Journal of the Korean Institute of Illuminating and Electrical Installation Engineers Vol. 12, No.2, pp. 54~61 May 1998

Sequential Function Chart 그래픽 언어로 記述된 공정제어 시스템에서 인터록의 실현

(Implementation of Interlock in Process Control System Described by Sequential Function Chart Graphical Language)

유정봉* · 우광준** · 허경무***
(Jeong-Bong You · Kwang-Joon Woo · Kyung-Moo Hyu)

요 약

PLC를 사용한 공정제어시스템의 설계에서 PLC표준언어중 LD언어가 가장 널리 사용되고있다. 그러나 LD언어는 데이터처리와 유지보수에 대한 단점이있다. 반면에 SFC그래픽언어는 복잡한 순차동작을 간결하게 記述할 수 있는 완벽한방법이지만,인터록조건을 記述하는데 문제점이있다. 본논문에서는 기존의 SFC 컴파일러를 사용하여 인터록을 실현하는 방법을 제시하고,실예로서 In-Line Spin Coater에 적용하여 타당성을 확인하였다.

Abstract

Ladder Diagram(LD) is the most extensively used among Programmable Logic Controller(PLC) standard languages for the design of process control system with PLC. But LD has the disadvantages for data processing and maintenance. On the other hand, there is full support for describing sequences so that complete sequential behavior can be easily broken down using a concise graphical language called Sequential Function Chart(SFC). Inspite of those characteristics, SFC is not suitable for describing interlock logic. In this paper, we propose the method for implementing interlock logic by using conventional SFC compiler and verify the effectiveness by applying proposed scheme to the In-Line Spin Coater.

1. 서 론

현대의 공정제어는 시스템의 대규모, 고도화, 복잡화에 따라 다양한 형태의 제어시스템을 요구하고 있다. 산업분야의 기능이 복잡해지고, 장비 규모가 커짐에 따라서 그들의 제어가 훨씬 복잡하게 되었으며, 각 공정들의 유기적 연결을 위해 더욱 고도화된 제어시스템이 필요하게 되었다. 이러한 제어의 필요

*정회원 : 단국대학교 전자공학과 박사과정
**정회원 : 단국대학교 전자공학과 교수
***정회원 : 단국대학교 전자공학과 조교수
접수일자 : 1997. 12. 9

성에 의해 PLC가 개발되었다. 마이크로프로세서의 발전에 의해 고도의 기능을 갖는 PLC가 출현하여, 대규모의 입출력을 처리할 수 있고, 공정간의 유기적 연결을 위한 통신 네트워크에 의해 링크 될 수 있으며, 다양한 산술 및 로직 연산기능을 가지게 되었다[1].

PLC는 공장자동화(Factory Automation, FA)의 요구에 맞춰 기능이 향상되었고 적용 범위가 확대되었으며, 특히 프로세스 제어 분야에서 중요한 역할을 하게 되었다. 산업 제어시스템에서, 소프트웨어의 질적인 향상뿐만이 아니라 개발효율을 높이고 프로그램 기술을 향상시키고자 PLC 프로그램 언어의 표준이 나오게 되었다[2]. 표준언어로는 텍스트기반의 언어로서 Instruction List (IL), Structured Text (ST)가 있고, 그래픽기반의 언어로는 Ladder Diagram (LD)과 Function Block Diagram (FBD)이 있으며, 프로그램을 구조적으로 표현가능한 Sequential Function Chart (SFC)가 있다.

IL언어는 기존의 어셈블러와 유사한 구조를 갖는 Low Level 언어이며, 뉴마닉(Mnemonic)이라고 불리웠던 언어를 표준화 한 것이다. ST언어는 Pascal과 유사한 구조를 갖는 High Level언어이고, 산업제어 분야를 위해 개발된 언어로서, 복잡한 수식계산이 필요한 시스템에서 유용하다. FBD언어는 프로그램의 기능적 요소를 블록으로 표현하고, 이들 기능 블록들의 상호 연결에 의해 제어 요소간에 정보나 데이터의 흐름이 있는 시스템에서 사용된다. LD언어는 PLC에 가장 많이 사용되는 표준언어로 릴레이 제어반의 여러 요소를 소프트 웨어로 대치시켜 접점 및 코일로 표현할 수 있게한 언어로서, 조합논리의 표현에 기반을 둔 언어이므로 조건과 인터록 논리의 記述에는 우수한 장점이 있으며, 조건과 인터록 논리에서는 다른 언어보다 장점이 있지만, 제어흐름을 나타내는 순차제어논리의 기술에는 많은 어려움이 있다. SFC언어는 1977년에 프랑스에서 개발된 GRAFCET에 근거한 국제표준언어로서, 이산제어 프로그램에서 순차 제어 논리의 기술에 적합한 강력한 그래픽언어이므로 제어의 흐름을 이해하기 쉬우며, 유지보수가 용이하고, 프로그램의 記述性이 뛰어나고, 기계장애의 진단성이 우수하다는 장점이 있다[3~6].

