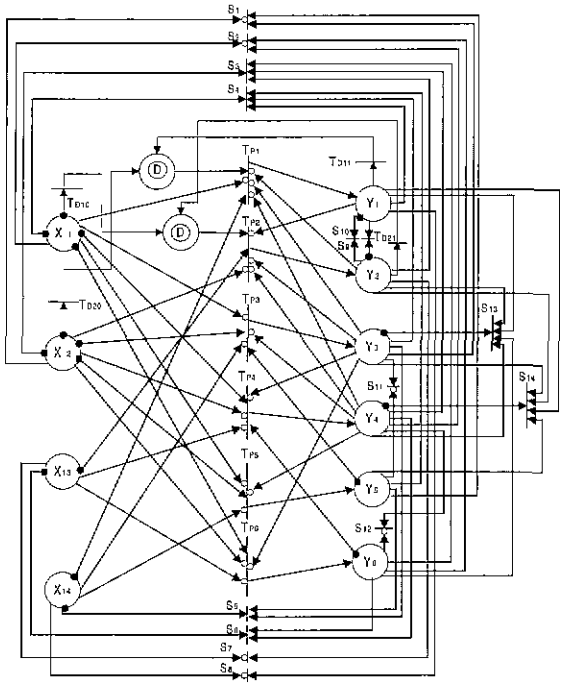


그림 13. LD 제어언어의 페트리네트 표현

PLC 입력단자에 해당하는 입력 플레이스(X_{11}, X_{12}, X_{13}, X_{14})는 좌측에 배치하고, 출력단자에 해당하는 출력 플레이스(Y_1 - Y_6)는 우측에 배치한다. 그리고 LD 제어언어를 기준으로 AND 또는 OR 로직 연결 관계에 따라 페트리네트의 아크를 연결하여 나간다. 한편 Y_1과 Y_2 출력의 경우, OR와 AND 로직이 직렬로 연결되어 있으므로 더미 플레이스를 사용하여 연결하여야 한다. 각 접점의 토큰 제거 방법은 흡수 트랜지션(S_1 - S_{14})을 사용하여 연결한다.

7. 통합 페트리네트 구현

그림 10에서 구한 원료 수송카의 작업 공정 페트리네트와 그림 13에서 구한 LD 제어언어의 페트리네트를 결합한 통합 페트리네트를 그림 14와 같이 구할 수 있다. 두 개의 페트리네트를 결합할 때 입력 플레이스(X_{11}, X_{12}, X_{13}, X_{14})는 입력 플레이스와 결합시키고, 출력 플레이스((Y_1 - Y_6))는 출력 플레이스와 결합시킨다. 여기서 PLC 프로그램부의 페트리네트는 토큰의 흐름을 분석하는데 있어 보다 명확하게 보여주기 위하여, 그림 13의 페트리네트 그래프를 간략하게 표현하여 결합하였다.

7.1 통합 페트리네트 분석

원료 수송카의 작업 공정에 대한 페트리네트에 PLC 프로그램부의 페트리네트가 추가 결합됨으로서 전체 작업 공정에 대한 토큰의 흐름 분석이 가능하

게 되었다.

PLC 프로그램부는 PLC 제어부분를 담당하는 페트리네트로 구성되며, 입력 디바이스 플레이스에서 토큰을 받아 PLC 프로그램 연산을 수행한 뒤 출력 디바이스 플레이스로 전달됨을 알 수 있다. 이와 같이 전달된 토큰은 제어 대상 플레이스의 실제적인 공정과 일치하여 순차적으로 진행될 수 있도록 해당 트랜지션을 발화시키게 된다.

제어대상 공정이 순차 진행됨에 따라 해당 공정에서의 토큰은 입력 디바이스 플레이스의 상태를 On, Off로 진행될 수 있도록 해당 트랜지션을 발화시킨다. 결과적으로 입력 디바이스 플레이스의 토큰은 PLC 입력단자 플레이스로 이동됨으로서, PLC 프로그램부에서는 입력단자 플레이스에 입력된 토큰의 발화 조건에 따라 순차적으로 출력단자 플레이스로 토큰을 생성시키게 된다.

토큰의 흐름은 원료 수송카의 작업공정 뿐만 아니라 원료 수송카를 제어하는 PLC 프로그램까지도 표현하여 주므로서, 외부적인 토큰의 생성 또는 소멸됨이 없이 단일 페트리네트 구조상에서 공정 분석이 가능하게 되었다.

