教育用 로보트의 設計

金 容 得
亞洲大學校 電子工學科 敎授(工博)

Ⅰ. 로보트 개요

1968년에 미국의 play back형 로보트가 소개된 것을 계기로 많은 연구가 진행되기 시작하였으며 일본의 전자기술종합연구소(ETL)에서도 손과 눈에 대한 연구를 개시하여 1970년 ETL 로보트를 공개하므로서 학계에서도 새로운 연구 분야로 인정받게 되었다.

1970년대 초반 마이크로프로세서의 출현은 급속한 로보트의 발전을 가져와 1977년경에는 캘리포니아 공과대학에서는 이동형 自立 知能 로보트를 개발하였다.

여기서 로보트라함은 로보트가 처해 있는 환경에서 데이타를 얻기 위한 기능인 感覺系와 로보트가 그 의지에 따라 환경에 작동하기 위한 기능, 즉 손 또는 발과 같은 動作系, 주위 환경과 로보트 자신의 상태를 알아서 다음에 해야 할 동작을 결정하는 고도의 推論系를 갖고 있어야 한다.

이러한 로보트를 설계, 제작하기 위하여는 기계공학, 전자공학 및 인공 지능(computer software) 분야에 걸쳐 협력이 이루어져야 하겠다.

국내에서도 로보트에 관한 연구가 여러 분야에서 매우 활발히 진행되고 있으나 아직 초기 단계이며, 따라서 모든 사람이 로보트에 대하여 좀더 쉽게 접할 수 있고 산업용 로보트 이전의 早期敎育이나 컴퓨터에 의한 제어 동작등을 이해시키기 위하여 본 연구팀과 G사가 공동 개발한 교육용 로보트에 대하여 기술하고자 한다.

Ⅱ. 시스템 특성

교육용 로보트는 多關節 高知能 미니 로보트로서 각급 학교의 첨단 기술 조기 교육용 또는 연구소의 시험 연구 실용화 및 단순 산업용에 이용할 수 있도록 설계된 것으로 6개의 스텝 모터들을 구동시키므로서 자유 자재의 동작을 가능하게 한다. 즉 모터1은 몸통의 회전 동작(회전 각도 180°)을 제어하며 모터2는 팔의 어깨에 해당되는 것으로 상하 동작(135°), 모터3은 팔꿈치 부분으로 상하 동작(165°), 모터4와 5는 손목의 회전 동작과 상하 동작(각각 180°와 360°) 또 모터6은 손가락의 조임과 이완 동작(최대 8㎝)을 제어하는 것으로 그림1과 같다. 특히 이 로보트는 교육용이므로 산업용과는 달리 물체에 의한 손가락의 압력이 감지되어 더 이상 닫아지지 않으며 무게가 어느 한계 이상이면 동작이 되지 않게 되어 있다.

그림 1. 교육용 로보트

표1에서는 교육용 로보트의 작동 방법과 이들에 대한 기능을 설명한 것으로 이는 독자적으로 사용되기도 하지만 일반 퍼스널 컴퓨터나 또는 로보트 제어를 위한 教示箱子(teaching box)와 연결하여 사용할 수도 있다. 즉 독자적으로 사용할 때는 몸체에 부착되어 있는 push botton을 누르는 정도에 따라 Test Ⅰ mode와 Test Ⅱ mode로 나뉘어 동작한다. 이들 test mode는 각 모터들의 상태를 점검하기 위한 것으로 Test Ⅰ mode에서는 push botton을 누르는 상태에

