한국 CAD/CAM 학회 논문집
제 14권 제 1호 2009년 2월 pp. 60-68

PLC 프로그램 생성을 위한 SOS-Net

고민석*, 홍상현**, 왕지남***, 박상철****

SOS-Net for Generattion of PLC Program

Min-Suk Ko*, Sang-Hyun Hong**, Gi-Nam Wang*** and Sang-Cheul Park****

ABSTRACT

Because of the reduced product life-cycle, industries are making an effort to bring down the process planning time. In the traditional approach, we have to analyze established process planning, then design the time chart based on process information and drawing the formal time chart such as SOP (sequence of operation). Thereafter, it will be converted to PLC code that is a time consuming and redundant job. Similarly, Industrial automated process uses PLC Code to control the factory; however, control information and control code (PLC code) are difficult to understand. Hence, industries prefer writing new control code instead of using the existing one. It shows the lack of information reusability in the existing process planning. As a result, to reduce this redundancy and lack of reusability, we propose SOS-Net modeling method. Unlike past stabilized process planning that is rigid to any change; our proposed SOS-Net modeling method is more adaptable to the new changes. The SOS-Net model is easy to understand and easy to convert into PLC Code accordingly. Therefore, we can easily modify the control information and reuse it for new process planning. The proposed model plays an intermediary role between process planning and PLC code generation. It can reduce the process planning and implementation time as well as cost.

Key words : SOS-Net, PLC (Programmable Logical Control), State, SOP (Sequence of Operation), LD

1. 서 론

생산시스템은 일반적으로 다양한 자동화 설비로 구성되어 있으며, 이러한 생산시스템의 설계 및 검증은 대단히 중요하다. 생산시스템의 설계는 매우 복잡한 단계를 거치며, 특히 생산시스템의 제어로직 설계 및 검증은 많은 기간 및 노력을 요구한다. 왜냐하면, 생산시스템의 제어는 다양한 구성설비들의 동적인 상황을 실시간으로 감시하면서 모든 발생 가능한 상황(Possible States)을 고려하여 그에 알맞은 명령(Command)을 설비들에게 내려 주어야 하기 때문이다. 또한 만약 이러한 제어로직에 문제가 발생하면 생산시스템의 능률 저하는 물론이고 심각하게는 생산시스템 자체가 파손될 수도 있어서 기업의 큰 손실을 야기 할 수 있다.

일반적으로 생산시스템의 제어를 위해서는 PLC를 사용하며, PLC에 연결된 각 설비들의 작업 시작 신호 및 완료 신호를 토대로 공정을 진행시킨다[9].

Fig. 1은 생산시스템을 제어하는 PLC의 역할 관계를 나타낸다. 공정을 구성하는 각 설비는 현재 상태를 PLC에 알리기 위해 센서를 사용하는데, 이 센서들로부터 감지되는 신호는 PLC 입력신호 역할을 한다. PLC는 입력신호를 바탕으로 공정 및 각 설비의 현재 상태를 파악하여, 다음 작업 진행을 위한 출력 신호를 내보낸다[4,12].

설비는 PLC 출력 신호를 바탕으로 작업을 진행하고, 작업진행결과는 센서를 통해 다시 PLC로 전해진다. 이 같은 프로세스를 갖는 PLC는 스캐닝 사이클(Scanning Cycle)을 통해 프로그램 진행 중에도, 다음 프로세스를 계속 업데이트 하는데, 이는 입, 출력 정

*학생회원, 아주대학교 산업공학과
**정회원, 현대기아자동차, 설비제어기술팀 과장
***비회원, 아주대학교 산업공학과
****교신저자, 종신회원, 아주대학교 산업공학과
- 논문투고일: 2008. 11. 03
- 논문수정일: 2008. 12. 09
- 심사완료일: 2008. 12. 24

보 및 내부 연결(Internal Relay) 신호를 이어주는 불리안 로직(Boolean Logic)을 스캔하는 것이다. 사이클을 스캔하는 동안 출력 및 내부 연결테이블은 계속 업데이트 되어 다음 사이클을 위한 새로운 신호를 준비한다[8].

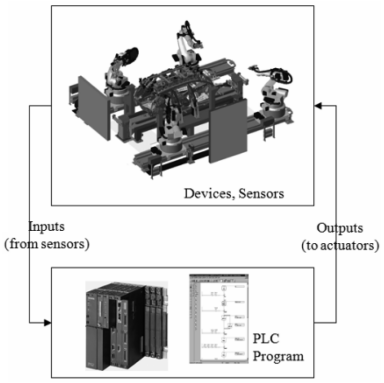


Fig. 1. Production system controlled by a PLC program.

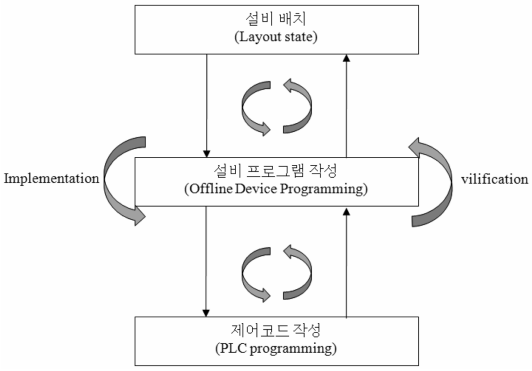


Fig. 2. Three layers of production system development.