7.2 토큰의 전이 과정

초기상태 토큰으로부터 토큰의 이동이 가능한 순서에 따라 진행하면서 실제 공정의 흐름과 일치하는가를 분석한다.

1) 수송카의 S_2 위치 도착

모든 초기상태에서 플레이스(X_{12})에 있는 토큰이 이동 가능함을 알 수 있다 따라서 플레이스(X_{12})의 토큰은 플레이스(Y_5)로 이동한다. 이때 플레이스(X_{12})의 토큰은 보존아크에 연결되어 있으므로 토큰이 제거되지 않는다.

2) 수송카의 S_2 위치 정지

플레이스(Y_5)의 토큰은 트랜지션(T_{K11})을 발화시켜 플레이스(K_{11})로 토큰을 이동시키게 되고, 플레이스(K_{11})의 토큰은 트랜지션(T_6)을 발화시켜 플레이스(P_6)에 토큰을 플레이스(P_1)로 이동시키게 된다.

3) 수송카 원료 탱크에 원료 입력

플레이스(Y_5)의 토큰은 트랜지션(T_{P3})을 발화시켜 플레이스(Y_3)에 토큰을 생성시킨다. 그리고 플레이스(Y_3)의 토큰은 트랜지션(T_{V11})을 발화시켜 플레이스(V_{11})로 토큰을 이동시키게 되고, 플레이스(V_{11})의 토큰은 트랜지션(T_1)을 발화시켜 플레이스(P_1)에 토큰을 플레이스(P_2)로 이동시키게 된다.

4) 원료 탱크에 원료 입력 완료

플레이스(P_2)의 토큰은 트랜지션(T_{L10})을 발화시켜 플레이스(X_{13})에 토큰을 제거하고, 트랜지션(T_{L21})을 발화시켜 원료가 레벨 L_2에 도달할 때까지 지연(200초)된 후에 플레이스(L_{21})로 토큰를 이동시킨다. 플레이스(L_{21})의 토큰은 트랜지션(T_{X13})을 발화시켜, 플레이스(X_{13})에 토큰을 생성시킨다. 그와 동시에 흡수 트랜지션(S_5)이 발화되어 플레이스(Y_3)와 플레이스(Y_5)의 토큰은 제거된다.

5) 수송카가 S_2 위치에서 S_1 위치로 이동

플레이스(X_{14})의 토큰은 트랜지션(T_{P1})을 발화시켜 플레이스(Y_1)에 토큰을 생성시킨다. 그리고 플레이스(Y_1)의 토큰은 트랜지션(T_{C11})을 발화시켜 플레이스(C_{11})로 토큰을 이동시키게 되고, 플레이스(C_{11})의 토큰은 트랜지션(T_2)을 발화시켜 플레이스(P_2)에 토큰을 플레이스(P_3)로 이동시키게 된다.

6) 수송카의 S_1 위치 도착

플레이스(P_3)의 토큰은 트랜지션(T_{S20})을 발화시켜 플레이스(X_{12})에 토큰을 제거하고, 트랜지션(T_{S11})을 발화시켜 수송카가 위치 S_1에 도달할 때 까지 지연(100초)된 후에 플레이스(S_{11})로 토큰을 이동시킨다. 플레이스(S_{11})의 토큰은 트랜지션(T_{X11})을 발화시켜, 플레이스(X_{11})에 토큰을 생성시킨다. 그와 동시에 흡수 트랜지션(S_4)이 발화되어 플레이스(Y_1)의 토큰은 제거된다.

7) 수송카의 S_1 위치 정지

플레이스(X_{11})의 토큰은 플레이스(Y_6)로 이동한다. 플레이스(Y_6)의 토큰은 트랜지션(T_{K21})을 발화시켜 플레이스(K_{21})로 토큰을 이동시키게 되고, 플레이스(K_{21})의 토큰은 트랜지션(T_3)을 발화시켜 플레이스(P_3)에 토큰을 플레이스(P_4)로 이동시키게 된다.

8) 수송카 원료 탱크에 원료 방출

플레이스(Y_6)의 토큰은 트랜지션(T_{P4})을 발화시켜 플레이스(Y_4)에 토큰을 생성시킨다. 그리고 플레이스(Y_4)의 토큰은 트랜지션(T_{V21})을 발화시켜 플레이스(V_{21})로 토큰를 이동시키게 되고, 플레이스(V_{21})의 토큰은 트랜지션(T_4)을 발화시켜 플레이스(P_4)에 토큰을 플레이스(P_5)로 이동시키게 된다.