표 1. 시스템 오퍼레이션

1. Test Ⅰ Mode	이 mode에서는, 초기상태에서 finger close된 후에 입력되는 test key값에 따라 모터가 동작하게 된다. Test key가 push된 상태에서만 모터가 동작하게 되는데 push된 횟수에 따라 각 모터와 그 방향이 결정된다. Push된 횟수가 0, 1일때는 M_1이 ± 방향으로, 2, 3일때는 M_2가 ± 방향으로, 4, 5일때는 M_3가 ±방향으로, 6, 7일때는 M_4가 ±방향으로, 8, 9일때는 M_5가 ±방향으로, 10, 11일때는 M_6이 ±방향으로 이후에는 다시 M_1부터 되풀이 하여 모터와 방향이 결정되며 그 값에 따라 각 모터가 동작하게 된다. 각 모터가 동작하는 시간은 test key가 push된 시간에 비례한다.
2. Test Ⅱ Mode	이 mode에서는, ARM이 position/chome position) 부터 position 6까지, 그리고 다시 home position으로 연속적으로 동작하게 되며 이러한 과정이 3번 되풀이 되는데 처음에는 speed value/속도(느리게)로, 두번째는 speed value 3 속도(보통)로 그리고 세번째는 speed value 5 (빠르게) 속도로 움직인다. 이 동작이 끝나면 다시 test key를 check하게 되며, test key가 push 상태이면 동작을 멈추고 no push 상태이면 이 과정을 처음부터 다시 되풀이 한다. 또 지연 데이타값(6개)은 ROM에 내장되어 있다.
3. Teaching Mode	이 mode에서는, jog mode, program mode, run mode가 있으며, jog mode에 의하여 각 mode를 동작시켜 원하는 위치들을 선정할 수 있으며 이들 위치들을 프로그램 모드에서 프로그래하여 RUN mode에서 연속적인 동작을 하도록 할 수 있다. 이 동작을 계속 반복하여 수행하도록 할 수도 있다.
4. Online Mode	이 mode에서는, 시스템이 자동적으로 동작하지 않고 일단 입력 key값을 받아 들인다. 입력되는 key값들에 의하여 ARM이 동작하게 되는데 key값의 선택에 따라 여러가지 function을 수행할 수 있다. 현재 이 시스템에서 수행되는 function에는 finger open(close), here, home, move, mover, limit, nest, position, out 8, out 4, speed, time, switch high, switch low 등이 있다.

따라 각 모터가 모터 1부터 차례로 시계방향(+) 또는 반대 방향(−)으로 이동된다.

Test Ⅱ mode는 push botton을 약 5초이상 지속적으로 누르면 내장되어 있는 프로그램을 작동하여 위치 1 (home position)부터 위치 6까지 그리고 다시 위치 1으로 같은 동작을 3회 연속적으로 구동 속도를 바꾸면서 작동한다(처음은 느리게 후에는 빠르게).

이들 로보트는 5단계 속도로 구동시킬 수 있다.

ON-line mode는 로보트를 퍼스널 컴퓨터에 연결시킴으로써 컴퓨터 key 입력 값들에 의하여 여러가지 기능을 수행하게 되며, 이들 기능에는 finger open (close), here, home, move, position, speed, time 등이 있다. 이들 기능에 대한 설계 방식은 다음 장에서 설명된다.

Teaching mode는 教示箱子를 로보트에 부착함으로써 이루어지는데 이 mode는 다시 jog, program, run mode로 작동된다. Jog mode에서는 각 모터를 동작시켜 원하는 위치들을 설정할 수 있으며 이들 위치들을 프로그램 모드에서 프로그램하여 다음의 run 모드에서 연속적인 동작을 수행할 수 있게 된다.

이 동작은 계속 반복할 수 있으며 중간에 다른 동작을 삽입하거나 혹은 불필요한 동작을 삭제하도록 할 수도 있다.

Ⅲ. 시스템 設計

로보트 시스템은 크게 힘을 전달하는 기계부와 주어진 명령어에 따라 기계부를 제어할 수 있는 전자 제어부로 나누어지며 본 설계에서는 전자 제어부에 대하여만 고찰하도록 한다.

전자 제어부는 教示 箱子 또는 외부로부터 프로그램에 의하여 로보트를 작동시킬 수 있는 마이크로프로세서 제어부와 이들 논리신호로부터 스텝모터를 구동시킬 수 있는 motor buffer board 및 직접 key 입력에

