공장자동화에 있어서 프로그램식 제어기의 역할

(1)

강 영 채

전 인천대학교 교수

머리말

현대 산업사회의 인력부족과 악화되는 기업경쟁에 대비하기 위하여 산업체에서는 에너지절약, 인건비 절감 및 정보 시스템화 등 자동화의 필요성이 절실하여 자동화 투자가 매년 증가하고 있다.

넓은 의미의 공장자동화(FA)란 것은 수주에서 물품 출하까지 일체의 생산활동, 즉 설계, 가공/처리, 조립, 시험/검사, 반송/보관 및 생산관리/제어 등의 제기능을 효율적, 유기적으로 결합시키는 시스템 기술을 의미하며 좁은 의미에서의 FA란 제품을 만드는 실제에 있어서 생산공정 또는 계측제어의 자동화 혹은 설계자동화의 여러 기능간의 유기적 결합이 없는 상태로의 국부적인 자동화를 의미한다.

전자의 경우 FA의 궁극적인 목표는 무인화 공장으로서 여기에는 상당한 투자가 요구되는 반면에 후자는 기존설비를 바탕으로 하여 단순한 기능의 자동화에서부터 출발하므로 소규모의 초기 자본으로 시작할 수 있는 특징이 있다.

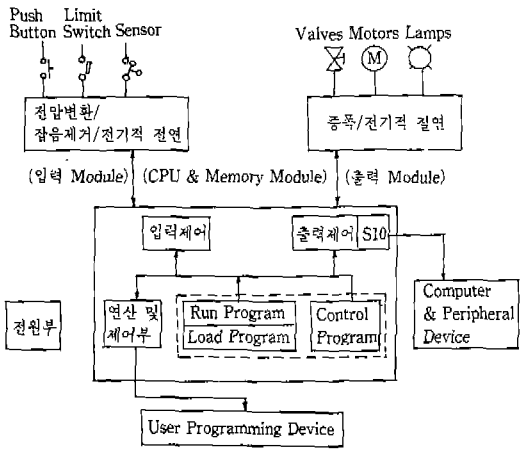
본고에서는 자동화의 필수불가결한 존재로 떠오른 PC(Programmable Controller)에 대하여 주로 기술하고자 한다.

1. 프로그래머블 컨트롤러(PC)의 구성

일반적으로 사용되고 있는 제어를 크게 나누면 피드백제어와 시퀀스제어가 되는데 시퀀스제어(Sequence Control)는 제어 대상의 움직임을 미리 정해 놓고 이에 따라 일방적으로 명령을 해 나가는 순서제어와 조건제어로 분류되어 산업계 현장의 요구를 실현시켜 온 것이다.

이러한 요구에 의해 등장한 시퀀스는 시퀀스제어를 위해 사용된 릴레이회로에 공정 변화에 따른 유용성을 향상시켰으며 초기에는 로터리 캠방식에 의한 것과 핀보드방식에 의한 것으로 구현되었다.

이것은 설계상의 문제, 시운전 조정상의 문제, 설치 보수면의 문제, 기능상의 문제 등이 대두되어 시퀀스제어를 소프트웨어로 처리하여 고도의 기능을 추가하는 필요성을 고려하게 되었다.



<그림 1> 프로그래머블 컨트롤러(PC)의 블록도

즉 릴레이회로의 로직(Logic)을 조립하고 있던 배선을 프로그램으로 대체하는 메모리가 필요하게 되었다(그림 1 참조).

또한 마이크로프로세서와 마이크로컴퓨터의 보급에 따라서 소위 축적 프로그램방식(Stored Program Method)에 의한 시퀀서(Sequencer)로 발전, 1978년 NEMA Standard 1-28에 의해서 정식으로 PC(Programmable Controller)로 명명하게 되었다.

이러한 PC는 제너럴모터스사의 자동차 조립라인의 릴레이반 대신에 Gould사에서 Micro-84 PC를 제시한 것이 효시였으며 그 후 아날로그 신호처리, PID 제어기능 뿐만 아니라 다수의 PC와 주 컴퓨터와의 인터페이스를 통한 큰 규모의 플랜트 컨트롤러까지 확장되어 계속 발전해 왔다.

한편 PC를 6개의 모듈로 나눌 수 있는데 어떤 모듈은 옵션으로 제공되기도 한다. 6개의 모듈을 보면

(가) 이용자의 프로그램이 용이하도록 한 프로그래밍 장치

(나) 센서와 제어대상과의 인터페이스를 위한 입·출력 모듈

(다) 상용 전원을 공급받아 PC 내부에서 사용하는 전원으로 변환해 주는 전원부 및 이상시 정전복구 기능을 위한 파워 모듈

(라) CPU, 메모리 및 타이머/카운터 등의 PC 중심 모듈

(마) PC 주변의 주변장치 모듈 및 네트워크 모듈

(바) PC에 내장되어 있는 모니터 및 컨트롤 소프트웨어 모듈

등을 들 수 있다.

1·1 프로그래머블 컨트롤러의 하드웨어

초기의 프로그래머블 컨트롤러는 On/Off 타입의 제어(디스크리트)를 필요로 하는데 사용되었으나 1970년대 중반에서 후반에 걸쳐 PC에 매우 큰 유연성을 추가시킴과 동시에 보다 큰 기억용량과 원격 입출력 기능, 아날로그, 위치제어, 통신 네트워크 및 소프트웨어가 보강되었다. 다음에 각각에 대하여 약술한다.

(1) 중앙연산처리장치

이 부분은 프로세서와 시스템 메모리로 나뉘어지며 마이크로프로세서의 발달로 통신기능, 다기능 등의 높은 레벨 수행이 가능한 시스템으로 되었으며 타이머/카운터 등의 기능을 소프트웨어적으로 처리한다.