9) 원료 탱크에 원료 방출 완료

플레이스(P_5)의 토큰은 트랜지션(T_{L20})을 발화시켜 플레이스(X_{14})에 토큰을 제거하고, 트랜지션(T_{L11})을 발화시켜 원료가 레벨 L_1에 도달할 때까지 지연(200초)된 후에 플레이스(L_{11})로 토큰를 이동시킨다. 플레이스(L_{11})의 토큰은 트랜지션

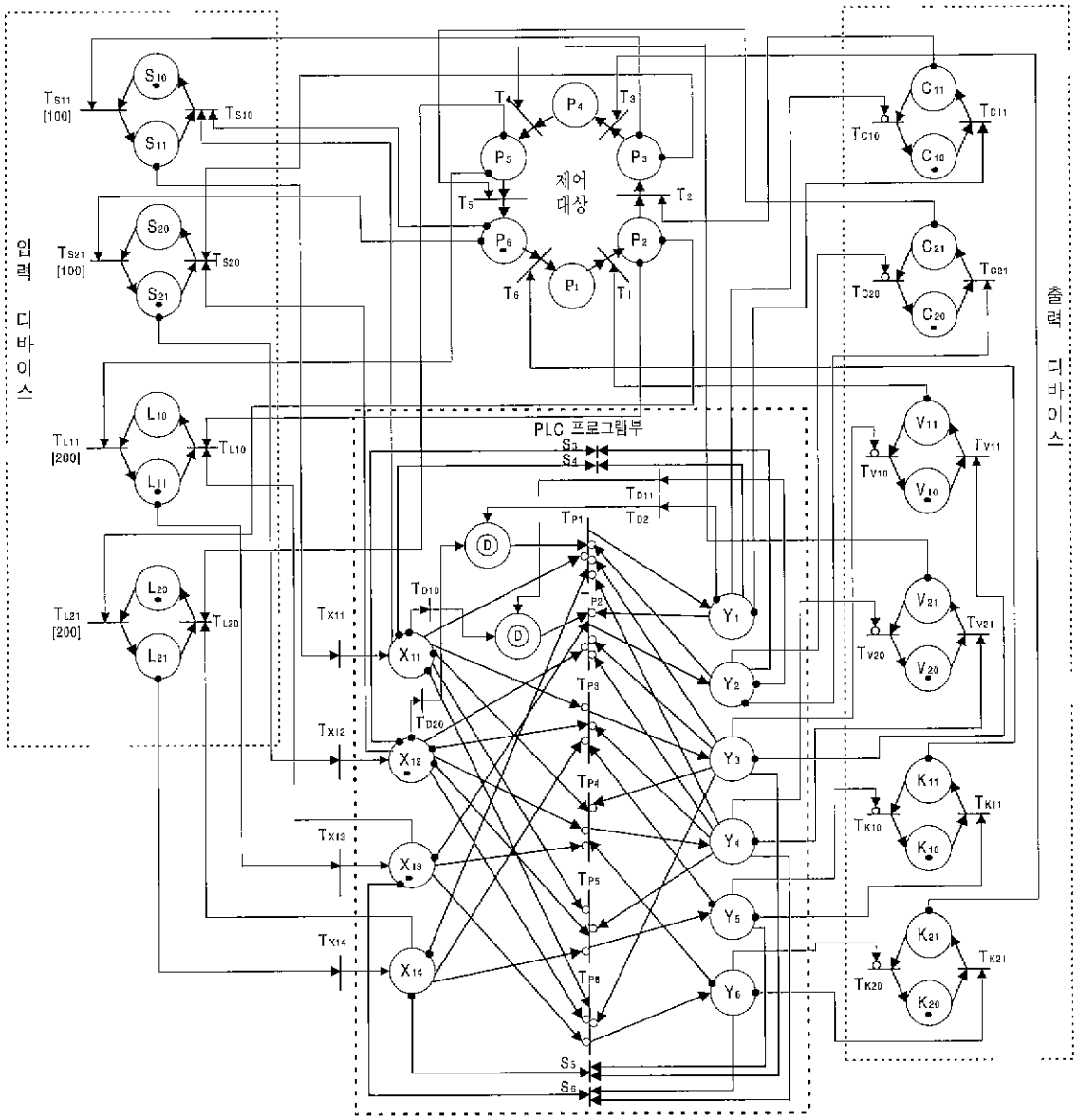


그림 14. 원료 수송카 작업 공정과 PLC 프로그램부를 결합한 통합 페트리네트

(T_{X13})을 발화시켜. 플레이스(X_{13})에 토큰을 생성시킨다. 그와 동시에 흡수 트랜지션(S_6)이 발화되어 플레이스(Y_4)와 플레이스(Y_6)의 토큰은 제거된다.