표 1. PLC 프로그램 언어의 비교
Table 1. Comparison of PLC's Program Language

	LD	SFC	IL
조건/인터록	○	×	×
순차제어	△	○	×
데이터처리	×	△	○
유지보수	×	○	×

○ : 우수함 △ : 보통임 X : 나쁨

표 1에 PLC에서 사용되는 대표적인 언어를 비교하였다[3]. 프로세스 제어시스템 설계에서 LD언어를 사용하면 순차 제어 논리의 記述에 어려움이 있고, 데이터 처리 및 유지보수의 단점이 있으며, 반면, SFC언어를 사용하면 조건과 인터록 記述에 단점이 있으므로, 어느 하나의 언어만을 사용하여 시스템을 설계하면 취약한 부분에 대한 처리가 어렵게 된다. 따라서 프로세스 제어 시스템 설계시, 순차 제어 논리의 記述에는 SFC언어를 사용하고, 조건과 인터록 논리의 記述에는 LD언어를 사용한다면, 복잡한 프로세스 제어 시스템 설계를 효율적으로 할 수 있을 것이다.

본 논문에서는, 프로세스 제어 시스템 설계시 순차 제어 논리의 기술에 탁월한 장점을 갖는 SFC언어를 사용하고 조건과 인터록 논리를 SFC언어중 매크로 블록으로 기술하여 처리하였으며, 매크로 블록의 내용인 조건과 인터록 논리는 LD언어로 기술하여 처리함으로써 순차 및 조합 제어 논리를 용이하게 기술할 수 있도록 하였고, 프로그램 메모리 용량을 대폭 줄일 수 있게 하였다. 또한 제안된 방법을 25inch TFT-LCD 제조용 In-Line Spin Coater에 적용하여 SFC언어에 의한 인터록 논리의 記述 및 실현이 가능함을 확인하였고, PLC를 사용한 공장자동화 설비 및 건물자동화 설비에 효율적으로 적용할 수 있을 것으로 기대된다.

2. 인터록의 記述 및 실현

2.1 LD언어에서의 記述 및 실현

인터록이란, 하나의 동작을 수행하는데 있어서의 잠금장치이며, 동작을 수행하기 위한 기본적인 조건이 되기도 한다. LD언어에서 인터록 논리가 記述될 때에는, 인터록 블록은 전체 스캔(Scan)프로그램의 앞부분 또는 뒷부분에 일정한 블록에 설정되어, 일련의 동작들에 대한 조건으로 사용된다. 그림 1의 LD 프로그램에서 P0는 인터록 블록으로서 이 블록 내의 접점을 스캔하여 각각의 출력에 영향을 미치게 된다.

인터록 논리식은

$$L431 = X40$$
$$L432 = X49 \cdot X4A \cdot X4B$$
$$L433 = X4D \cdot X4E \quad (1)$$

이다. 한편, 인터록 논리가 반영된 시퀀스 프로그램의 상태방정식은

$$M001 = \{(M000 \cdot X07) + (M001 \cdot \overline{M002})\} \cdot L431$$
$$M002 = \{(M001 \cdot X08) + (M002 \cdot \overline{M003})\} \cdot L432$$
$$M003 = \{(M002 \cdot X09) + (M003 \cdot \overline{M004})\} \cdot L433$$
$$M004 = \{(M003 \cdot X0A) + (M004 \cdot \overline{M005})\} \cdot L432 \quad (2)$$

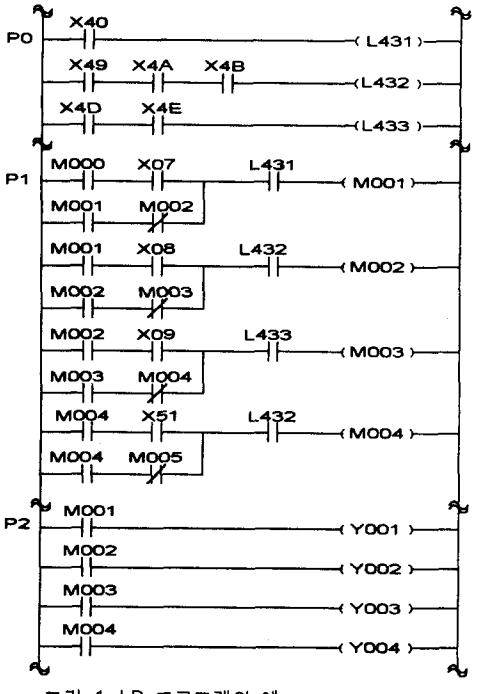


그림 1. LD 프로그램의 예
Fig. 1. LD Program to Process Example

이다. 또한, 인터록 논리가 반영된 출력 프로그램의 출력 논리식은

$$Y001 = M001$$
$$Y002 = M002$$
$$Y003 = M003$$
$$Y004 = M004 \quad (3)$$

이다.