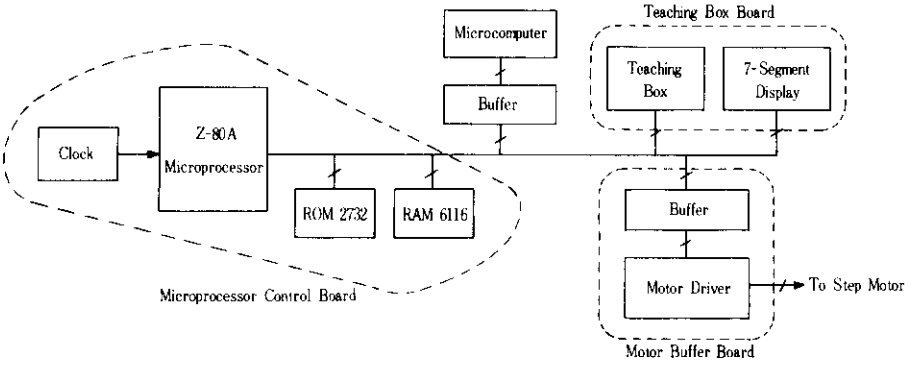


그림 2. 시스템 블럭 다이어그램

의하여 로보트를 구동시킬 수 있는 教示箱子로 그림 2와 같이 구성되어 있다.

마이크로프로세서 콘트롤 보오드는 Z-80A, ROM, I/O port 등을 가진 완전한 마이크로컴퓨터 시스템을 이루고 있고, 필요에 따라 기억장치의 확장이 가능하며 이때의 memory map과 I/O map은 표 2와 같다.

표 2. 기억장치와 포트의 구성

(a) Memory Map

Addr.	Data
0000H	MONITOR (EPROM)
3FFFH	Net Used
8000H	RAM (8K Byte)
A000H	Not Used
FFFFH	

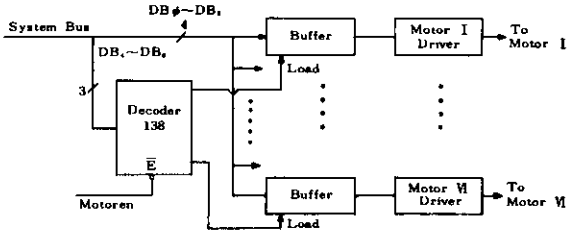
(b) Input Port

Addr.	Command	Comment
00H	D READY	Bitϕ H : Ready L : Not Ready
40H	D READ	DB_0~DB_7
80H	Test K	Bitϕ H : No Push L : Push
A0H	TEACH	Bitϕ H : Not Connect L : Connect
C0H	Limit	
E0H	Key RD	DBϕ~DB_4

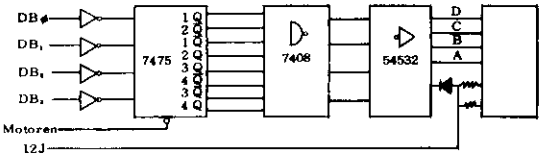
(c) Out Port

Addr.	Command	Comment
00H	DNTR	DBϕ~DB_3
20H	Ack	DBϕ
40H	LAMP	DBϕ~DB_7
60H	PDOUT	DBϕ~DB_7
80H	7 SEG	DBϕ~DB_7
A0H	Error	DB_7 : H : SET L : RESET
E0H	Motor EN	DBϕ~DB_7, 표 3 참조

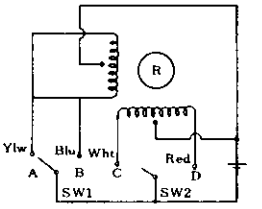
모터 버퍼 보오드는 표 3과 같이 OUT EOH 명령문 즉 MOTORN 신호가 발생되었을 때 시스템 버스의 값에 따라 각각의 모터를 시계 방향 또는 반시계 방향으로 회전시킬 수 있으며 이때의 모터 구동회로는 그림 3과 같다.



(a) 블럭 회로



(b) 버퍼 및 드라이버단



(c) 스텝핑 모터 와이어링 다이어그램

그림 3. 모터 버퍼 보오드

표 3. 모터 제어 상태표(OUT EOH 명령문 수행시)

(a) 모터 선택 상태도

DB_6	DB_5	DB_4	선택모터
0	0	0	모 터 I
0	0	1	모 터 II
0	1	0	모 터 III
0	1	1	모 터 IV
1	0	0	모 터 V
1	0	1	모 터 VI

(b) 모터의 회전 방향

DB_0~DB_7	모터 동작 방향
XAH, X6H X5H, X9H	반 시계 방향
X9H, X5H X6H, XAH	시계 방향

데이타 버스의 하위 4비트(DB_0~DB_3)로는 모터의 방향, 상위 3비트(DB_4~DB_6)로는 모터를 표 3과 같이 선택한다.