Fig. 2는 제어로직(Control Logic) 작성을 위해 필요한 생산시스템 개발 단계를 나타낸다. 처음 설비 배치 단계(Layout State)는 생산 시스템 청사진을 작성하는 과정으로써, 생산 요구 사항에 따른 설비 목록을 정리하고, 이를 위한 기계 도면 작성 및 배치를 수행하는 단계이다. 설비 프로그램 작성과정(Offline Device Programming)은 배치된 설비들의 기능 및 역할을 토대로, 적합한 동작 수행을 위한 프로그램 작성 및 시뮬레이션을 수행하는 단계이다. 이 단계를 통해 설비간섭(Collision)을 미연에 방지하고, 설비 동작 오차를 최소화(Calibration)시킬 수 있다. 특히, 현대 생산 시스템의 주축을 이루는 로봇 공정의 경우, OLP (Off-Line Programming)를 통해 로봇간 협업 및 간섭을 고려한 최적 경로 작성이 본 단계에서 수행된다. 마지막 제어코드(PLC 코드) 작성과정은 공정이 순차적, 계획적으로 진행될 수 있도록 PLC 프로그램을 작성하는 단계이다. 이 단계에서는 배치된 설비의 입, 출력 점과 PLC를 연결하여, 프로그램 테스트 및 시운전에 들어간다. 이 같은 세 단계를 진행하는 동안, 제어 프로그램의 오류 및 설비 프로그램의 오류를 계속적으로 수정하여, 라인 구동이 가능한 PLC 코드가 만들어 진다[3].

설비 프로그램작성 단계는 크게 두 가지 경우로 나눠지는데, 첫째, 독립적인 컨트롤러를 갖고 있는 설비의 프로그램과 둘째, PLC에 의해 직접 운용되는 설비의 프로그램이다. 독립적 컨트롤러를 갖는 가장 대표적 설비는 로봇이다. 로봇 프로그램의 주 목적은 로봇 작업 과정에서 주변 사물과 충돌이 일어나지 않고 정확하게 목표에 도달하도록 하는 것과, 가장 효과적 경로를 작성하는 것이다. 과거에는 로봇 펜던트(Pendent)를 통해 직접 프로그램을 한 반면, 현재는 컴퓨터와 연계되는 OLP를 통해 보다 정확하고 효과적인 로봇 프로그램이 가능해 졌다. 이와 같은 OLP 시스템을 활용함으로써 현장 작업자의 작업시간을 최소화하고 사항환경상에서 로봇의 작업자세나 작업 순서 등의 조건을 바꾸어 가며 다양한 시뮬레이션을 수행하여 최적의 로봇 작업 프로그램을 작성할 수 있게 되었다[13]. PLC에 의해 직접 제어되는 설비들은 신호에 따른 작업 동작을 명시하고, 각 작업 수행을 위해 필요한 하위 동작 제어프로그램을 작성한다. 이 같은 하드웨어, 개별 제어 프로그램 작성 단계를 거친 후, 개략적 타임차트(Time-chart)인 SOP(Sequence of Operation)가 만들어진다[11].

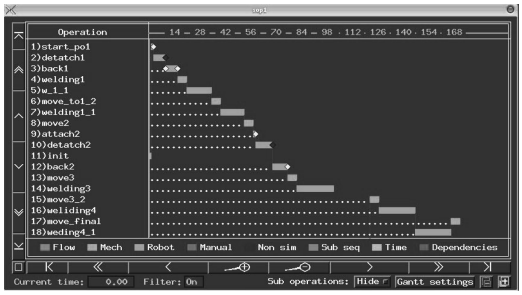


Fig. 3. Sequence of Operation (SOP).

SOP는 전후 공정 순서, 투입 디바이스의 자세한 연결관계를 정의하여, 프로세스의 큰 틀을 제시하는 시간에 따른 공정 계획이다. Fig. 3은 현업에서 사용하는 로봇 시뮬레이션의 SOP 예제이다. 하지만, Fig. 3

과 같은 SOP는 개략적 공정 연결에 의미를 두기 때문에 PLC 코드 작성을 위한 정보가 충분히 포함되어 있지 않다. 왜냐하면, PLC 코드는 센서조건의 만족을 통해 동작을 진행하는데, SOP에는 센서, 액츄에이터(Actuator), 인터락(Interlock)과 같은 제어 신호 정보가 포함되어 있지 않고, 센서를 통한 동작 상태확인이 불가능하기 때문이다. 상용 소프트웨어인 UG사의 EM-PLC는 SOP로부터 PLC코드를 생성하는 기능을 지원하고 있지만, 이 역시 선후 연결 정의만 가능하고 제어정보가 불충분하여 현장 사용에 부적합하다.

본 논문에서는 SOP가 갖는 제어 정보 부재에 따른 문제점을 해결하고, 계획 초기 단계부터 제어단계를 고려한 모델링을 수행할 수 있는 새로운 모델링 방법론인 SOS-Net을 제안하고자 한다. 제 2장에서는 간단한 예제 설명과 함께, 기존 SOP를 이용한 PLC코드 생성의 한계를 설명할 것이며, 제 3장에서는 SOS-Net의 정의 및 모델링 방법을 설명할 것이고, 제4장에서 실제 모델링 사례를 말할 것이며, 제 5장에서 결론을 말할 것이다.

2. 기존 SOP를 이용한 PLC 생성의 한계

본 2장에서는 AGV를 이용해 파트를 운반하는 간단한 예제를 설명하고, 이를 SOP로 만들어 봄으로써, 제어 정보에 관한 SOP의 한계를 말할 것이다. Fig. 4는 파트를 Position 1으로부터 Position 4로 운반하는 간단한 예제이다.