이상의 예에서 보인 바와 같이 LD언어에서는 순서 제어 논리의 記述은 명확하지 않으나, 인터록 논리와같은 조합논리를 명확하게 記述하는 것이 가능하다.

2.2 SFC언어에서의 記述 및 실현

1) 기존의 방법

① 기존의 방법 1

그림 2는 그림 1의 LD프로그램을 SFC언어로 記述한 것으로, 각각의 스텝에 해당하는 인터록 논리들을 모두 부가하는 방법이다. 이와같은 방법을 사용하면, 記述상 불편함이 있고, 또한 많은 용량의 프로그램 메모리가 필요하게 된다.

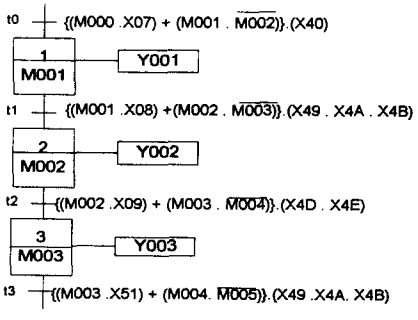


그림 2. SFC 프로그램의 예
Fig. 2. SFC Program to Process Example

② 기존의 방법 2

또다른 방법으로는 그림 3과 같이 1번 스텝을 인터록 논리 스텝으로 설정하고 1번 스텝 내부 전체에 대해 SET(그림 3에서 1번 스텝의 SE)[2],[9] 명령을 사용하는 방법이다. t1 천이조건이 만족되어 2번 스텝이 활성화되면 1번 스텝은 비활성 상태가 되어야 하지만, SET 명령 때문에 활성상태로 유지된다. t0 천이조건을 만족하여, 1번 스텝이 활성화 될

때 1번 스텝의 접점들을 스캔하여 L431, L432, L433을 셋 시키고 t1 천이조건이 만족되면 2번 스텝을 활성상태로 만들게 된다. 이때 1번 스텝의 접점들은 다시 스캔하지 않는다. 따라서 2번 이후의 스텝이 활성으로 될 때 1번 스텝의 인터록 논리가 바뀌어도 인터록 논리식 L431, L432, L433에 영향을 주지 못하여 인터록 논리가 성립하지 못하게 된다. 즉 이와같은 방법은 인터록의 논리 기능 보다는 초기 조건으로서 처음 스캔한 조건만 가능하고, 이후에 바뀌는 조건에 대해서는 의미를 갖지 못하게 된다.

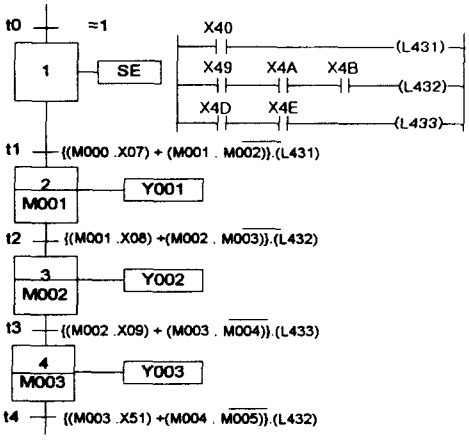


그림 3. SET 명령을 사용한 SFC 프로그램의 예
Fig. 3. SFC Program using Set Instruction to Process Example

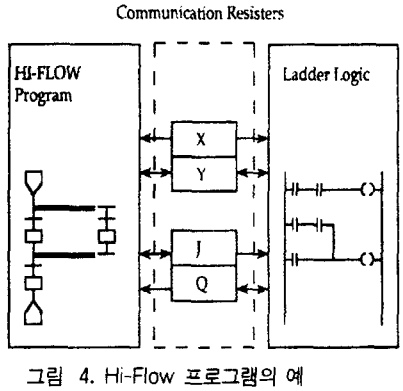


그림 4. Hi-Flow 프로그램의 예
Fig. 4. Example of Hi-flow Program

2) Hi-flow 프로그램을 이용한 방법

Takao Kokubo[7]는 Hi-flow 프로그램에서 인터록 논리들을 LD언어로 프로그램 한후 인터록 논리들을 통신으로 접수하여 사용하였다. 그림 4와같이 통신 레지스터를 경유하여 Hi-Flow프로그램에서 인터록 논리들을 접수하게 된다. 이와같은 방법은 통신 레지스터 처리시간이 길어질 뿐만 아니라, 통신 레지스터 운용 프로그램을 작성해야 하기 때문에 복잡한 프로세스시스템 설계에는 적합하지 않다.