이들은 프로그래밍 디바이스 또는 주변장치와의 통신, 필드간의 모니터링, 시스템 진단, 프로세스제어 및 다중처리 등의 기능을 수행한다.

(2) 메모리 유닛

프로그래머블 컨트롤러에서 일반적인 메모리 시스템은 실행, 스크래치 패드(Scratch Pad), 응용 메모리 및 데이터 테이블로 분류되며 실행 프로그램은 시스템 자체를 수행하는 시스템 모니터와 주변장치와의 통신 프로그램이 들어 있다.

스크래치 패드(자료가 임시로 저장되는 곳)는

CPU의 모니터 수행이나 제어 프로그램 수행중 생기는 일시적인 데이터를 보관하며 응용 메모리는 이용자에 의해서 프로그램 명령어가 저장되는 영역과 제어 프로그램이 들어있는 영역이다.

데이터 테이블은 제어 프로그램과 관련된 데이터를 저장한다. 예를 들면 타이머/카운터값, 제어 프로그램의 상수, 변수값 등이 저장되며 이것을 메모리맵(Memory Map)으로 나누면 시스템 메모리와 응용 메모리로 나눌 수 있다.

(3) I/O 유닛

입출력부는 주로 현장에서의 공정테스트, 센서 등의 입력들과 밸브, 모터 등의 작동기로서의 출력을 의미한다. 실질적으로 이러한 입·출력의 연결로서 프로그래머블 컨트롤러를 현장용 컴퓨터라고도 부르며 아날로그 입력/출력, 디지털 입력/출력, 스페셜 입력/출력 및 원격 입력/출력의 4가지로 분류한다.

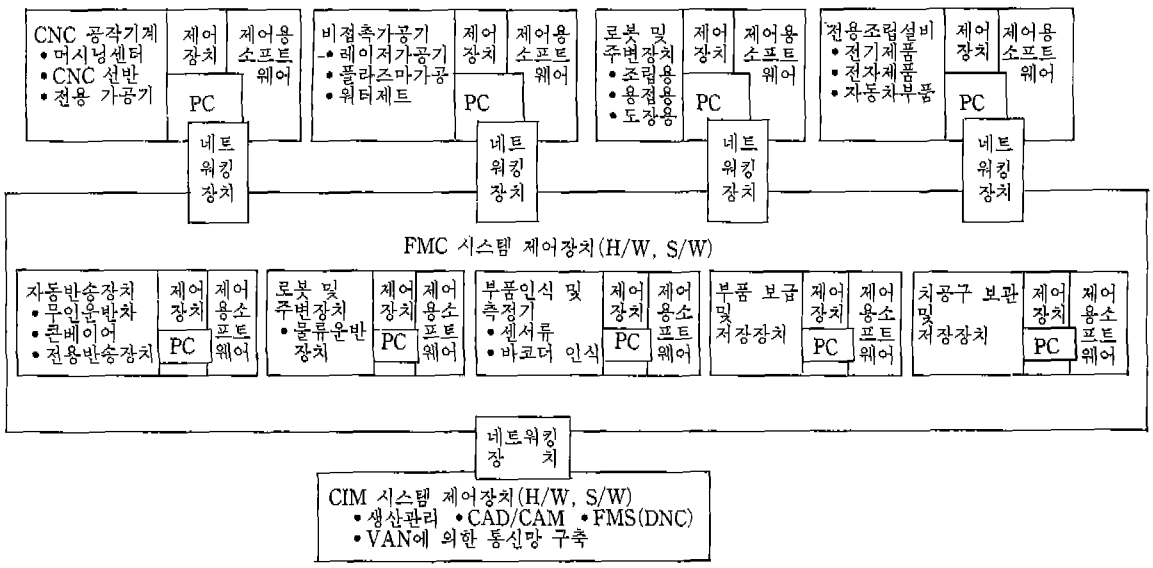
이들 각각의 입출력부 모듈에서는 특히 입·출력 점수(點數)를 구하는 것이 중요하며 실질적으로 아날로그 점수와 디지털 점수를 최대로 16점, 128점들로서 구성되어 있다.

또한 이 입·출력부에서는 각각의 점수, 입출력 형태와 정격 등이 고려되고 있으며 입출력부의 신호조정회로 및 입출력 상태를 점검하는 표시장치 및 입출력 점수의 증설을 고려하여 여유점수를 주고 있다.

이외에 프로그래머블 컨트롤러의 주변장치로서 프로그래밍 디바이스(프로그램 로더, 프로그래머 등) 파워부 및 네트워크를 위한 포트들로 구성되어 있으며 외부와의 접속시 EIA RS232C, RS422, 4~20㎃ 전류 루프가 사용된다(그림 2 참조).

1·2 프로그래머블 컨트롤러의 소프트웨어

PC에서의 시스템 소프트웨어라고 할 때 일반적인 운영체제 또는 모니터 프로그램에서 특수용도의 운영체제를 의미한다. 즉 PC에서 적합한 제어분야의 소프트웨어를 의미하며 ROM에 내장시켜



<그림 2> 하드웨어 측면에서의 공장자동화의 범위

마이크로 프로세서와 함께 제공한다.

이로 인하여 자체적으로 개발할 필요없이 응용 프로그램의 개발에 전념할 수 있도록 하고 ROM에 내장된 운영체제를 사용함으로써 소프트웨어의 신뢰성 제고와 시스템 실행속도를 단축시킬 수 있는 장점이 있다.