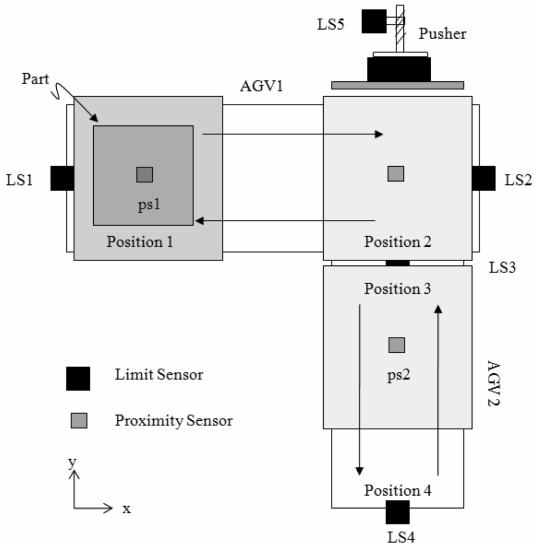


Fig. 4. AGV cell.

파트가 AGV 1을 통해 position 1에서 position 2로 이동되고, 푸셔(Pusher)는 파트를 position 3으로 이동시킨다. 이동된 파트는 AGV 2를 통해 position 4에 도착하고, 작업은 완료된다. 본 예제에서의 상태 변화는 아래와 같이 총 12단계로 진행된다.

① AGV1은 Position1위치에서 파트를 싣고 있다.
② AGV2는 Position4 위치에서 파트를 싣고 있지 않다.
③ AGV1은 파트를 싣고 Position 1에서 Position 2로 이동한다.
④ AGV2는 파트를 싣지 않고 Position 4에서 Position 3로 이동한다.
⑤ AGV1은 Position 2에 도착한다.
⑥ AGV2은 Position 3에 도착한다.
⑦ 푸셔는 AGV1에있는 파트를 AGV2로 이동시킨다.
⑧ 푸셔는 자신의 초기위치로 복귀한다.
⑨ AGV1은 파트를 싣지 않고 Position 2에서 Position 1로 이동한다.
⑩ AGV2는 파트를 싣고 Position 3에서 Position4로 이동한다.
⑪ AGV1은 Position 1에 도착한다.
⑫ AGV2은 Position 4에 도착한다.

이 같은 3개 설비의 상태 변화를 통해 파트는 Position 1에서 Position 4로 이동되며, 이를 SOP로 모델링하면 Fig. 5와 같이 나타낼 수 있다. 단, Fig. 5의 P1은 Position 1, P2는 Position 2, P3는 Position 3 그리고 P4는 Position 4를 의미한다.

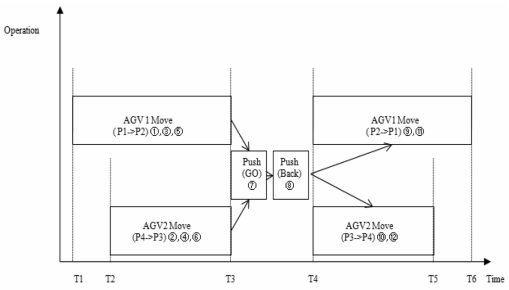


Fig. 5. SOP for AGV cell.

Fig. 5의 "AGV1 Move"와 같은 표기는 사람이 이해할 수 있는 작업내용을 말하며, 아래번호는 12가지 상태 중 해당되는 상태 번호를 말한다. 실제 이 상태 번호는 SOP에 표기되지 않으나, 이해를 위해 함께 표기 하였다. SOP는 그림과 같이 작업내용을 시간에 따라 표기 함으로써, 작업의 선후 관계를 명확히 할 수

있고, 전체 소요시간을 통해 개략적 사이클 타임을 계산할 수 있다. 하지만, SOP는 시간에 국한하여 작업을 연결하기 때문에, 실제 동작과는 큰 차이를 갖는데, 가장 큰 이유는 작업의 상태를 확인시켜 주는 센서정보가 존재하지 않기 때문이다. 센서는 파트의 감지, 설비 상태 확인, 작업의 시작 및 종료 등 공정의 여러 부분에 걸쳐, 입력신호를 산출하는 역할을 한다. PLC에 따른 공정의 진행 역시 이 같은 센서신호에 의해 결정되는데, SOP는 이 같은 논리 제어 정보가 없기 때문에, 실제 공정 진행과 큰 차이를 갖게 되는 것이다. SOP가 갖는 제어정보의 부재는 공정 진행에서 문제를 야기시킬 수 있는데, 예를 들어 AGV 2가 position 2에 도달하지 않은 상태에서 푸셔가 작동할 경우, 파트는 경로를 이탈하게 된다. 또는 파트가 position 2에서 position 3으로 이동하지 않은 상태에서 AGV 1이 이동하게 되면, 새로운 파트가 잘못 놓여지거나 추락하는 문제가 발생할 수 있다. 센서에 의한 제어가 아니고, SOP에 의한 움직임으로 공정을 계획, 진행시킬 경우, 이 같은 잠재적 문제점을 갖게 된다.

만약 공정 계획 초기에 제어정보를 고려한 계획을 수립할 수 있다면, 앞의 문제점과 같은 불합리한 점들은 사전에 막을 수 있게 된다. 시간에 기초한 SOP와 달리 제어 신호를 기반으로 계획된 공정계획은 단계가 진행될수록 정보가 추가되기 때문에 보다 생산적이며, 사실적인 모델링이다.