10) 수송카가 S_1 위치에서 S_2 위치로 이동

플레이스(X_{13})의 토큰은 트랜지션(T_{P2})을 발화시켜 플레이스(Y_2)에 토큰을 생성시킨다. 그리고 플레이스(Y_2)의 토큰은 트랜지션(T_{C21})을 발화시켜 플레이스(C_{21})로 토큰을 이동시키게 되고, 플레이스(C_{21})의 토큰은 트랜지션(T_5)을 발화시켜 플레이스(P_5)에 토큰을 플레이스(P_6)로 이동시키게 된다.

11) 수송카의 S_2 위치 도착

플레이스(P_6)의 토큰은 트랜지션(T_{S10})을 발화시켜 플레이스(X_{11})에 토큰을 제거하고, 트랜지션(T_{S21})을 발화시켜 수송카가 위치 S_2에 도달할 때 까지 지연(100초)된 후에 플레이스(S_{21})로 토큰을 이동시킨다. 플레이스(S_{21})의 토큰은 트랜지션($T_{\lambda 12}$)을 발화시켜. 플레이스(X_{12})에 토큰을 생성시킨다 그와 동시에 흡수 트랜지션(S_3)이 발화되어 플레이스(Y_2)의 토큰은 제거된다.

이와 같이 수송카의 작업공정에 대한 제어대상부와 수송카를 제어하는 PLC 프로그램부가 유기적으로

연결되어, 순차적 토큰의 흐름으로 해석될 수 있다.

8. 결론

원료 수송카 시스템에 대한 전체 공정을 모델링하고 해석하기 위하여 Modified 페트리네트를 도입하였다. 도입된 페트리네트를 사용하여 LD 제어언어를 페트리네트로 대응 변환하였다.

원료 수송 시스템의 진행 공정에 대한 대상 플랜트를 선정하고, 진행 공정에 따른 상태 플레이스를 설정하였다. 그리고 제어대상 및 제어장치의 상태 플레이스로부터 대상 플랜트에 대한 페트리네트 모델링을 구성하였다. 모델링된 페트리네트로부터 제어 대상부에 대한 수학적인 해석 방법을 제시하였다. 원료 수송카의 작업공정을 모델링한 페트리네트로 부터 상태표를 구하고, 그 상태표로부터 원료 수송카를 제어하는 LD 프로그램을 구하였다. 그리고 LD 프로그램으로부터 대응 페트리네트를 구하였다.

원료 수송카의 작업공정에 대한 페트리네트와 LD 프로그램에 의한 페트리네트를 결합한 통합 페트리네트를 구현하였다. 두가지 프로세서를 하나로 표현함으로서, 전체 작업 공정 및 신호 흐름도에 대한 일괄적인 분석이 가능하게 되었다. 또한 단일 페트리네트 구조상에서 외부로부터 토큰의 생성 또는 소멸됨이 없이 토큰의 이동에 따른 공정 분석이 가능하게 되었다.

참고문헌

[1] A. Falcione and B. H. Krogh, "Design recovery for relay ladder logic," IEEE Control Systems, pp. 90-98, Apr., 1993.

[2] Rene David and Hassane Alla, Petri Nets and Grafcet, Prentice-Hall Inc., 1992.

[3] I. G. Warnock. Programmable Controllers. Operation and Application, Prentice-Hall Inc., 1988.

[4] R. W. Lewis, Programming industrial control systems using IEC 1131-3, The Institution of Electrical Engineers, 1995.

[5] Kurpati Venkatesh, Meng C. Z., Reggie J. Caudill, "Comparing Ladder Logic Diagrams and Petri Nets for Sequence Controller Design Through a Discrete Manufacturing System," IEEE Trans. on industrial electronics, Vol. 41, No. 6, pp. 611-619, Dec., 1994.