(1) PC의 일반적인 플로차트형

소프트웨어의 전체적인 구조는 PC를 초기화한 후 모니터 프로그램에서 이용자 프로그램을 이용자 프로그램영역에 입력하는 일과 이용자가 원하는 내용을 화면에 나타나게 하는 일 및 새로운 이용자 프로그램 입력을 쉽게 해주고 이용자가 원할 때 입력부의 신호를 받아들임으로써 실행 프로그램(제어 프로그램)을 수행하도록 한다.

또한 갑작스런 정전에 대하여 시스템을 정전전의 상태로 복구시키는데 필요한 정전복구 프로그램 즉 초기화 프로그램, 모니터 프로그램 및 실행 프로그램(제어 프로그램)의 세 부분으로 나누어져 있다.

(2) PC의 프로그램언어

PC에서 주로 사용되는 프로그램언어는 기본언어인 릴레이 래더 도형(Relay Ladder Diagram), 불 니모닉(Boolean Mnemonics) 및 고급언어인 기능블록(Function Block), 컴퓨터식 언어로 나눌 수 있다(그림 3 , 4 , 5 참조).

실질적으로 이상 4가지 언어를 복합적으로 사

명령어 심벌	기 능
TXF	입력이 OFF에서 테스트 플래그(Flag)를 세트
TXN	입력이 ON에서 테스트 플래그를 세트
TYF	출력이 OFF에서 테스트 플래그를 세트
TYN	출력이 ON에서 테스트 플래그를 세트
SYF	출력을 OFF로 한다.
SYN	출력을 ON으로 한다.
CLR	전출력을 OFF로 한다(클리어 한다).
JFF	테스트 플래그 OFF에서 점프(테스트 플래그는 OFF로 한다)
JFN	테스트 플래그 ON에서 점프(테스트 플래그는 ON으로 한다)
JMP	무조건 점프
SKP	다음의 명령을 건너뛴다.
NOP	무효명령
JMS	서브 루틴에 점프
JMR	서브 루틴에서 리턴
TXD	입력상태를 출력, 레지스터에 표시한다.
TYD	출력상태를 출력, 레지스터에 표시한다.
TRM	메모리 내용을 출력, 레지스터에 운송한다.

<그림 3 > 논리 플로방식의 명령예

명 령		내 용
LDA	X	$X \rightarrow Acc$
LDAC	X	$\bar{X} \rightarrow Acc$
AND	X	$X \cdot Acc \rightarrow Acc$
ANDC	X	$\bar{X} \cdot Acc \rightarrow Acc$
OR	X	$X+Acc \rightarrow Acc$
ORC	X	$\bar{X}+Acc \rightarrow Acc$
STO	X	$Acc \rightarrow X$

<그림 4 > 논리연산방식의 명령예

명령어 심벌	기 능
⊣⊢	a 접점
⊣/⊢	b 접점
⊣⊢	병렬 a 접점
⊣/⊢	병렬 b 접점
A	AND 요소
OR	OR 요소
OR / A	AND 연산의 결과와 전의 로직의 OR 연산
A / OR	OR 연산의 결과와 전의 로직의 AND 연산

<그림 5 > 도형처리용 명령어 심벌의 예

용하고 있으며 기본언어는 계전기식 순차제어에 필요한 순차제어, 계시 및 계수 제어와 논리회로 구성 등을 프로그램하는데 주로 사용한다.

고급언어는 아날로그제어, 데이터처리 및 보고 등 기본언어로는 수행하기 힘든 복잡한 기능을 프로그램하는데 사용된다.

1·3 프로그래머블 컨트롤러의 네트워크 기능

PC가 공정제어 시스템으로 분산제어 등 범위가 넓어짐으로써 PC에도 다음과 같은 기능이 부여되고 있다.

(가) 원격입출 기능 : 연산부와 최하위에 있는 입·출력간을 통신케이블에 의하여 데이터 및 신호를 주고 받는 기능

(나) PC LAN 기능 : 프로그래머블 컨트롤러(PC)의 CPU와 CPU간의 통신기능

(다) 인터페이스 기능 : PC와 컴퓨터와의 통신기능

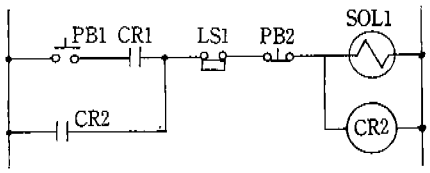
(라) 데이터 접속기능 : PC와 컴퓨터와의 사이에 다중통신 할 수 있는 기능

(마) 맵(Map) 기능 : 맵상에서의 통신

2. 프로그래머블 컨트롤러(PC)의 일반적인 동작개요

여기서는 릴레이 래더 다이어그램(Relay Ladder Diagram) 방식의 시퀀스 컨트롤러를 예로 들어서 목적하는 제어 내용이 어떻게 프로그램에 변환되고 그 프로그램을 시퀀스 컨트롤러가 어떻게 실행하고 그 결과 어떠한 기능을 발휘하는가에 대하여 설명한다.

그림 6은 모델 릴레이 시퀀스회로를 나타내고 있으며 그림 7은 시퀀스 컨트롤러의 내부구성을 나타내고 있다. 각부의 기능은 다음과 같다.



