韓國通信學會論文誌'87-10 Vol. 12 No. 5

論 文

Binary-Decision 방식을 이용한 프로그래머블 콘트롤러의 개발에 관한 연구

正會員 田 炳 實* 正會員 李 俊 煥** 正會員 嚴 景 培***

The Development of Programmable Controller Using Binary-Decision Method

Byoung Sil CHON*, Joon Whoan LEE**,
Kyoung Bae EUM*** *Regular Members*

要 約 Binary Decision방법은 출력을 얻는데 요구되는 결정스텝이 최대 입력변수의 수를 넘지 않도록 한다. BD-PC 모듈은 스캔 스피드를 개선하기 위해 이 방법을 이용하여 설계제작되었다. Binary Decision방식에 수반되는 메모리문제를 개선하기 위해 컴파일러 시스템을 개발하였다. 또한, 컴파일된 BD-PC목적프로그램을 BD 머신의 메모리에 로딩하기 위해 MDS와 BD-PC모듈간에 통신채널을 구성하였다.

ABSTRACT The Binary Decision method can evaluate any switching function in the number of steps not exceeding the number of input variables. A Binary Decision Programmable Controller module is designed using this method so as to improve scan speed. A compiler system is also developed to relieve the memory problem which the Binary Decision method entails. A communication channel between MDS and BD-PC modules is also constructed to load the compiled BD-PC object program into the memory of BD machine.

I. 서 론

반도체 산업의 발전으로 종래의 릴레이로된 순서제어기는 1970년대 초반에 프로그래머블 콘트롤러(Programmable Controller 이하 PC라 약함)로 대치되어 소형화 되었을 뿐 아니라, 전력소모를 줄이고 프로그램을 교환하면 다른 공정에도 이용할 수 있도록 되었으며 현재 미국에서만도 150종류 이상이 제품화 되어 연간 2억불 가량의 거래가 이루어지고 있다.

1970년대초 미국의 제너럴 모우터(G. M.)사에서 처음으로 PC가 개발된 이래 그 기능면에서 크게 향상되었는데, 첫째로, 입, 출력 단자수

*, **, ***全北大學校 工科大學 電子工學科
Dept. of Electronics Engineering
Chonbuk National Univ., ChonJusi, 520 Korea.
論文番號 : 87-48(接受 1987. 7. 25)

가 증가되었고 다양화 되었으며, 둘째, 사용자의 프로그램 입력방식 및 고장진단이 보다 용이해졌으며, 소자의 발달에 따라 악조건의 작업환경속에서도 신뢰도가 높아졌고, 처리속도가 빨라졌으며 마이크로 프로세서의 성능향상에 따라 PID(Proportional Integral Derivative) 제어, 순서제어 등 알고리즘만으로 가능했던 현대의 복잡한 제어이론들을 직접 실현할 수 있는 기능까지도 부가되게 되었다.

국내에서, PC의 개발은 1983년 서울대 제어계측공학과, 한국 전자기술연구소 등에서 발표된 바 있고, 업계에서도 처음에는 선진외국에서 도입하여 판매하던 단계를 넘어서 자체 개발을 서두르며, 일부에서는 그 성과를 얻는 단계에 있다[1].

그러나, 대부분의 PC의 경우 Boolean 방식을 그대로 마이크로프로세서에 이식하였기에 입력수가 증가할 경우 스캔 시간(scan time)이 기하급수적으로 늘어나는 단점이 따르게 되어[2] 고속을 요하는 공정의 실시간 처리가 곤란한 단점을 가지고 있다.

이러한 스캔 시간의 문제는 입력변수들로 부터 출력을 얻는데 요구되는 결정스텝이 최대 입력변수의 수를 넘지 않는 Binary Decision 방식을 PS에 적용(이하 Binary Decision Programmable Controller를 BD-PC로 약함)함으로써 보완할 수 있음을 보였는데, 이 BD방식을 PC에 적용한 것은 1979년 캐나다 Mc Gill 대학의 Zsomber-Murray등에 의해서이다[2].