본 논문에서는 SOP가 갖는 제어 정보 부재에 따른 문제점을 해결하고, 계획 초기 단계부터 제어단계를 고려한 모델링을 수행할 수 있는 새로운 모델링 방법론인 SOS-Net(Sequence of States-Net)을 제안하고자 한다.

3. SOS-Net의 정의

본 논문에서 제안하는 SOS-Net은 Sequence of State Nets의 약자로써, 공정 진행상태을 순차적 관계로 연결한 제어기반의 공정 모델링 방법론이다. SOS-Net을 제안하는 이유는 제어 정보 기반의 모델을 만들어 체계적인 PLC 코드를 생성하는 것이다. 여기서 체계적 PLC코드 생성이란, 디바이스, 단위 공정 과 같이, 그 단위가 명확하게 정의되는 대상에 대하여, 사전에 정의된 입, 출력 신호를 이용해 제어코드의 큰 틀을 만드는 것을 말한다.

특정 시점에서, 디바이스 상태를 토대로 작업 수행 여부를 결정짓는 것이 제어코드의 역할이다. 따라서 코드에 기술될 수 있는 디바이스의 상태를 사전에 명확하게 정의하는 것 체계적 코드의 필수조건이라 말할 수 있다. SOS-Net을 만들기 전에 디바이스 별 신호조합을 만들어야 하는데, 이를 표로 정리한 것이 Table 1이다.

Table 1. Device sensor signal configuration

Device	Sensor signal configuration
AGV 1	LS1-LS2-PS1
AGV 2	LS4-LS3-PS2
Pusher	LS5

예제에서 사용되는 디바이스는 Table 1첫 번째 열에 있으며, 두 번째 열에는 각 디바이스의 상태를 알려주는 센서 신호 조합이 표기되어 있다. 특히, 센서 신호 조합을 구성하는 신호명의 순서는 Table 2, 3, 4의 신호 값과 연결되는데, 각 센서의 역할을 설명하면 아래와 같다.

① LS(Limit Sensor): 리미트 센서는 임의의 물체가 물리적으로 닿아 있을 때, 신호가 “ON”이 되는 신호로써, AGV의 위치 상태를 나타낸다. 예를 들어, AGV1의 위치는 LS1과 LS2의 조합으로 말할 수 있는데, LS1이 ON 되고 LS2가 OFF된 상태는 AGV1이 Position1에 놓여 있음을 말하는 것이다.

② PS(Proximity Sensor): 근접 센서는 금속 물체에 의한 자기장의 변화로 물체를 감지할 때, “ON”되는 신호로써, AGV의 파트감지 상태를 나타낸다. 예를 들어, PS1이 ON되어 있으면, 이는 AGV1위에 파트가 놓여 있음을 말하는 것이다.

Table 2. Operation signal configuration (AGV1)

State	Sensor signal configuration
$S1_a$	1-0-1
$S1_b$	0-0-1
$S1_c$	0-1-1
$S1_d$	0-1-0
$S1_e$	0-0-0
$S1_f$	1-0-0

Table 3. Operation signal configuration (Pusher)

State	Sensor signal configuration
$S2_a$	1
$S2_b$	0

Table 4. Operation signal configuration (AGV2)

State	Sensor signal configuration
$S3_a$	1-0-0
$S3_b$	0-0-0
$S3_c$	0-1-0
$S3_d$	0-1-1
$S3_e$	0-0-1
$S3_f$	1-0-1

간단한 예제이기 때문에, 실제 현업에서 사용되는 많은 종류의 센서를 다 다루지는 않았다. 하지만 센서 역할이 특정 행동, 상태를 감지하여, "ON", "OFF" 신호를 출력하는 역할을 하는 것이므로, 위 예제로 SOS-Net에서의 센서의 역할은 충분히 설명될 수 있다. SOS-Net에서 말하는 제어시스템의 state는 의미 있는 센서 조합을 말하는데, Fig. 6의 센서조합을 이용해 디바이스 별 state를 표로 정리한 것은 표(Table 2, 3, 4)와 같다.

위 표와 같이 디바이스 별 state를 명확하게 정의한 목적은 의미 있는 state 조합을 추출하여, 디바이스 작업(Operation)을 순차적으로 제어하기 위함이다. 예를 들어 AGV1의 경우 3개의 센서 조합이 8(=2^3)개의 상태를 만들 수 있지만, "1-0-0", "1-1-0"과 같은 경우는 본 공정에서 의미가 없는 조합이기 때문에 Table에서 제외하였다. 아래 Table 5는 Fig. 5에서 언급된 디바이스 별 작업을 상태조합으로 설명한 표이다.

Table 5. Device operation configuration

Device	Operation description	Operation configuration
AGV1	AGV 1 Move (p1→p2)	$S1_a$-$S1_b$-$S1_c$
AGV1	AGV 1 Move (p2→p2)	$S1_d$-$S1_e$-$S1_f$
AGV2	AGV 2 Move (p4→p3)	$S3_a$-$S3_b$-$S3_c$
AGV2	AGV 2 Move (p3→p4)	$S3_d$-$S3_e$-$S3_f$
Pusher	Push the Part	$S2_a$-$S2_b$

Table 5의 두 번째 열은 Fig. 5에서 언급한 SOP의 작업내용과 같으며, 이 내용을 state 조합으로 표기한 것이 세 번째 열이다. 두 번째 열의 작업내용 기술부분은 사람이 인지하기 위한 표기이며, 세 번째 열은 제어 관점에서 디바이스 상태 변화를 표기한 것이다. 예를 들어 표의 "AGV1 Move(p1→p2)"는 AGV1이 Position 1에서 Position 2로 이동하는 작업을 나타내며, 이때 AGV1의 상태는 $S1_a$-$S1_b$-$S1_c$로 변화한다. SOS-Net의 기본 목적이 사람이 인지하기 쉬운 제어프로그램 생성에 있으므로, Table 5 같은 정보 연결은 필수적이라 하겠다.