다음에 教示箱子의 key board는 그림 4와 같이 배열되며 이 원리는 key를 누르면 그 키에 대하여 기능이 정의된 서브루틴 프로그램을 수행하도록 일종의 하드웨어에 의한 서브루틴 CALL 방식으로 설계하였다.

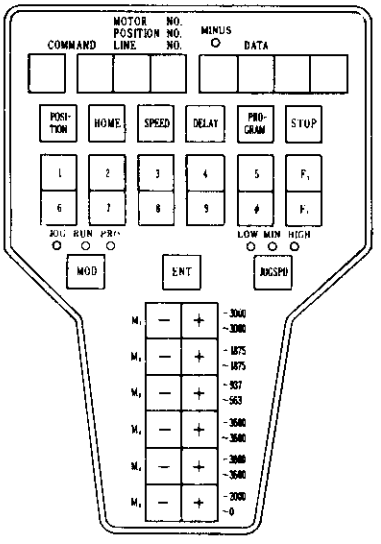


그림 4. Teaching box의 key 자판

Ⅳ. 시스템 프로그램 設計

앞에서 설계된 하드웨어가 Test Ⅰ mode, Test Ⅱ mode, teaching mode 및 on-line mode중 어느 것으로도 구동시킬 수 있도록 작성한 시스템 프로그램의 흐름표는 그림 5와 같다.

전원 스위치가 ON되면 먼저 RAM을 clear하고 각 모터에 초기치(동작속도와 현재의 위치등)를 설정한 다음 test key를 확인한다. Test 키는 누르는 시간에 따라 Mode Ⅰ과 Mode Ⅱ로 프로그램에 의하여 검지된 후 수행하도록 하고 test 키가 눌려지지 않은 경우에는 教示箱子가 연결되었는가를 확인한후 연결된 경우에는 항상 teaching mode로 작동하도록 설계하였다.

Teaching mode는 모드변환 키에 의하여 jog, program, run mode로 작동하게 된다.

Jog 모드란 사용자가 로보트의 팔을 시험 작동시키면서 원하는 위치와 속도등을 기억시키는 것으로 흐름표는 그림 6과 같다.

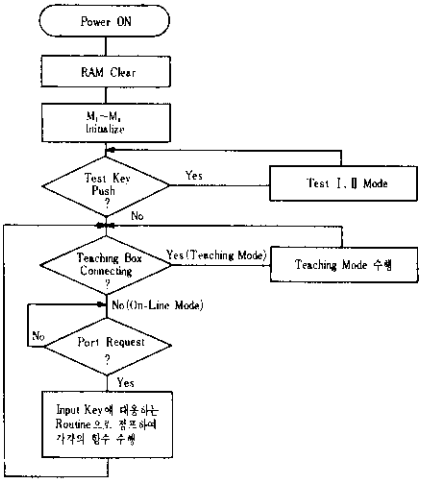


그림 5. 시스템 프로그램

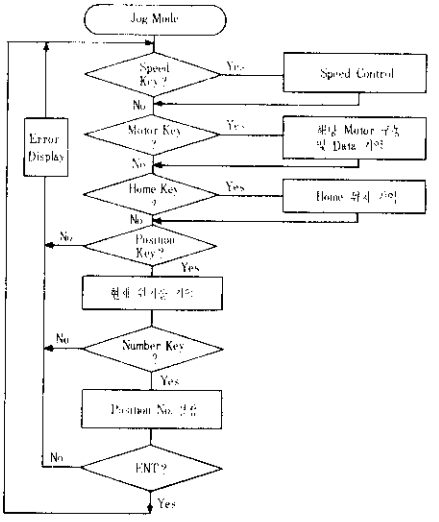


그림 6. Jog mode의 흐름표

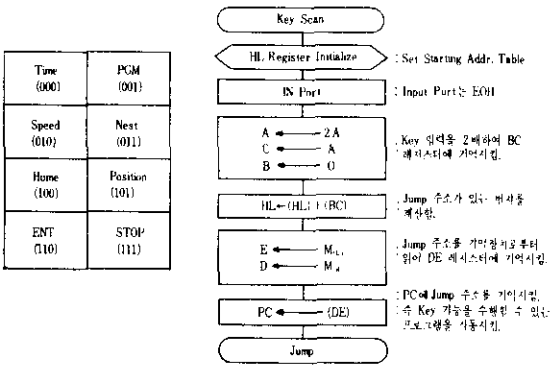
프로그램 모드는 로보트 팔 동작을 jog 모드에서 정의된 동작을 연속적으로 수행할 수 있도록 프로그램을 시스템에 배열시킬 수 있는 모드이고, run 모드는 앞의 프로그램 모드에서 입력시킨 프로그램들을 연속적으로 실행시키는 모드로서 jog 모드와 비슷한 흐름표를 갖는다.