PB1 : 시동버튼 PB2 : 정지버튼
CR1 : 시동조건 LS1 : 정지조건 리미트스위치
SOL1 : 부하 솔레노이드 CR2 : 보조 릴레이

<그림 6 > 모델 시퀀스회로

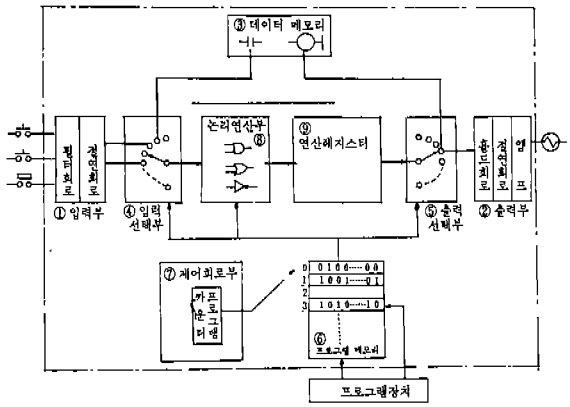
(1) 입력부

외부의 強電 입력을 시퀀스 컨트롤러 내부의 IC 신호레벨로 변환한다. 입력에 포함되는 노이즈 서지를 흡수하는 필터회로와 노이즈가 내부에 들어가지 않도록 하기 위한 절연회로로 구성된다. 절연소자로는 포토커플러나 트랜스포머가 사용된다.

(2) 출력부

시퀀스 컨트롤러의 내부지령을 IC 신호레벨에서 외부의 강전 출력레벨로 변환하며 홀드회로와 잡음대책용의 절연회로 및 출력레벨 증폭용의 앰프로 구성된다.

절연소자로는 포토커플러나 릴레이 등이 사용되며 앰프소자로는 전력용 트랜지스터나 트라이



<그림 7 > 시퀀스 컨트롤러의 내부구성

악(Triac) 또는 릴레이 접점이 사용된다.

(3) 데이터 메모리

릴레이 시퀀스 안에 있는 보조릴레이로서(외부에 접속되지 않은 내부 릴레이) 타이머나 카운터로 사용할 수 있는 기종도 있다.

기억소자로는 IC 메모리나 코어 메모리 등이 사용된다.

(4) 입력선택부

프로그램 내용의 지시에 따라 입력부나 데이터 메모리를 선택하여 연산 동작용의 데이터로 한다.

(5) 출력선택부

프로그램 내용이 지시에 따라 데이터 메모리나 출력부의 출력점을 선택한다.

(6) 프로그램 메모리

이용자가 프로그램장치를 이용하여 시퀀스 컨트롤러에 입력한 프로그램을 저장한다.

이 프로그램은 시퀀스 컨트롤러를 실행하는 상태에서는 시퀀스 컨트롤러 내부의 동작순서를 나타내는 것으로 저장된 순번에 따라 읽어내고 목적으로 하는 릴레이 시퀀스를 실행하게 된다.

기억소자로는 IC 메모리나 코어 메모리가 사용되며 이용자는 릴레이 심볼에 따라서 프로그램을 작성한다.

(7) 제어회로부

프로그램을 정확하게 실행시키기 위해서는 메모리를 읽어내는 타이밍이나 입력을 선택하는 타이밍 등의 각부 동작 타이밍을 일정한 순서로 해둘 필요가 있다.

이러한 동작의 타이밍을 취급하는 부분이 제어회로부이며 실행할 때에 프로그램 메모리를 읽어내는 순서를 지정하는 프로그램 카운터를 내장하고 있다.

(8) 논리연산부

프로그램 메모리에서 읽어낸 프로그램에 따라 입력선택부에 의하여 선택된 데이터와 연산 레지스터의 내용 데이터에 의하여 필요한 연산을 행한다.

(9) 연산 레지스터

릴레이 시퀀스 내의 접점과 코일의 동작상태(또는 전기의 도통상태)에 대응하는 1비트의 기억회로이다.

릴레이 시퀀스를 실행할 때에 중심적인 존재이며 프로그램을 실행한 결과는 항상 이 레지스터에 기억되고 있다.

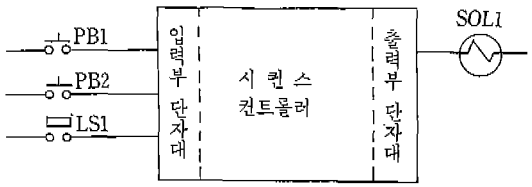
(10) 프로그램장치

이용자가 프로그램을 시퀀스 컨트롤러에 입력할 때 사용하며 사람이 쉽게 익숙할 수 있는 릴레이 심볼과 10진수에 의한 프로그램을 프로그램 메모리에 저장시킨다. 즉 심볼이나 10진수를 프로그램 메모리 내에 있는 2치논리의 조합에 번역시키는 기능을 갖는다.

2·1 제어내용은 어떻게 프로그램화되는가

시퀀스 컨트롤러는 입출력기기만 접속하고 내부 릴레이는 접속할 필요가 없다. 그림 6의 모델 릴레이 시퀀스는 시퀀스 컨트롤러에서 그림 8과 같이 접속된다.

입출력기기는 입력부와 출력부의 端子臺에 접



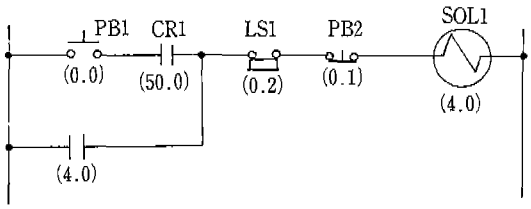
<그림 8 > 시퀀스 컨트롤러의 접속

속되는데 프로그램상에서는 입력부와 출력부의 단자대 번호(입출력 어드레스라고 한다)를 가지고 입출력기기의 명칭으로 한다.