그러나, BD 방식을 이용할 경우 사용자가 BD-PC 프로그램을 만들기 위해 트리(tree)형태로 표시되는 BD다이어그램에서 직접 엔코드해야만 하는 불편함이 있을 뿐만 아니라 입력수가 증가함에 따라 BD다이어그램의 결정node의 수가 식(1)과 같이 증가하여 결국 BD-PC 프로그램을 저장하기 위한 메모리가 기하급수적으로 늘어나게 된다[2], [3].

$$d = \sum_{i=1}^{n} 2^{i-1} \qquad (1)$$

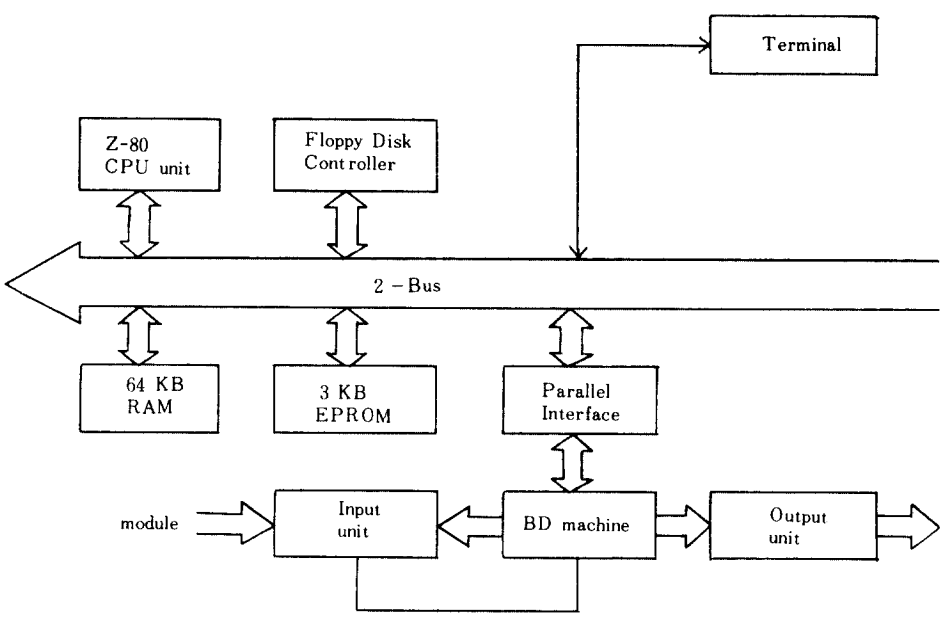

그림 1 BD-PC 시스템의 블럭선도
Block diagram of BD-PC system.

여기서 d는 결정 node수, n은 입력수를 표시한다.

따라서, BD다이어그램에서 잉여(redundancy)의 브랜치(branch)들을 제거하여 최소의 메모리 용량을 갖는 BD-PC의 목적프로그램들을 얻어내기 위한 컴파일러를 필요로 한다.

본 논문에서는 Zsomber-Murray등에 의해 제안된 BD방법을 기초로한 BD-PC모듈을 설계제작하고 이를 Zilog사의 ZDS-1/40 MDS(Microcomputer Development System)에 연결하여 BD-PC의 목적프로그램을 얻어내기 위한 컴파일러를 MDS에 내장시켜, 순서제어 등 비교적 간단한 일은 BD-PC모듈에서 처리하고, BD-PC의 목적프로그램을 얻어 내는 일과 계산을 요하는 일은 MDS에서 처리하는 계층적인 BD-PC 시스템을 연구 개발하였다.

Ⅱ. 본 론

가. 시스템 개관

그림 1 에는 개발된 BD-PC 시스템의 블럭선도를 도시하였다.