教示箱子가 연결되지 않은 상태는 on-line 모드로서 연결된 컴퓨터에서 신호가 입력되는가를 검지하는 동작을 반복한다. 물론 이때 신호가 입력될 때는 그 신호에 입력되는 기능을 수행하게 된다.

예를들어 그림 4의 키들중 "N"키(Nest(N)키는 로

보트의 팔이 현재 위치에서 처음에 설정되었던 원점위치로 이동하라는 키임.)가 눌려지면 모니터 프로그램은 "ODBB"번지의 N키 수행 프로그램으로 점프하게 된다.

이들 과정을 그림 7의 (a)와 같이 몇 개의 키에 대하여 배열하여 설명하기로 하겠다.



(a) Function key와 key 코드된 값

(b) Sub routine jump 흐름도

그림 7. Key 정의에 따른 모니터 프로그램의 jump 방식

이들 키의 프로그램 시작 주소는 1000번지부터 각각 2바이트씩 기억되어 있다고 한다면 N키의 프로그램 시작 번지는 하위 주소는 1006번지, 상위 주소는 1007번지에 즉 1006번지에는 "BB", 1007번지에는 "OD"가 있게 된다.

이때 "N"키가 눌려졌다고 한다면 이 키는 "011"의 2진수로 코드화되어 마이크로프로세서에 인지된다. 이렇게 인지된 "011"이 1006번지로 변환하기 위하여는 b에 보여준 흐름표에 따라 2를 곱해 주고 1000(HL레지스터의 값)을 더하여 HL 레지스터에 기억시키면 된다.

여기서 실질적으로 프로그램이 점프될 주소는 HL 레지스터가 지시하는 기억장치의 내용을 읽어 D와 E 레지스터에 기억시킨 후 DE 내용으로 프로그램을 점프시키면 "N"키의 기능이 수행하게 된다.

V. 예제 프로그램

로보트 팔을 속도와 시간의 조절하에 원점에서부터 연속적으로 동작시키는 프로그램의 예를 教示箱子와 파스컴에 의한 프로그램을 각각 표 4, 표 5에 보여 주었다.

표 4. 教示箱子에 의한 프로그램 예

HOME , ENT	: Arm이 원점위치에 이동한다.
SPEED , 5 , ENT	: Arm의 동작속도를 speed 5로 놓는다. 이후의 arm은 항상 speed 5로 움직인다.
POSITION , 2 , ENT	: Speed 5로 position 2로 arm 이 이동된다.
DELAY , 80 , ENT	: Position 2에서 80초동안 정지한다.
POSITION , 4 , ENT	: 그 후 position 4 위치로 speed 5로 이동된다.
SPEED , 3 , ENT	: Arm의 동작속도를 speed 3으로 놓는다.
HOME , ENT	: 원점 위치로 이동한다.

표 5. 퍼스컴에 의한 프로그램 예

```
10  LPRINT  "N"
20  LPRINT  "S 5"
30  LPRINT  "P 2"
40  LPRINT  "T80"
50  LPRINT  "P 4"
60  LPRINT  "S 3"
70  LPRINT  "N"
80  END
```

VI. 결 론

본 교육용 로보트는 산업용 로보트 이전의 조기교육이나 컴퓨터에 의한 제어 동작을 이해시키기 위하여 제작된 것으로서 로보트 팔의 구동방식은 가장 많은 펄스 값을 갖는 모터부터 구동하기 시작하여 전체적으로 남은 구동 펄스 수가 같게 되면 M_1부터 차례로 한 펄스씩 작동된다.