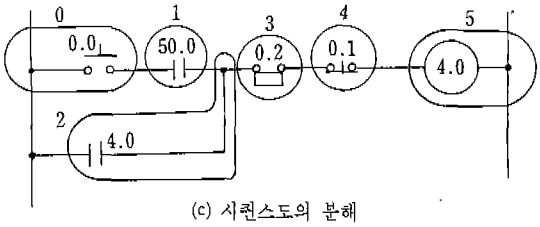
그리고 실제의 입출력 기기명과 프로그램상에 있는 입출력 어드레스의 관계를 기록한다. 또한 내부 릴레이에도 시퀀스 컨트롤 내에 고유번호가

기 기 명	입출력 어드레스	내 용
PB 1	0.0	시동 푸시버튼
PB 2	0.1	정지 푸시버튼
LS 1	0.2	정지조건 리미트스위치
SOL 1	4.0	부하 솔레노이드
CR 1	50.0	시동조건

(a) 입출력 어드레스



(b) 모델 시퀀스에의 입출력 어드레스 기입예



(c) 시퀀스도의 분해

<그림 9 > 모델 릴레이 시퀀스에 입출력 어드레스를 기입한 예

있으며 사용상태에 따라 이것을 기록한다.

그림 9 의 (a)는 입출력 어드레스를 기록한 예이며 그림 9 의 (b)는 이 결과에 따라서 모델 릴레이 시퀀스에 입출력 어드레스를 기입한 예이다.

그림 6 에서 自己維持用의 CR2가 있었으나 그림 9 의 (b)에서는 「4.0」에 의하여 자기유지를 하고 있다. 이것은 시퀀스 컨트롤러에서 데이터 메모리가 입출력부까지 기능을 커버하고 있기 때문에 가능한 것이다.

일반적으로 입출력부는 여러 點의 입출력을 한곳으로 모은 카드를 재차 여러 개로 집합시킨 구성이다. 그림 9 (a)의 입출력 어드레스는 카드번호와 카드 내에서 입출력 번호를 「•」으로서 구별하고 있다.

이와 같이 입출력 어드레스가 기입된 시퀀스도에 따라서 프로그램을 실행하지만 시퀀스 컨트롤러에 비치된 AND, OR 등의 프로그램을 나타내는 언어(명령어)의 기능과 종류는 기종에 따라서 약간의 차이를 나타낸다(표 1 참조).

표 1 의 명령체계에 따라서 그림 9 (b)를 각 제어소자마다(접점/코일, 입출력 어드레스, a 접점/b 접점) 그림 9 의 (c)와 같이 분해한다.

이와 같이 분해한 시퀀스 내용을 순번으로 기

<표 1 > 명령체계의 일부 예

X : 입출력 어드레스

명 령 어	약기법	심 볼	
READ ×	R ×	X	입력
READ NOT ×	RN ×	X	반전입력
AND ×	A ×	X	직렬접속
AND NOT ×	AN ×	X	반전직렬접속
OR ×	O ×	X	병렬접속
OR NOT ×	ON ×	X	반전병렬접속
WRITE ×	W ×	X	출력

<그림10> 코딩 예(코딩 시트)

프로그램 스텝	명 령 어		릴레이 심볼
0	R	0.0	0.0
1	A	50.0	50.0
2	O	4.0	4.0
3	AN	0.2	0.2
4	AN	0.1	0.1
5	W	4.0	4.0

록(코딩)한다(그림10 참조).

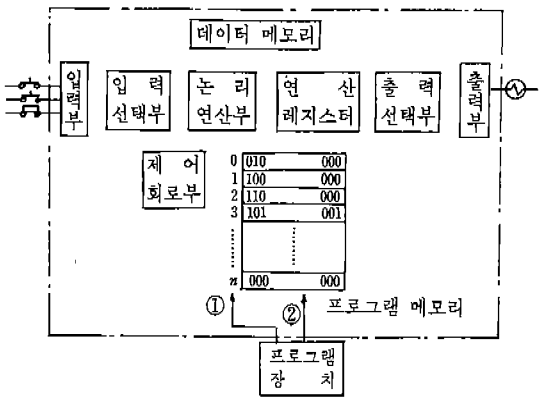
이상과 같이 릴레이 시퀀스에서 코딩까지의 작업이 프로그래밍이며 그림10을 작성하는 시점에서 릴레이 시퀀스가 프로그램화된 것이다.

2·2 프로그램은 어떻게 시퀀스 컨트롤러에 저장되는가

그림10의 코딩 시트에 따라서 이용자는 프로그램장치를 사용해서 프로그램 메모리에 각 프로그램의 스텝마다 명령어를 저장시킨다. 이 상태를 그림11에 나타내었다. 프로그램 메모리에는 코딩 시트의 프로그램 스텝에 대응하는 번호(프로그램 어드레스)가 있으며 각 명령어가 프로그램의 어드레스마다 메모리가 기억된다.

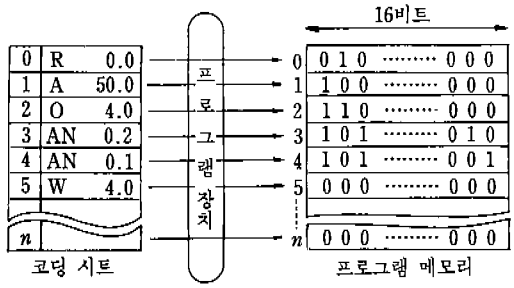
그림11에서 프로그램장치는 프로그램 어드레스 ①을 송출하고 명령어 ②를 프로그램 메모리에 저장시킨다.

명령어는 코딩 시트에서 사람이 알기 쉬운 것으로 표현되나 프로그램 메모리 내에서는 1, 0의 2치논리를 조합하여 명령어를 표현한다. 이러한 상태를 그림12 (a)에 나타내며 1명령어는 16비트로 구성되었다.