3. SFC언어에서 제안된 인터록 논리의 記述 및 실현

3.1 SFC언어에서 매크로 블록의 표현

매크로 스텝의 목적은 복잡한 시스템의 기술을 용이하게 하는 것이다. 매크로 스텝은 한 스텝을 분리해서 세부적으로 SFC언어의 그래픽 표현을 명확히 할 수 있게 한다. 매크로 스텝의 개념은 그림 5에서 설명된다[8].

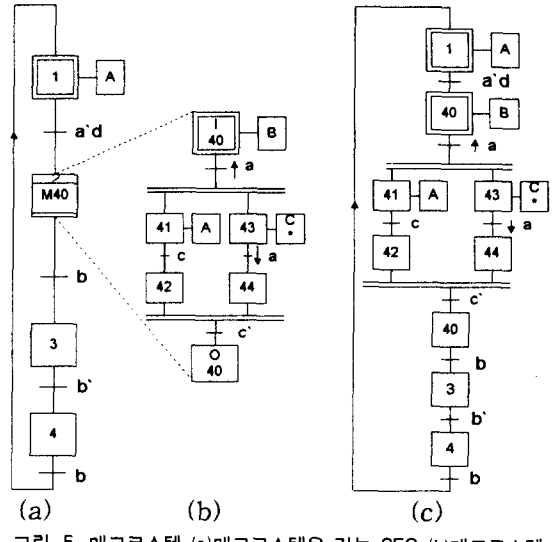


그림 5. 매크로스텝 (a)매크로스텝을 갖는 SFC (b)매크로스텝의 확장 (c)매크로스텝이 없는 등가 SFC
Fig. 5. Macrostep (a)SFC with Macro-step (b)Macrostep Expansion (c)Equivalent SFC without Macro-step

2/M40으로 주어진 매크로 스텝은 그림 5(a)에서 표현된다. 그림 5(b)는 M40에 대응하는 확장이다. 이 매크로 스텝의 확장이 매크로 스텝 2/M40에 대치되면, 그림 5(c)의 SFC가 얻어진다.

매크로 스텝과 그의 확장은 다음의 규칙을 만족한다.

① 매크로 스텝 확장은 하나의 입력 스텝(I)와 출력 스텝(O)를 갖는다.

② 매크로 스텝의 이전 천이조건의 모든 점화는 그 확장의 입력 스텝을 활성화시킨다.

③ 매크로 스텝의 확장의 출력 스텝은, 그 매크로 스텝을 포함하는 SFC의 구조에 따라, 이후천이조건을 가능하게 한다.

3.2 매크로 스텝을 이용한 제안된 방법

매크로 스텝을 두가지로 정의할 수 있다[9].

정의 1 :

매크로 스텝은 매크로 스텝의 확장이 END를 확인한후 다음스텝으로 진행 가능하다.

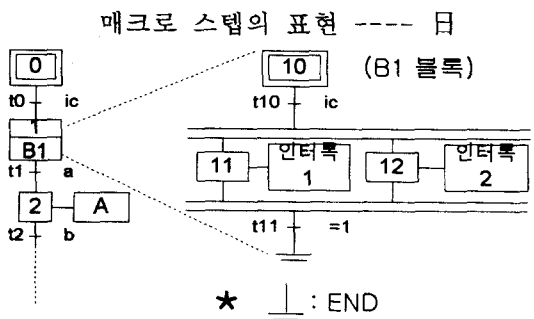


그림 6. END 를 검사하는 매크로스텝
Fig. 6. Macrostep that checks END

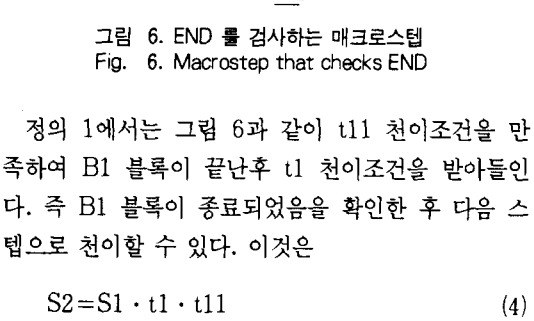
정의 1에서는 그림 6과 같이 t11 천이조건을 만족하여 B1 블록이 끝난후 t1 천이조건을 받아들인다. 즉 B1 블록이 종료되었음을 확인한 후 다음 스텝으로 천이할 수 있다. 이것은