BD-PC시스템은 MDS모듈과 BD-PC 모듈로 나누어지며 이들 모듈은 병렬 인터페이스를 통하여 연결된다. 사용자는 ZDS-1/40 MDS의 터미날에서 결정법칙(decision rule)을 입력하여, 이를 컴파일러에 의하여 BD머신에서 정의된 목적프로그램으로 변환하며, 변환된 BD-PC 프로그램은 BD-PC 모듈에서 요구가 있으면 병렬인터페이스를 통하여 BD머신의 메모리에 다운로딩(down loading)된다.

BD-PC 모듈에서는 MDS모듈로 부터 다운로딩된 BD-PC 프로그램이나 사용자가 직접 BD-PC모듈의 입력키(key)를 이용하여 적재한 프로그램을 낮은 속도로 운용하여 확인할 수 있고, 확인된 프로그램은 고속 클럽을 이용하여 선택입력을 입력부(input unit)로 부터 스캔하여 필요한 출력을 출력부(output input)로 출력하게 된다.

기존의 BD-PC 프로그램들의 back-up을 위하여 플로피디스크를 이용할 수 있고, MDS와 BD-PC 모듈사이에 통신채널을 구성함으로서 계층적인 시스템의 가능성을 확인하였다. 따라서, MDS에서 개발된 PID제어등 각종 콘트롤러와 BD-PC의 협동작업이 가능해졌다[4], [5], [6].

나. Binary Decision방법 및 BD-PC 모듈의 설계

(1) Binary Decision 방법

일반적으로, 조합논리와 순서논리는 각각 진리표나 스윗칭함수 또는 상태도및 천이표등에 의

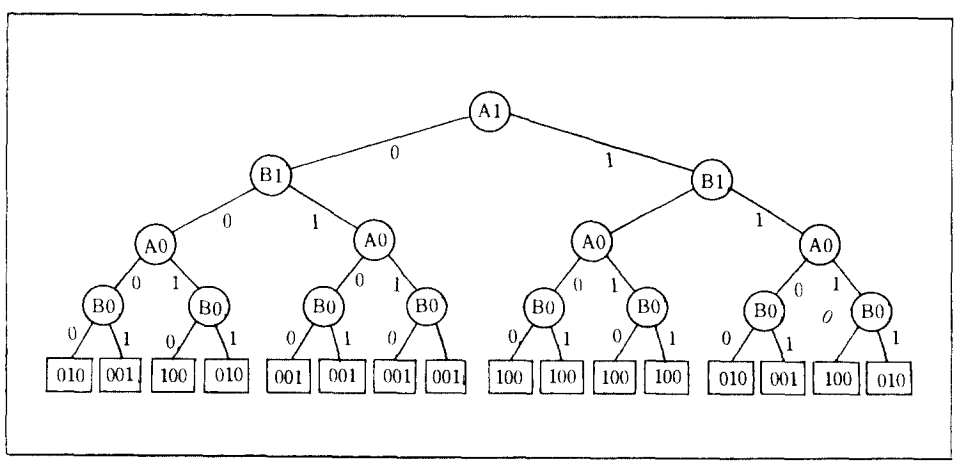

그림 2. (a) 원시 BD다이어그램
Source BD diagram.

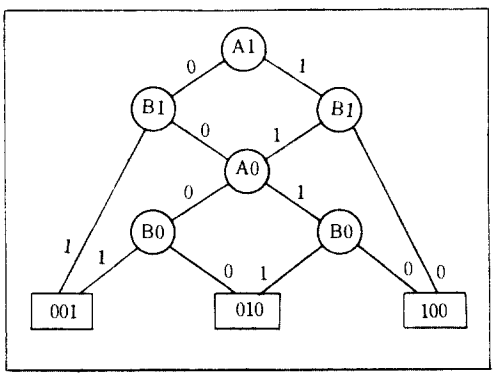

그림 2. (b)최적화된 BD다이어그램
Optimized BD diagram.