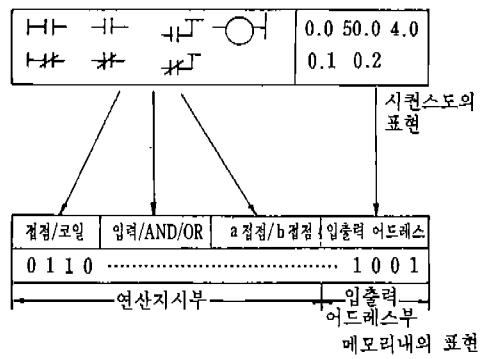


<그림11> 프로그램의 저장

프로그램장치는 이러한 표현의 변환을 자동적으로 실행하나 변환할 때에 프로그램 메모리 내에 있는 1명령어의 비트패턴을 기호언어의 내용



(a) 코딩 시트로부터 프로그램 메모리로



(b) 프로그램 내로의 명령어의 변환

<그림12> 명령어 상태

을 분해하여 시퀀스 컨트롤러 내부가 이해하기 쉽도록 의미를 붙이고 있다. 이러한 상태를 그림 12의 (b)에 나타내었다.

2·3 프로그램은 어떻게 실행되는가

목적으로 하는 제어내용은 앞에서 설명한 것처럼 프로그램 메모리에 저장되며 또한 프로그램의 실행은 제어회로로부터 동작지령 타이밍에 의하여 이루어진다. 이러한 상태를 그림13의 (a)~(h)에 나타낸다. 각 스텝의 실행순서를 보면 다음과 같다.

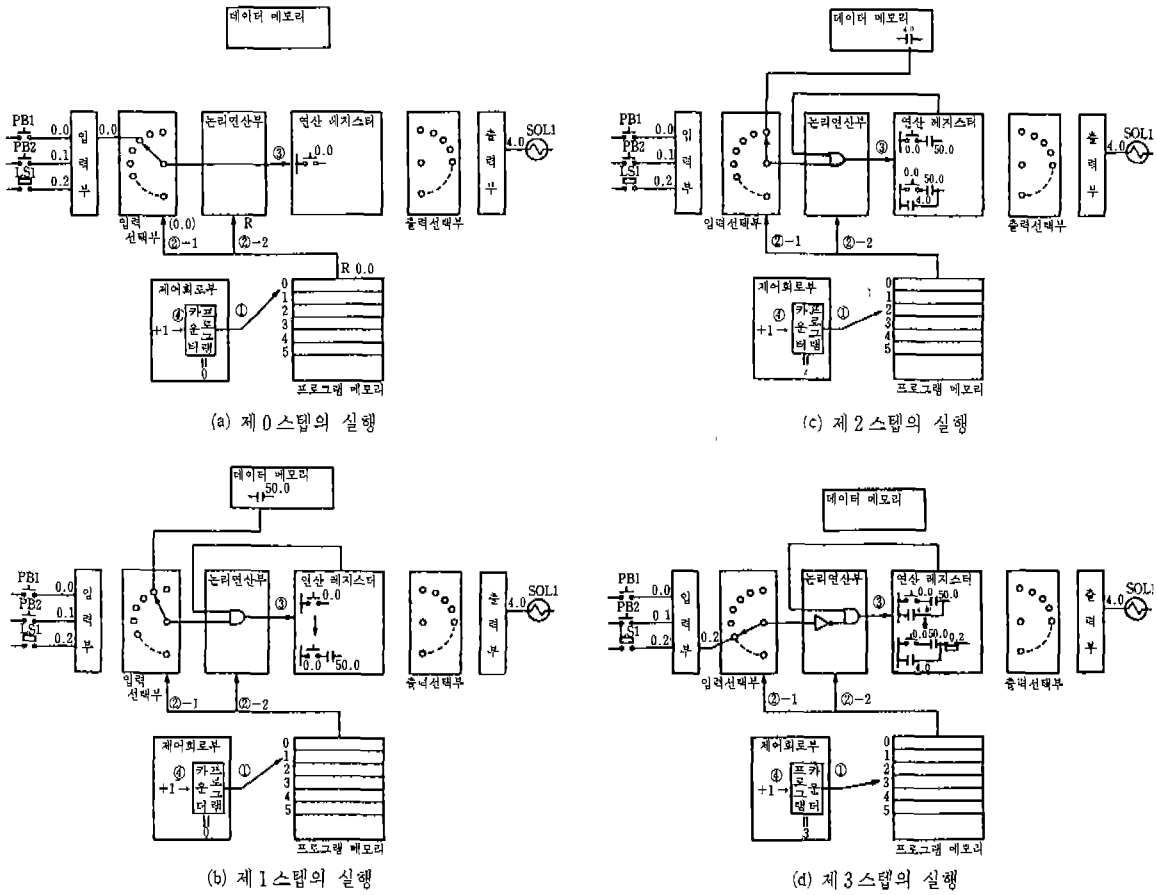
(1) 제로스텝의 실행(그림13 (a))

(a) 프로그램 카운터는 "0"에서 시작하며 프로그램 카운터의 내용에 따라 지정되는 프로그램 어드레스(=0)의 프로그램 메모리에 있는 명령어를 인출한다.

(b) 명령어의 내용에 따라서 다음과 같이 두 가지 동작이 이루어진다.

① 입력선택부가 입력「0.0」을 선택한다(명령어 내의 입출력 어드레스부가 송출된다).

② 선택된 입력이 논리연산부 내를 그대로 통과한다(명령어 내의 연산지시부가 송출된다).