이러한 경우 만일 M_6(finger)가 임의의 모양을 지닌 물체를 잡는다고 할때 M_6는 M_1부터 M_5가 완전히 그 물체에 접근한 후 닫히면서 물체를 잡아야 하는데 M_1부터 M_6가 동시에 구동되므로 물체 모양에 따라 불가능한 경우가 있으며 또한 본 교육용 로보트에서는 모터의 회전시간과 로보트 팔의 위치 판단을 자체적으로 할 수 있는 기능이 없다.

따라서 어떠한 경우에도 작동을 잘 수행하기 위하여는 상황에 따라 모터의 구동 우선순위를 정할수 있고, 진행상태의 위치와 접근 위치를 자체적으로 계산하여 각 모터의 속도를 자체로 제어할 수 있도록 보완된다

면 완벽한 知能 로보트로서 산업용으로도 응용이 가능하리라 기대된다.

參 考 文 獻

[1] 三菱電機, "教育用 マイクロ ロボット", 三菱電機, 1982, 3.

[2] 花房菌郎, "ロボットの 機構と 制御", 日本機械学会 論文集, 48卷 435号.

[3] M. Vukobrato Vic, Scientific Fundamentals *of Robotics.* Springer-Verlag, 1982.

[4] B rady, Hollerbach, *Robot Motion.* MIT Express, 1983.

[5] Harold S. Stone, *Microcomputer Interfacing.* Addison-Wesley Inc., 1982.

[6] 北川一, "制御用 マイコソの プログラミング", オーム社, 1983. *

♣ 用 語 解 說 ♣

Mapped Random File

Mapped random file에서는 index를 사용하지 않고도 record를 任意로 직접 찾아간다. 이렇게 하기 위해서는 key를 통해서 그 key에 대한 record 가 들어 있는 주소를 계산하여 내야 한다. 이와같이 주소를 구해내는 방식을 key-to-address변환, 혹은 hashing이라고 한다. 이와같은 계산방법에는 다음과 같은 것들이 있다. Record들을 저장할 기억장소가 주소 P에서 부터 시작하여 주소 R 까지라고 생각한다.

1. 나누기 방법

P가 0이고 R이 999라면 기억장소는 1000개가 있는 셈이다. 나누기 방법에서는 예를 들어서 36741이라는 key가 있을 때 1000에 제일 가까운 素數는 997이므로

36741÷997의 나머지=849

주소 849를 key 36741에 대한 record가 저장되어 있는 주소로 택하는 것이다. 즉 나누기 방법에서는 record들을 저장하도록 허용되어 있는 장소(이것을 extent라고 함)의 총 수효를 R-P라고 하면 R-P보다는 작으나 이것에 제일 가까운 素數 N을 찾아내어 key를 N으로 나누었을 때 나오는 나머지를 key에 대한 record가 저장되어 있는 주소로 택하는 것이다.

2. Mid-Square 방법

Key가 367415라고 하면 key를 제곱하여

367415×367415=134993782225.

이때 extent의 수효가 1000이라면 134993782225에서 중간에 있는 숫자들을 택한다. 즉 앞의 네자리인 1349를 버리고 뒤의 다섯자리인 82225를 버리면 중간의 937이 나온다. 그리하여 P+937을 key 367415에 대한 record가 저장되어 있는 주소로 택하는 것이다.

3. 접는 방법

Key가 29367415라면 중간의 숫자를 중심으로 다음과 같이 접는다.

여기서 주소범위가 세자리로 택한 것은 extent의 수효를 1000이라고 하면 key에 대한 record가 저장되어 있는 주소를 세자리 숫자로 택해야 할 것이기 때문이다. 여기서 화살표는 자리이동을 나타낸다.

즉 다음과 같이 만들어서

```
    92
   514
29 | 367 | 415
-------------
   791  상대주소→(displacement)
```

주소를 P+791로 택하는 것이다. 여기서 791은 P로부터 떨어져 있는 상대주소인 것이다. 이들 中에서 query를 提起했을 때 response time이 빠른 것으로 multilist file, inverted list file, cellular partitioned file이 많이 추천된다.

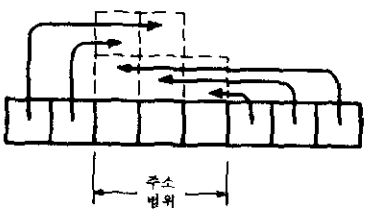


그림. 접고자리 이동