해 표현할 수 있고, BD 다이어그램이나 X : A, B와 같은 형식의 인스트럭션으로 표현할 수 있다. 여기서, X는 입력변수 A와 X의 true(1) 브랜치와 false(0) 브랜치의 어드레스를 표시한다.

BD 다이어그램이나 이진(binary) 트리는 진리표에 포함되어진 정보로부터 만들어 지는데, 이 점이 Boolean방식과는 다르다[2]. 트리는 모든 가능한 출력결과들을 보여주는데 입력변수들은 임의의 순서에 의해 선택되어질 것이다. 그러므로, 입력변수가 n 인 경우에는 같은 스윗칭 함수를 나타내는 n ! 개의 서로다른 이진 트리들이 만들어진다. 조합논리의 경우 2 -bit magnitude comparator의 예에 대한 완전한 BD다이어그램은 그림 2(a) 와 같다. 이 트리는 그림 2(b) 와 같이 동일한 출력을 내는 잉여의 브랜치를 제거한 최적화된 BD다이어그램으로 간략화될 수 있다[2].

표 1 은 BD 프로그램을 보여주고 있다.

2-bit magnitude comparator의 예에서와 같이 출력을 얻어내기 까지 필요한 결정스텝은 최대로 입력변수의 갯수를 초과하지 않으며, Boolean방식을 이용하여 2개의 입력변수들의 논리연산으로 출력을 결정하는 경우에 비해 크게 연산횟수를 줄일 수 있다.

BD 방식의 장점은 최적화된 BD트리를 형성할 수 있다는 것인데, 이것의 근간이 되는 것이 Huffman부호(code)이론이다[3], [7], [8].

Huffman 부호이론은 첫번째로 여러가지의 원시기호(source symbol)들의 발생 확률을 이용하고자 하는 것이다.

i 번째 기호의 확률이 Pi 이고 길이가 Li 라고하면 해당 부호의 평균길이는

$$Lav = \sum_{i=1}^{q} Pi\, Li$$

이다. 단, 여기서 $P1 \geq P2 \geq \ldots \geq Pq$ (내림차순)

$L1 \leq L2 \leq \ldots \leq Lq$ (오름차순)

BD스텝 수의 축소의 관점에서 볼때, 균등하게 분포된 발생 확률을 갖는 변수에 대해서는 BD 방식의 장점을 많이 이용할 수 없으며 편향(bia-

표 1 2 비트 크기 비교기의 BD프로그램
BD program of 2-bit magnitude comparator.

ADDRESS	INPUT VARIABLE	TRUE BRANCH	FALSE BRANCH
0	A_1	6	1
1	B_1	2	3
2	OUTPUT : XYZ=001	GO TO O	
3	A 0	8	4
4	B 0	2	5
5	OUTPUT : XYZ=010	GO TO O	
6	B 1	3	7
7	OUTPUT : XYZ=100	GO TO O	
8	B 0	5	7

s)된 발생 확률을 갖는 변수에 대해서 잘 이용된다[3].

(2) BD방식과 Boolean 방식의 비교

그림 3 과 같은 2개의 FORTRAN 부호로 된

```
C     BOOLEAN SEQUENCE
      LOGICAL A.B.C.D
      READ (IRDR,100) A. B. C. D
      IF(A.AND.B.AND.C.AND.D)GO TO 10
      WRITE(IPRTR,200)
      STOP
 10   WRITE(IRPTR, 300)
      STOP
100   FORMAT(4I1)
200   FORMAT(8H△, FALSE)
300   FORMAT(7H△, TRUE)
      END
```

그림 3. (a) Boolean 순서
Boolean sequence.

```
C     BINARY DECISION SEQUENCE
      LOGICAL A, B, C, D
      READ(IRDR,100) A, B, C, D
      IF(NOT. A) GO TO 10
      IF(NOT. A) GO TO 10
      IF(NOT. A) GO TO 10
      IF(NOT. A)GO TO 10
      WRITR. (IRPTR, 100)
      STOP
 10   WRITE(IRPTR, 200)
      STOP
100   FORMAT(4I1)
200   FORMAT(8H△.FAL.SE)
300   FORMAT(7H△.TRUE)
      END
```

그림 3. (b) BD 순서
BD sequence.