$$S2=S1 \cdot t1 \cdot t11 \quad (4)$$

즉, 식 (4)와 같이 다음 스텝 S2는 전 스텝 S1과 B1블록의 마지막 천이조건 t11, 그리고 천이조건 t1을 동시에 만족할때에 활성화된다.

정의 2 :

매크로 스텝은 매크로 스텝의 확장이 END 를 확인하지 않아도 다음스텝으로 진행 가능하다.



정의 2에서는 그림 7과 같이 t0 천이조건을 만족하면 매크로 스텝 1이 활성으로 되어 B1블록이 기동하게 된다. 이때 B1블록의 종료를 확인하지 않아도 t1조건을 만족하면 2번 스텝이 활성으로 되지만, B1 블록은 무한 루프로 된다.

$$S2=S1 \cdot t1 \quad (5)$$

즉, 식 (5)와 같이 前스텝 S1과 천이조건 t1만으로 다음스텝으로 이동하게된다.

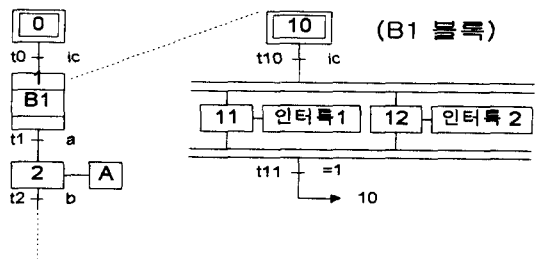


그림 7. END를 검사하지 않는 매크로 스텝
Fig. 7. Macrostep that doesn't check END

일반적으로 SFC언어에서 前스텝은 비활성상태 또는 강제 셋상태(SET 명령을 사용할 때)로 된다. 그러나 그림 7에서 나타낸 것과 같이 B1매크로블록을 강제로 무한 루프로 만들고, 매크로 스텝의 다음 천이 조건(t1)을 만족 시키면, B1블록 내부에 있는 접점의 논리가 바뀌어도 이 접점을 사용하는 그 이후의 스텝에 이 접점의 논리가 그대로 반영된다.

본 논문에서는, 정의 2의 매크로 스텝을 사용하여, SFC언어에서 인터록 블록을 기술하는 방법을 제시한다. 그림 7과 같이 1번 매크로 스텝을 인터록 블록으로 설정하고, B1블록내의 각 스텝, 즉 11번 스텝과 12번 스텝에 각각의 인터록 논리들을 설정하면 SFC언어에서 가장 큰 단점인 조건과 인터록 논리를 명확히 記述하고 실현할 수 있게 된다.

4. 적용 예

4.1 시스템개요

본 연구에 사용된 시스템은 그림 8과 같이 입구대기부, 스핀 코터부, 감압건조부, 단면세정부, 출구대

기부 그리고 셔틀부의 6개 유닛(Unit)으로 이루어지는 In-Line Spin Coater이다. In-Line Spin Coater는 TFT-LCD 생산공정중의 일부 장비로서, 유리기판에 포토레지스트의 도포액을 슬릿 노즐을 통해 기판 표면에 얇게 도포한후, 회전 도포하는 장비이다. 이 시스템은 셔틀에 의한 리니어 이송방법을 채택하고 있으며, 스핀코터부의 진공 척(Chuck)과 단면세정부의 진공 척의 온도차이로 인한 유리기판 이면에 흔적이 발생하는 것을 방지하기 위하여 스핀코터부와 단면세정부 사이에 감압건조부를 도입하였다.

위의 기능중에서 스핀코터부가 가장 핵심부분으로 포토레지스트가 도포된 기판을 회전시켜 포토레지스트를 기판 전체에 균일하게 분산도포시켜 주어야 한다.