Fig. 6은 Fig. 4예제를 SOS-Net으로 모델링 한 결과이며, 이를 구성하는 요소들은 다음과 같다.

① Terminal-Node: Terminal-Node는 단위 공정의 시작과 종료를 나타내는 노드이다. 본 예제에서는 파트가 Position 1에서 Position 4로 되는 상태변화가 하나의 단위 공정이 된다. 비록 본 예제는 작은 단위의 공정이지만, 현업의 라인은 매우 복잡한 연결관계를 갖는 방대한 공정이다. 따라서, Terminal-Node를 통해 제어코드 생성의 대상이 되는 공정의 범위를 명확히 구분해야 한다.

② State-Node: State-Node는 센서신호 조합으로 만들어진 state로써, Table 2, 3, 그리고 4에서 정의된 각 디바이스의 상태를 나타낸다.

③ Operation-Node: Operation-Node는 디바이스 동작을 나타내는 노드로써, 상태 표시와 함께, 출력 신호 산출을 표기한다. 본 노드는 State-Node와 같이 센서 신호 조합으로 만들어진 스테이트이다. 그러나 디바이스 동작을 위한 출력 신호의 정보가 추가되었다는 점에서 차이를 갖는다. 예를 들어, $S1_b$의 경우, AGV1이 Position 1에서 출발하여 Position 2로 이동하는 전진 상태를 설명하며, AGV1 이동을 위한 실린더 동작 출력신호를 산출해야 함을 말하고 있다. AGV1동작 출력 신호는 $S1_b$(전진), $S1_c$(후진)에서, AGV2은 $S3_b$(전진), $S3_c$(후진), 푸셔는 $S2_b$에서 출력된다. AGV의 경우 전, 후진을 위한 복동(復動) 실린더, 푸셔는 단동(丹動) 실린더 사용을 가정하기 때문에, 푸셔의 후진 Operation-Node는 고려하지 않는다.

④ AND-Node(OR-Node): AND-Node 및 OR-Node는 노드로 입력되는 신호를 연산하여 결과 신호를 산출하는 노드이다. AGV1과 AGV2가 서로 연결되는 AND 2 노드에서, 입력 신호는 $S1_c$, $S3_b$이다. 따라서 AND 2노드는, AGV1이 Position 2에, AGV2는 Position3에 있음을 보장해 주는 역할을 한다. 만약에 이 같은 보장이 전제되지 않으면, AGV2가 Position 3에 도착하지 않은 상태에서 푸셔가 작동할 수 있고, 그 결과 파트가 추락할 수 있다. 즉, 이 노드는 신호의 논리 연산을 통해 조건을 만드는 역할을 한다.

⑤ Output Signal: Operation Node로 도달하기 위한 디바이스 동작 신호이다.

⑥ State Signal: 현재 state와 다음 state를 연결해주는 신호로써, 선, 후 state의 연결을 표기하는

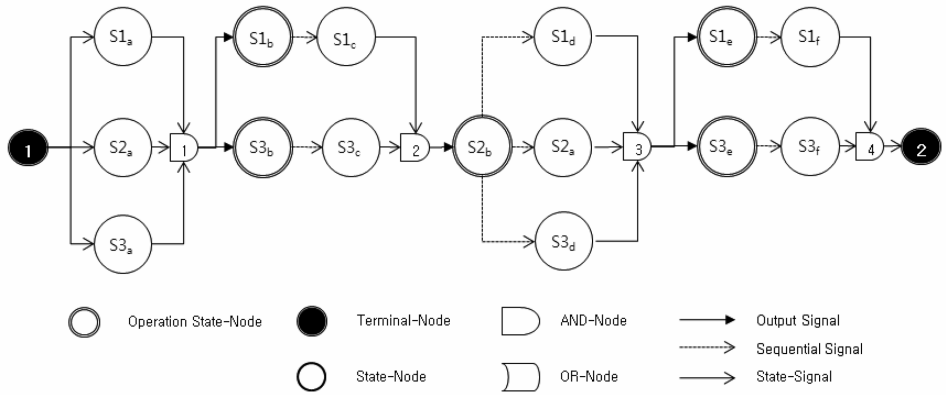


Fig. 6. SOS-Net for AGV cell.

역할을 한다. 특히 동작의 결과가 반영되는 state와 연결될 때 사용된다.

⑦ Sequential Signal: 실제로 공정 진행에 영향을 주지는 않지만, SOS-Net의 순차적 연결을 보여주는 역할을 한다. 특히 Output 노드 실행 후, state로 연결되는 부분에 사용되어, 실행 결과가 영향을 미치는 state를 가리켜 준다.

이상의 내용을 토대로, Fig. 6의 내용을 정리하면 다음과 같다.

① Terminal-Node_1에서 공정 시작을 위한 전제 조건(전 공정 완료신호, 안전 신호등)을 확인하고, 공정 시작을 나타낸다.