(a) 제 0 스텝의 실행 (c) 제 2 스텝의 실행

(b) 제 1 스텝의 실행 (d) 제 3 스텝의 실행

<그림13> 동작지령 타이밍에 의한 프로그램 실행 (a) (b) (c) (d)

(c) 논리연산부의 출력(=「0.0」)을 연산 레지스터로 읽어들인다.

(d) 프로그램 카운터의 내용을 +1한다. 이 결과로 프로그램 카운터의 내용은 1이 되고 프로그램 스텝이 1로 진행하게 된다.

(2) 제1스텝의 실행(그림13 (b))

(a) 프로그램 카운터의 내용에 따라 지시되는 프로그램 메모리에 명령어를 읽어낸다.

(b) 명령어에 따라 다음 2가지 동작이 이루어진다.

① 데이터 메모리의 보조릴레이 「50.0」가 입력선택부에 의하여 선택된다.

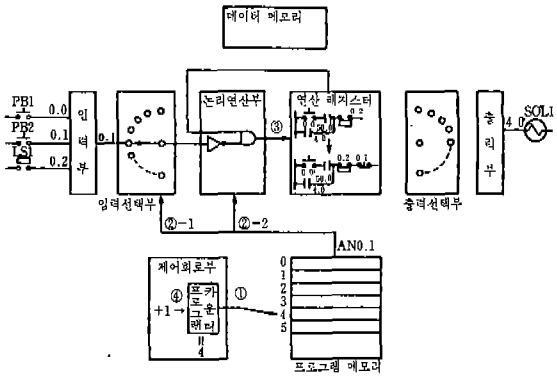
② 선택된 입력은 논리연산부 내에서 연산 레지스터의 내용과 AND가 된다.

(c) 논리연산부의 출력을 연산 레지스터로 읽어들인다.

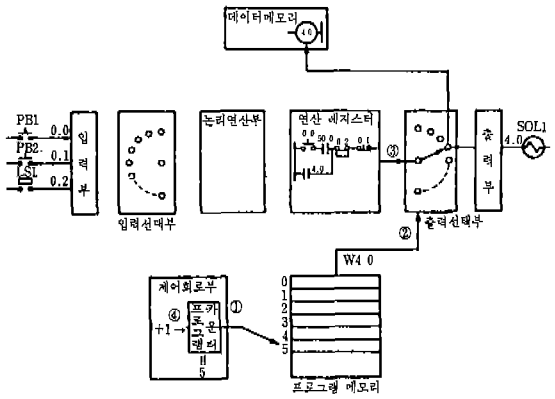
(d) 프로그램 카운터의 내용을 +1 한다.

(3) 제2스텝의 실행(그림13 (c))

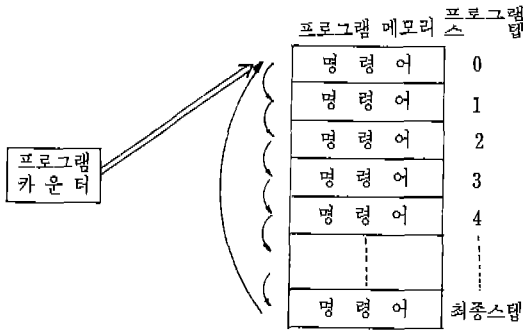
(a) 프로그램 카운터의 내용에 따라서 지시되는 프로그램 메모리의 명령어를 읽어낸다.



(e) 제4스텝의 실행



(f) 제5스텝의 실행



(g) 프로그램 카운터에 의한 프로그램내의 명령어 취출 (스캐닝)

프로그램 스텝	연산결과(연산 레지스터 내용)
0	0.0
1	0.0 50.0
2	50.0 4.0
3	0.0 50.0 0.2 4.0
4	0.0 50.0 0.2 0.1 4.0
5	0.0 50.0 0.2 0.1 4.0 4.0

(h) 프로그램 스텝의 진행에 의한 연산결과

<그림13> 동작지령 타이밍에 의한 프로그램 실행 (e)(f)(g)(h)

(b) 명령어의 동작에 따라 다음 2가지 동작이 이루어진다.

① 데이터 메모리에서 출력부 「4.0」에 상당하는 보조릴레이 「4.0」이 입력선택부에 의하여 선택된다.

② 선택된 입력은 논리연산부 내에서 연산 레지스터와 OR된다.

(c) 논리연산부의 출력을 레지스터로 읽어들인다.

(d) 프로그램 카운터의 내용을 +1 한다.

(4) 제 3 스텝의 실행(그림13 (d))

(a) 프로그램 카운터의 내용에 따라 지시되는 프로그램 메모리의 명령어를 읽어낸다.

(b) 명령어의 내용에 따라 다음과 같이 두 가지 동작이 이루어진다.

① 입력선택부가 입력 「0.2」를 선택한다.

② 선택된 입력은 논리연산부 내에서 반전되고 연산 레지스터의 내용과 AND된다.

(c) 논리연산부의 출력을 레지스터로 읽어들인다.

(d) 프로그램 카운터의 내용을 +1 한다.

(5) 제 4 스텝의 실행(그림13 (e))

(a) 프로그램 카운터의 내용에 따라 지시되는 프로그램 메모리의 명령어를 읽어낸다.

(b) 명령어의 내용에 따라 다음과 같이 두 가지 동작이 이루어진다.

① 입력선택부가 입력 「0.1」을 선택한다.

② 선택된 입력은 논리연산부 내에서 반전되고 연산 레지스터의 내용과 AND된다.

(c) 논리연산부의 출력을 연산 레지스터로 읽어들인다.

(d) 프로그램 카운터의 내용을 +1 한다.