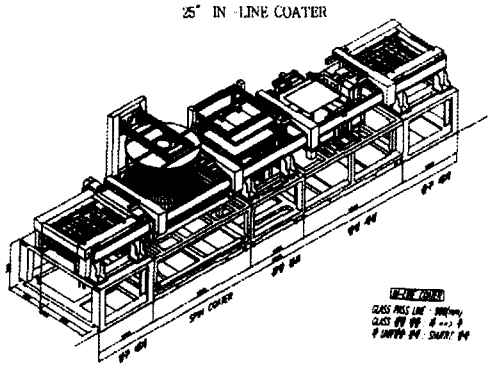


그림 8. In-Line Spin Coater 구성도
Fig. 8. Configuration of In-Line Spin Coater

4.2 알고리즘 설계

각 6개부의 개별 알고리즘을 설계한후 셔틀에 동기시킨 하나의 전체 알고리즘으로 설계하였다. 기구부의 위치를 판별하는 기구부(Mechanism)위치 판정부와 전원을 투입한후 원점복귀를 하여 원점에서부터 출발하는 원점복귀부, 그리고 인터록 논리 부분이 공통 프로그램으로 전체 알고리즘의 맨 앞부분에 위치하게 된다. 인터록의 기능을 설명하기 위해 간단히 입구대기부의 알고리즘만 나타내었고, 이 알고리즘이 그림 9와 같다. 그림 9의 알고리즘에서 3번 스텝에 인터록 블록을 설정하였고, 이 인터록 블록은 3장 정의 2에서 정의된 END를 확인하지 않는 매크로 블록을 사용하였으며, 그림 9의 SFC에서 활성 스텝이 어느곳에 있든 이 인터록 블록은 무한 루프로 되어있다. 그림 10에 입구대기부의 인터록 블록을 나타낸다. 그림 10에서 t1 천이조건은 항상 만족되어 0번 스텝으로 점프하게되고, 다시 1번, 2번, 3번 스텝이 동시에 활성화되어 모든 인터록 논리들을 스캔하고, 이 인터록 논리들은 그림 9의 SFC에서 모든 스텝과 천이조건에 적용된다.

본 논문에서는, 알고리즘설계시 순차제어 논리의 記述에 탁월한 장점을 갖는 SFC 언어를 허용하고 조건과 인터록 논리를 SFC 언어중 매크로 블록으로 記述하여 처리 하였으며, 매크로 블록의 내용인 조건과 인터록 논리는 LD언어로 기술하여 처리함으로써 순차 및 조합 제어 논리를 용이하게 기술할 수 있도록 하였고, 프로그램 메모리 용량을 줄일 수 있게 하였다.

4.3 System에의 적용 결과

In-line spin coater를 제어하는 PLC는 미쓰비시 기종의 Q3A CPU를 사용하였으며, 편집 프로그램은 DOS-Version의 GPPQ를 사용하였다[9].