② 디바이스 초기 state($S1_a$, $S2_a$, $S3_a$) 상태를 확인하여, AND_1 신호를 만든다. 즉, AND_1은 AGV1이 Position 1에, AGV2가 Position 4, 푸셔는 복귀돼있는 상태를 나타내는 것이다.

③ AND_1 신호는 AGV1의 전진, AGV2의 전진 동작을 명령한다. AND_1 신호를 토대로 각 AGV 복동 실린더의 전진 출력을 수행하는 것이다.

④ AGV1은 Position 2, AGV2는 Position 3에 도달함에 따라 $S1_c$, $S3_c$ 상태가 되고, 두 디바이스의 상태를 보장하는AND_2 신호가 만들어 진다.

⑤ AND_2 신호는 푸셔의 전진 동작을 명하여, $S2_b$ 상태로 전이를 유도한다. 전진 후에 푸셔가 복귀하면, $S2_a$ 상태로 바뀌게 된다. $S2_b$와 $S2_a$는 sequential signal로 연결되어 있다.

⑥ AGV의 위치를 보증하는 AND_2 신호와 푸셔의 복귀 신호를 토대로 AND_3 신호를 만든다.

⑦ AND3신호는 AGV 상태 확인 state로 연결되어, $S1_d$, $S3_e$로 연결된다.

⑧ AND3신호는 AGV 상태 확인 state로 연결되어, $S1_e$, $S3_e$로 연결된다.

⑨ AGV1은 파트가 없는 상태에서 Position 1로 복귀하며($S1_f$), AGV2는 파트를 가지고 Position 4로 복귀한다($S3_f$).

⑩ AGV가 모두 복귀하면, AND_4 신호가 만들어지고 이는 Terminal-Node 2로 연결되어 공정이 종료된다.

Fig. 5 SOP에서도 알 수 있듯이, 디바이스 작업의 내용 및 작업 간 연결은 쉽게 모델링 될 수 있어야 한다. 하지만, 쉽게 모델링 된 작업내용일수록 제어 정보로 표현하기는 어려워 진다. 왜냐하면, 사람이 이해하는 직관적 모델링 정보를 디바이스 단위로 하향화 시키는 것은 많은 절차 및 준비 조건을 요구하기 때문이다. SOS-Net은 이 같은 정보의 하향화를 보다 쉽고, 체계적으로 진행시킬 수 있도록 하는 모델링 방법으로써, 사람이 이해하는 정보 모델을 제어단위까지 확장 시킬 수 있다.

4. PLC코드 생성

SOS-Net 모델의 궁극적인 목적은 PLC 코드 생성에 있다. 체계적으로 작성된 신호 명세와 공정 계획을 토대로 작성된 SOS-Net은 공정 선후 관계를 밝힐 뿐 아니라, 신호를 기반한 디바이스 state를 표기해 주기 때문에, 기존의 PLC 코드 보다 이해하기 쉽다. 아래 Fig. 7은 Fig. 6의 SOS-Net을 LD(Ladder Diagram)으로 변환한 PLC 코드이다.

Fig. 6과 Fig. 7를 함께 보면, 비 전문가도 쉽게 이해할 수 있을 정도의 수준으로 간단하고 명확하게 코드가 진행된 것을 알 수 있다. 많은 PLC 언어(LD, SFC, FB 등)들 중에서 LD로 변환한 이유는, LD가

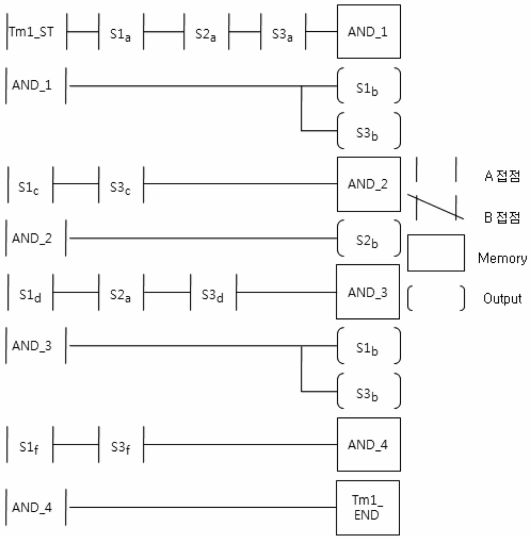


Fig. 7. PLC code for AGV cell example.

가장 기본이 되는 언어이기 때문이다[9].

Fig. 7 코드에 사용되는 요소는 4가지로, A접점, B 접점, 메모리, 아웃풋이 있다. A접점은 신호가 들어올 때 "True"가 되며, B 접점은 반대로 "False"가 된다. 메모리는 특정 조건을 만족 시킬 경우, "True"가 되며, Output은 디바이스와 연결되어, 동작 수행으로 이어진다. Fig. 7는 LD로 작성된 의사코드(pseudo code)이기 때문에, 메모리 해제(Reset) 및 디바이스 별 FB 코드 연결은 고려하지 않고, SOS-Net에 명시된 정보를 코드로 변환하는데 중점을 두었다. 특히, Table 2, 3, 4와 같이 각 state를 만족시키기 위한 조건들의 코드는 다루지 않았는데, 이는 코드 작성자의 특성에 따라 여러 종류의 PLC 코드로 작성될 수 있기 때문에 가장 위 범위의 LD코드를 제시하였다. SOS-Net의 코드 생성의 룰은 다음과 같다.