(6) 제 5 스텝의 실행(그림13 (f))

(a) 프로그램 카운터의 내용에 따라 지시되는 프로그램 메모리의 명령어를 읽어낸다.

(b) 명령어의 내용에 따라 출력선택부가 출력부 「4.0」과 출력부 「4.0」에 상당하는 데이터 메모리의 보조릴레이 「4.0」을 선택한다.

(c) 선택된 출력에 연산 레지스터의 내용을 기입한다. 이 출력은 홀드된다.

(d) 프로그램 카운터의 내용을 +1 한다.

이하 프로그램의 실행은 위와 같이 프로그램 카운터가 전진함에 따라 각 프로그램 스텝마다 명령어를 실행한다. 그리고 최종 프로그램 스텝까지 실행이 끝나면 다시 프로그램 카운터의 내용은 0으로 세트되고 재차 0스텝부터 반복 실행된다.

이와 같은 프로그램의 주사를 스캐닝이라고 하며 고속도로 반복된다. 이러한 상태를 그림13 (g)에 나타내었다.

출력부의 홀드는 각 주사마다 내용이 갱신되고 주사가 고속으로 이루어지기 때문에 릴레이의 동작시간과 동등한 응답속도로 목적하는 제어를 달성시킬 수 있다.

프로그램 스텝의 진행에 따라 연산 결과가 순차적

으로 변화하는 상태를 그림13 (h)에 나타내었다.

그림13 (h)는 프로그램을 작성하는 단계인 그림 9의 (c)에 의하여 분해된 릴레이 시퀀스를 재구성한 형태로 되어 있다.

따라서 명령체계인 표1의 명령을 보면 명령어는 릴레이 시퀀스와 같은 형태로 되어 있으나 시퀀스 컨트롤러 내부의 동작지령은 실제로 표2와 같은 의미로 되어 있다. 여기서 설명한 시퀀스 컨트롤러의 내부구성은 기본적인 것으로서 이용자는 각자 목적하는 대로 내부를 여러 가지로 구성할 수 있다.

맺음말

프로그래머블 컨트롤러는 FA 공정제어장치의 기초단위로서 매우 중요한 위치를 차지하고 있으며 기존의 릴레이 제어반의 단점을 매우 효율적으로 해결하고 있다. 장점을 보면 첫째, 유연성의 확보에 있다.

즉 공정제어논리의 변경이 용이하므로 제어장치 설치 및 시운전 시간이 단축되며 소량다품종 생산체제에 즉시 적응할 수 있고 사후에 기능추가와 변경이 쉽다.

둘째로 제어논리가 소프트웨어로 프로그래머블 컨트롤러에 내장되므로 기존 릴레이반에 비하여 소형화되고 설치면적이 줄어들어 소비전력이 절감된다.

셋째, 프로그래머블 컨트롤러는 일종의 공정제어용 전용컴퓨터로서 표준화 장비이므로 이것을 이용하여 제어반을 구성하면 제조품질의 평준화가 가능하고 배선작업이 크게 줄어든다.

또한 제조경비가 저렴해지고 생산 리드타임이 줄어들어서 납품기일이 단축된다는 것이다.

프로그래머블 컨트롤러의 규격으로서 가장 중요한 것은 最大入出力點數(프로그래머블 컨트롤러 CPU에 동시에 연결할 수 있는 입출력 채널의 최대수요)로서 보통 최대입출력점수 128점 이내인 프로그래머블 컨트롤러를 소형, 512점까지를 중형, 그 이상(대략 2048점까지)을 대형 프로그래머블 컨트롤러로 구분하고 있다.

한편 프로그래머블 컨트롤러의 국내 시장규모를 보면 본체만 500억원 부품까지 합쳐 1천억으로 추산하고 있으나 아직 국내 시장은 일본업체와의 기술제휴에 의존하고 있어 국산품이 경쟁력을 갖추기까지는 일본제품의 범람이 지속될 것이다.

또한 국내의 기술수준은 일본을 100으로 할 때 설계, 가공, 조립 등 생산기술은 80, 신뢰성, 소프트웨어 구성, 시스템 응용 등 제품기술은 70 수준으로 평가하고 있으며 프로그래머블 컨트롤러 업체는 '94년까지 처리용량 51,200점, 메모리용량 128kbyte 이상, 통신속도 10메가바이트(초당)까지의 개발수준을 목표로 하고 있다.

끝으로 본고에서는 프로그래머블 컨트롤러에 대한 H/W, S/W 및 동작개요만을 언급했으나 다음 호에 적용사례, 선정시 고려사항, 프로그래밍의 실무지식 및 노이즈 대책 등을 기술하고자 한다.

☞ 다음 호에 계속

<표2> 명령어의 시퀀서 내부동작에 대한 의미

명 령 어	시퀀서 내부동작에서의 지령
READ X	입력으로서 X를 선택하고 연산 레지스터에 저장하라.
READ NOT X	입력으로서 X를 선택하고 반전한 후에 연산 레지스터에 저장하라.
AND X	입력으로서 X를 선택하고 연산 레지스터의 내용과 AND한 후에 연산 레지스터에 저장하라.
AND NOT X	입력으로서 X를 선택하고 반전시켜서 연산 레지스터의 내용과 AND한 후 연산 레지스터에 저장하라.
OR X	입력으로서 X를 선택하고 연산 레지스터의 내용과 OR한 후에 연산 레지스터에 저장하라.
OR NOT X	입력으로서 X를 선택해서 반전시키고 연산 레지스터의 내용과 OR한 뒤에 연산 레지스터에 저장하라.
WRITE X	출력으로서 X를 선택하고 연산 레지스터의 내용을 저장, 출력하라.