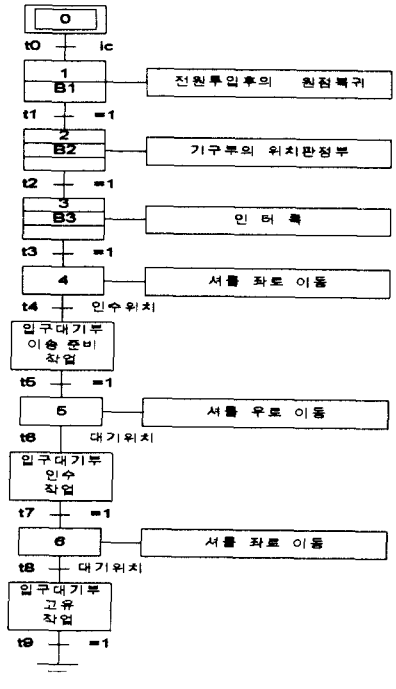


그림 9. 입구대기부 알고리즘
Fig. 9. Algorithm of Input Unit

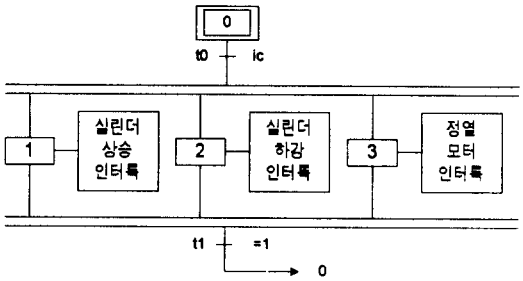


그림 10. 입구대기부의 인터록 블록 B1
Fig. 10. Interlock Block B1 of Input Unit

GPPQ 프로그램은 전체 알고리즘을 SFC언어로 작성하고, SFC의 각 스텝과 천이조건을 LD언어로 프로그램 할 수 있는 것이 특징이다. 그림 11은 기존의 방법 1의 결과를 나타내고, 그림 12는 기존의 방법 2의 결과를 나타낸다. 그림 11에서는 각 스텝과 각 천이조건에 인터록 조건들을 항상 부가 해야하기 때문에 메모리 용량이 커야한다. 이 결과에서는 1343 스텝의 메모리 용량을 나타내고 있으며, 그림 12에서는 1163 스텝의 메모리 용량을 나타내고 있어 기존의 방법 1이 큰 용량의 메모리를 필요로 하게 됨을 알 수 있다. 그러나 그림 12 는, 현재 활성 스텝이 3번 스텝이며, SET명령을 사용한 인터록 스텝인 1번 스텝에서는 L432 접점이 ON되어 있는 것을 알 수 있다. 이것은 1번 스텝이 활성으로되어 한번 셋 되면, 인터록 조건이 변경되어도 그 이후의 활성 스텝에 영향을 주지 못하게된다는 단점이 있다. 그림 13은 본 논문에서 구현하고자 하는 입구대기부의 알고리즘을 적용한결과를 나타내었고, 그림 14에 입구대기부에 대한 인터록 논리들을 결합한 인터록 블록을 적용한 결과를 나타내었다. 그림 13과 그림 14에서처럼 블록의 종료를 확인하지 않는 매크로 블록(정의 2)을 사용하여 인터록 논리를 記述하였다. 그림 13에서는 2번 스텝, 즉 B2 블록이 활성으로 되어있고, 그림 14는 그림 13의 3번 매크로 스텝 즉, B3 블록을 나타내주며 현재 B3 블록의 1번, 2번, 3번 스텝이 활성으로 되어있는 것을 알 수 있다. 이것은 2번 매크로 스텝인 B2 블록과 3번 매크로 스텝인 B3 블록이 동시에 진행이 되고 있음을 알 수 있다. 즉 B3 블록은 인터록 블록으로서 시퀀스 프로그램의 어느 스텝이 활성으로 되어있든 관계없이 계속해서 인터록 논리들을 스캔하여 시퀀스 프로그램의 접점에영향을 주게 되는 것이다.

결국, 2.2절의 정의 2에서 정의한 매크로 블록을 사용하여 인터록 논리를 記述하면 메모리 용량도 줄일 수 있을 뿐만 아니라, SFC언어에서의 조건/인터록 논리의 단점을 극복할 수 있음을 알 수 있다.

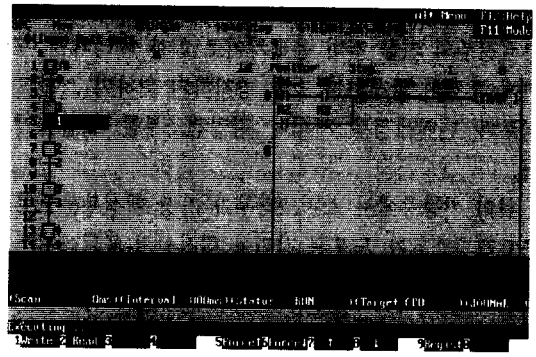


그림 11. 기존의 방법 1의 적용결과
Fig. 11. Results applied by Conventional Method 1

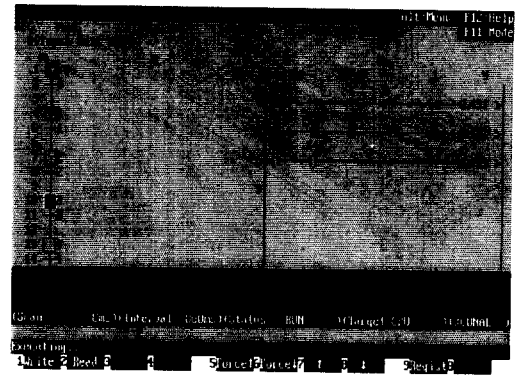


그림 12. 기존의 방법 2의 적용 결과
Fig. 12. Results applied by Conventional Method 2

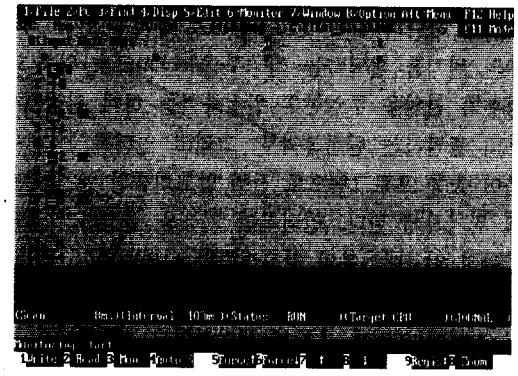


그림 13. 입구대기부의 적용결과
Fig. 13. Results appiled to Input Unit

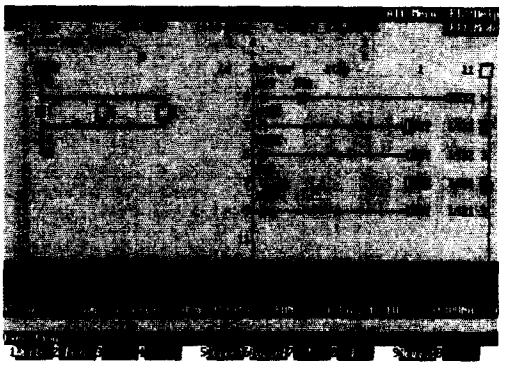


그림 14. 입구대기부의 인터록 블록의 적용결과
Fig. 14. Results applying Interlock Block to Input Unit