① Terminal Node의 시작은 코드의 가장 앞 단에 위치한다. 전(前)공정의 완료 신호를 토대로 현 공정을 진행시키는 것이기 때문에, 가장 앞에서 기본 조건으로 사용된다.

② State Node는 코드진행을 위한 조건으로 사용되며, 일반적으로 A접점을 통해 표현된다. Table 2, 3, 4에 명시된 센서조합을 만족시킬 경우, 해당 state의 A접점은 "True"가 된다.

③ AND(OR) Node는 입력되는 Node 신호의 연산을 통해 메모리 신호를 "Set"시킨다. 본 신호는 특정 조건을 만족시킬 때까지 메모리 상에서 Set 상태를 유지하며, 후에 Reset을 통해 상태를 회복한다.

④ Output Node는 output signal을 통해 디바이스 동작을 명하는 출력신호이다. 따라서 LD 상의 Output 접점과 연결되어, 동작을 행하는 역할을 한다.

⑤ Sequential Signal은 Output 신호 결과가 반영된 state를 바로 다음 줄의 조건으로 넣는 순서 보장의 역할을 한다.

⑥ State Signal은 State 조건 사이를 이어주는 신호이다. 따라서 A, B 접점 사이, 접점과 메모리 신호를 연결해 주는 일반적 코드연결과 같다.

⑦ Output Signal은 연결 시작 부분의 조건 만족을 Output Node로 연결시키는 역할을 한다. 따라서 A, B 접점으로부터 Output 접점으로 신호를 전달해 주는 역할을 한다.

Fig. 7 코드의 일부 내용을 간단히 설명하면 다음과 같다. 첫 줄은 SOS-Net의 처음 Terminal-Node에서 AND_1로 이어지는 부분까지이다. 처음 A접점 "Tm_1"은 Terminal-Node 1에 대응하는 신호로써, 직전 공정의 완료 신호를 조건으로 한다. "Tm_1_st"는 공정 1의 시작(Start)를 나타내고, "Tm_1_end"는 공정 1의 종료(end)를 말한다. 다음 3개의 접점($S1_a$, $S2_a$, $S3_a$)은 SOS-Net에서 설명하였듯, 디바이스 초기 상태를 점검하는 코드이다. 4개의 조건을 만족시키면 AND_1 메모리가 Set되며, 이는 두 번째 줄 코드에 입력조건으로 사용된다.

Fig. 7과 같이 생성된 코드는 초기코드(Initial Code)로서, 실제 사용할 수 있는 코드는 아니다. 하지만, 공정계획에 따른 코드의 가장 큰 흐름을 잡아주기 때문에, 추가해야 하는 정보의 범위가 명확하게 규명되며, 해석을 용이하게 하는 역할을 할 수 있다. 코드작성의 룰이 사람에 따라 다르기 때문에, 동일한 상황에 대해서도 여러 가지 코드가 작성될 수 있다. 코드의 다양성을 배제 하더라도, PLC 코드 같이 낮은 레벨의 코드는 가독성(Readability)의 어려움을 배가시킨다. 따라서 이 같은 어려움을 해결하기 위해서는 코드의 흐름을 잡아주는 일이 무엇보다 중요한데, SOS-Net은 공정 진행 흐름을 잡아 줄 뿐 아니라, 코드로 이어지는 논리 관계를 갖고 있기 때문에, 코드 작성의 문제 및 가독성의 문제를 해결하는 방안이 될 수 있다.

5. 결 론

생산주기 단축은 기업의 이익과 직결되기 때문에, 이를 줄이기 위한 노력이 많은 부분에서 일어나고 있

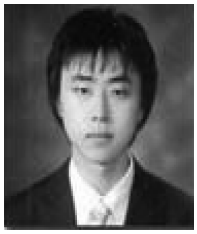
다. 상대적으로 공정이 크고 복잡한 자동차 산업의 경우 생산주기 단축을 위해 과거 공정을 변형하는 경우가 많다. 이때, 기존 공정을 분석한 후, 과거 공정을 보완, 수정하여, 새로운 차종을 생산한다. 이때, 가장 시간과 노력이 많이 들어가는 부분 중에 하나가 제어코드를 수정하는 일이다. 왜냐하면, 제어코드 특성상 해석이 어렵고, 재사용을 위한 표준화 정도가 낮기 때문이다. 최근 코드 표준화 부분을 위한 연구가 시작되고 있지만, 아직 그 수준이 미비하기 때문에, 코드 재활용 및 해석에 많은 어려움을 겪고 있다. 표준화 되지 않은 코드를 해석하는 것은 새로운 코드를 작성하는 것만큼 많은 시간이 요구된다. 또, 정확성이 보장되지 않는다는 점에서 문제를 갖고 있다. 본 논문에서 제안하는 SOS-Net은 공정 계획단계의 정보를 통일된 형식의 코드로 생성해 줌으로써, 분석 및 추가 정보 입력이 용이하도록 하는 역할을 한다. SOS-Net을 통한 공정모델링은 곧 제어정보 모델링을 의미한다. 따라서, 공정계획 초기부터, 최종 검증단계까지 SOS-Net 정보를 보전, 수정시켜 진행함으로써, 제어 정보 해석의 명확성 및 코드 작성의 표준화를 수행할 수 있다. 향후 연구 계획으로는 디바이스 기능, State를 기반으로 한 FB(Function Block)코드 생성을 연구할 것이다. 현재 LD 수준의 생성 코드는 전체 흐름을 잡아주는 것이 주 목적이다. 따라서 실제 디바이스 동작 수행부분을 FB로 생성해 줌으로써, 공정 전체 제어 코드 생성을 연구할 계획이다.