[6] Kurt Jensen, Coloured Petri Nets, Springer-Verlag, 1996.

[7] Luis Gomes, A. Steiger-Garcao, "Programmable controller design based on a synchronized colored petri net model and integrating fuzzy reasoning," pp. 218-237, Portugal.

[8] Martin O., Stefan J., Karl A., "Implementation aspects of the PLC standard IEC 1131-3," Control Engineering Practice, pp. 547-555, 1998.

[9] Melco, Melsec A-series PLC CPU manual, 1995

[10] N. Chamas, L. Anneberg, E. Yaprak, "Timed neural petri nets," IEEE, pp. 926-929. 1993.

[11] POSCON, POSFA PLC CPU Programming manual, 1994.

[12] R. Valette, M. Courvoisier, JM. Bigou, J. Albukerque, "A petri net based programmable logic controller," Computer Applications in Production and Eng., pp. 103-116, 1983.

[13] Klockner Moeller, FuzzyTECH 40 Programming software maual. 1997.

[14] Rene David, Hassane alla, "Petri nets for modeling of dynamic systems - a survey," Automatica, vol. 30, no. 2, pp. 175-202, 1994.

[15] Rene David, "Grafcet: A powerful toll for specification of logic controllers," IEEE Trans. on control systems technology, vol. 3, no. 3, pp. 253-268, 1995.

[16] Syed I. Ahson, "Petri net models of fuzzy neural networks," IEEE Trans. on systems, man, and cybernetics, vol. 25, no. 6, pp. 926-932, Jun., 1995.

[17] Tiehua Cao, Arthur C Sanderson. Intelligent task planning using fuzzy petri nets, World Scientific, 1996.

[18] Tadao, Murata, "Petri nets : Properties, analysis and applications," Proceedings of the IEEE, vol. 77, no. 4. pp. 541-580, Apr., 1989.

[19] 전기학회, 시스템 제어 공학, 오옴사(일본 출판사), 1988 .

[20] 장래혁, "페트리네트의 구현 방법과 순차 제 어기에의 응용." 서울대 박사학위논문, 1996.

[21] 이기범, "PLC O/S 알고리즘 구현," 대한전기학회 '95하계 학술대회논문집, Jul., 1995.

[22] 이기범, "중대형 고기능 PLC system 개발," 계측제어기술 심포지움, Jan., 1996.

[23] 이기범, 주문갑, "고신뢰성을 위한 PLC 시스템 이중화 구현," 대한전기학회 '97하계학술대회, pp. 2485-2487, Jul., 1997.

[24] 이기범, 이진수, "PLC의 고장 발생 주기에 대한 failure rate 산출과 redundancy 시스템의 MTTF/reliability 평가," '98자동제어학술회의논문집, pp. 384-387, Sep., 1998.

[25] 이기범, 이진수, "PLC 프로그래밍을 위한 가상 플랜트 시뮬레이터 구현," 대한전기학회 '98추계학술대회, pp. 475 - 477, Nov., 1998.

[26] 이기범, 이진수, "이산 사건 시스템에 대한 페트리네트 모델링과 해석," 제어계측.자동화 로보틱스 합동학술발표 논문집, pp. 206-209. Mar., 1999.

이 기 범(李 基 範)

1961년생. 1984년 광운대학교 전자공학과 졸업(공학사). 1986년 광운대학교 전자공학과(공학석사). 1996년-현재 포항공과대학교 전자전기공학과 박사과정. 1987년-1996년 ㈜포스콘 과장 : POSFA PLC개발. 1996년-현재 포항산업과학연구원 선임연구원. 주요 관심 분야는 공장 자동화 시스템 개발(PLC, DCS), 지능제어 및 페트리네트 응용.

이 진 수(李 振 秀)

1953년생. 1975년 서울대학교 전자공학과(공학사).

1980년Univ. of California, Berkeley(공학석사). 1984년 Univ. of California, Berkeley(공학박사). 1984년-1985년 AT&T Bell Laboratories 연구원. 1985년-1989년 GE Aerospace 고등기술연구소 책임연구원. 1989년-현재 포항공과대학교 전자전기공학과 조교수, 부교수, 교수. 주요 관심 분야는 Fuzzy Control, Adaptive Control, Robotics/Mobile Robotics, Factory Automation, Control Theory/Application.