5. 결 론

PLC를 사용한 프로세스 제어 시스템 설계시 지금까지는 대부분 LD 언어를 사용하여 왔다. 그러나 LD언어는 순차제어 논리의 記述에 어려움이 있고, 데이터 처리 및 유지보수의 단점이 있어서, 최근들어 이러한 단점을 보완하고자 SFC언어를 PLC에 적용하는 연구가 활발히 진행되고 있다[3, 7~9]. SFC언어를 PLC에 적용하기 위해서는 SFC언어의 단점인 조건과 인터록 記述에 대한 해결이 중요한 과제가 되어왔다. 본논문에서는 SFC언어에서의 인터록 논리에 대한 記述 및 실현 방법을 제시하였고, 적용 예를 통해 타당성을 확인함으로써 SFC언어를 사용하여 프로세스 제어 시스템 설계를 할 때 인터록 논리의 記述에 대한 단점을 해결하였다.

앞으로 SFC언어를 사용한 복잡한 프로세스 제어 시스템설계에서 순차 및 조합 제어 논리를 효율적이고 용이하게 記述하고 실현할 수 있을 것으로 기대된다.

본 연구는 단국대학교 교내 연구비 지원에 의하여 수행되었음

참 고 문 헌

[1] IAN G. Warnock "Programmable Cont-rollers-Operation and Application" Prantice Hall, 1992.

[2] R.W.Lewis "Programming Industrial Con-trol Systems Using IEC1131-3", The Insti-tution of Electrical Engineers, 1992.

[3] Masaharu Oku, Takao Kokubu, Shigeru Masuda, and Kenzo Kamiyama "Application of the Encapsulated Actuator Model to the Sequential Cotrol Machines", Short Papers IEEE/ASME Transactions on Mechatronics, Vol 1, No 4, pp. 290~294, Dec 1996.

[4] T.Kouthon, M. A. Peraldiabd J. D. Deco-tignie "Distributing PLC Control", Inter-national Conf. on IEC, IEEE 21'st Vol 2, pp. 1614~1619, 1995.

[5] II Moon "Modeling Programmable Logic Controllers for Logic Verification", IEEE Control Systems, Vol 14, No 2, pp. 53~59, April 1994.

[6] Takao Kokubo, Kenzo Kamiyama, Ma-saharu Oku and Hitoshi Saito, "Applica-tion of P.werful SFC Language (Hi-flow) To FA Systems In a TireIndustry", 16th annual Conference of IEEE IES, 1990.

[7] Rene David, "GRAFCET : A Powerful Tool for Specification of Logic Controllers", IEEE Trans on ContolSystems Techno-logy, Vol 3, No 3, pp. 253~268, 1995.

[8] "Programming Manual (SFC편)", Mitsubishi, QnA series, 1995.

◇ 著 者 紹 介 ◇

유 정 봉(庾正鳳)

1964년 3월 5일생. 1988년 단국대 전자공학과 졸. 1990년 동대학원졸(석사). 1990년~1993년 (주)신도리코. 1996년 12월 동대학원 박사과정 수료.

우 광 준(禹廣俊)

1946년 11월 8일생. 1974년 한양대 전자공학과 졸. 1977년 동대학원졸(석사). 1980년 Univ.de Stras-bourg 1 제어계측 공학 D.E.A. 1993년 Institut National Polytechnique de grenoble 졸 (박사). 현재 단국대학교 전자공학과 교수.

허 경 무(許慶茂)

1956년 9월1일생. 1979년 2월 서울대 공과대학 전자공학과 졸. 1981년 8월 한국과학기술원 전기 및 전자공학과 졸(공학석사). 1989년 8월 한국과학기술원 박사학위 취득(공학박사). 1981년 9월~1985년 2월 대우중공업(주) 대리. 1989년 9월~1993년 2월 대우중공업(주) 책임연구원. 1993년 3월~현재 단국대 공과대학 전자공학과 조교수.