감사의 글

본 연구는 국방부 방위사업청과 국방과학연구소의 지원으로 수행되었습니다(UD080042AD). 관계자 여러분께 감사 드립니다.

참고문헌

1. Al-Ahmari, A. and Ridgway, K., “An Integrated Modeling Method to Support Manufacturing System Analysis and Design”, *Computers in Industry*, Vol. 2, No. 38, pp. 225-238, 1999.
2. Anglani, A., Grieco, A. and Pacella, M., “Tolio T. Object-oriented Modeling and Simulation of Flexible Manufacturing System: A Rule-based Procedure”, *Simulation Modeling Practice and Theory*, Vol. 2, No. 10, pp. 209-234, 2002.
3. Zeigler, B. P., “Multifaceted Modeling and Discrete Event Simulation”, Academic Press, Orland. Chuang C. P., Lan, X., Chen, J. C. 1999.
4. Jang, J., Koo, P. H. and Nof, S. Y., “Application of Design and Control Tools in a Multirobot Cell”, *Computers and Industrial Engineering*, Vol. 2, No. 32, pp. 89-100, 1997.
5. Iwata, K., Onosato, M., Teramoto, K. and Osaki, S., “A Modeling and Simulation Architecture for Virtual Manufacturing Systems”, *CIRP*, Vol. 1, No. 44, pp. 399-402, 1995.
6. Klingstam, P. and Gullander, P., “Overview of Simulation Tools for Computer-aided Production Engineering”, *Computers in Industry*, Vol. 2, No. 38, pp. 173-186, 1999.
7. Ye, L. and Lin, F., “Virtual System Simulation - A Step Beyond the Conventional Simulation”, *22nd Int. Conf. on Computer and Industrial Engineering*, pp. 304-306, 1997.
8. Manesis, S. and Akantziotis, K., “Automated Synthesis of Ladder Automation Circuits based on State-diagrams”, Automated Synthesis of Ladder Automation Circuits based on State-diagrams, Elsevier Science Ltd. Oxford, UK, UK, http://portal.acm.org/citation.cfm?id=1063492.1063494&coll=&dl, Vol. 2, No. 36, pp. 225-233, 2005.
9. Rullan, A., “Programmable Logic Controllers Versus Personal Computers for Process Control”, *Computers and Industrial Engineering*, Vol. 2, No. 33, pp. 421-424, 1997.
10. Kim, T. G., “DEVSIM++ User’s Manual”, Department of Electrical Engineering, KAIST, Korea, 1994.
11. Richardsson, J. and Fabian, M., “Modeling the Control of a Flexible Manufacturing Cell for Automatic Verification and Control Program Generation”, *Int. J. Flex. Manuf. Syst.*, Vol. 2, No. 18, pp. 191-208, 2006.
12. Bani. Younis, M. and Frey, G., “Formalization of Existing PLC Programs: A Survey”, *Proceedings of CESA 2003*, NO.S2-R-00-0239, 2003.
13. Lee, S. C., Song, I. H. and Borm, J. H., “An Accurate and Efficient Method of the spray Paint Simulation for Robot OLP”, *Transactions of the Society of CAD/CAM Engineers*, Vol. 13, No. 4, pp. 296-304, 2008.
14. Noh, S. D. and Park, Y. J., “Material Planning Information Management for Automotive General Assembly using Digital Factory”, *Transactions of the Society of CAD/CAM Engineers*, Vol. 9, No. 4, pp. 325-333, 2004.



고 민 석

2006년~현재 아주대학교 산업공학부 석사과정
2003년~2006년 아주대학교 산업정보시스템 공학부 학사
연구분야: PLC, 시뮬레이션, 이산사건 모델링

홍 상 현

2008년~현재 아주대학교 산업공학부 박사과정
현대.기아자동차 설비제어기술팀 과장
연구분야: PLC, 시뮬레이션, 이산사건 모델링

왕 지 남

1987년 3월~1992년 12월 미 Texas A&M 대학 산업공학과
1983년 3월~1985년 2월 한국과학기술원 산업공학과
1979년 3월~1983년 2월 아주대학교 공과대학 산업공학과
1984년 12월~1987년 8월 현대전자(주) 연구원(Dept. of Production Informaion Control Systems)
1992년 12월~1993년 5월 미 Texas A&M University 연구원
1993년 9월~1997년 8월 현재 아주대학교 산업정보시스템공학부 교수
2000년 9월~2001년 8월 University of Texas at Austin Visiting Professor
연구분야: 자동감시제어, 신경망, PLC, 시뮬레이션, 이산사건 모델링

박 상 철

Ph.D. (2000) in Industrial Engineering, Dept of I.E., KAIST, Korea
B.S. (1994) in Industrial Engineering, Dept of I.E., KAIST, Korea
M.S. (1996) in Industrial Engineering, Dept of I.E., KAIST, Korea
2000년 9월~2001년 12월 큐빅테크, 선임연구원
2002월 1년~2004년 2월 DaimlerChrysler ITM Dept. Research Engineer
2008년 2월~현재 아주대학교 산업정보시스템 공학부, 부교수
2005년 1월~현재 Computer-Aided Design and Applications
2006년 9월~현재 EJIT(Entrue Journal of Information Technology)
2007년 2월~2009년 1월 31일 한국CAD/CAM 학회 편집위원
연구분야: CAD/CAM, 시뮬레이션, PLC, 이산사건 모델링