논리 순서를 비교해 보면, 먼저 Boolean 순서에서는 출력에 도달하기 위해서 4개의 변수 A, B, C, D가 모두 AND를 거쳐야 한다. BD 순서에서는 출력에 도달하기 위해서 많아야 4개의 변수가 실행되지만 첫번째 변수 A가 false 이면 출력은 첫번째 evaluation 후 바로 얻어진다.

이러한 근거로부터, Boolean 순서와 BD순서 사이의 정량적 비교를 할 수 있다[3].

변수 A, B, C, D가 범위 0-15까지인 균등하게 분포된 random정수라고 하면 16개의 각 상태는 똑같은 발생확률을 가지므로, 출력에 도달하는데 필요한 BD 평균 스텝수는 표 2에서 볼 수 있다.

여기서 (A.AND.B.AND.C.AND.D)가 (((A.AND.B).AND.C).AND.D)로서 얻어지고, AND

표 2 출력에 도달하는데 필요한 BD스텝수.
BD step number for output.

A	B	C	D	NO • OF EXAMINATIONS
0	0	0	0	1
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	2
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	3
1	1	0	1	
1	1	1	0	4
1	1	1	1	
				30

표 3 BD-PC 모듈의 규격
Specification of BD-PC module.

규		격	
Input Unit	동 작 방 식	sequential input	
	접 점 수	64 inputs	
	입 력 잉 태	AC/DC input	
Output Unit	동 작 방 식	parallel output	
	접 점 수	14 Latched Long/Short output	
	출 력 잉 태	AC/DC output	
Memory		256×16 bits SRAM	
Clock	Auto-Operation	FAST	1 MHz
		SLOW	0.7 Hz
	Manual Clock	가능	
Program 입 력 방 식	down load	ZDSI/40 으로 부터	
	manual load	manual switch의 조작으로 입력	
Display	input indicator	64 LEDs	
	output indicator	14 LEDs	

결정(decision)이 비교 명령어 만큼 시간을 소요한다고 가정하면, 세개의 AND 결정수를 필요로 하는 Boolean 순서는 16개의 상태에서 총 48개의 결정스텝을 요하므로, Boolean순서는 BD순서보다 평균 1.6배 만큼 더 많은 시간을 소요한다.

(3) BD방식을 이용한 BD-PC 모듈의 설계

Boute에 의해 제안된 BD 머신[9]을 입력 최대 64채널과 출력 최대 14채널을 수용하며, 256개의 인스트럭션을 수용할 수 있는 discrete IC를 이용하여, 설계 제작한 BD-PC 모듈의 규격과 블럭선도를 표 3 과 그림 4 에 각각 도시하였다.

또한 BD-PC 프로그램의 인스트럭션 포맷(format)은 그림 5 에 도시하였다[2].

그림 5 에서 비트14와 비트15는 연산부호(operation code)로서 비트15가 "0"이면, 비트 8-13에 지정된 입력변수를 선택하여 비트14와 함께 테스트되고 그 결과에 따라 다음 인스트럭션으로 브랜치한다.

또한, 비트15가 "1"이면, 출력 인스트럭션으로서 비트14에 따라 비트 0-13의 내용을 출력하거나 비트 8-13을 출력하고 비트 0-7로 무조건 브랜치 할 것인가를 결정한다.

또한, 그림 4 에서 데이터 스윗치는 BD-PC 프로그램을 메모리에 입력시키기 위해 설정하였으며 제어 스윗치들은 키 조작에 의해 메뉴얼(manual)클럭을 발생시키거나 클럭의 스피드를 1 MHz 또는 0.7Hz로 선택하는 등의 사용자의 편

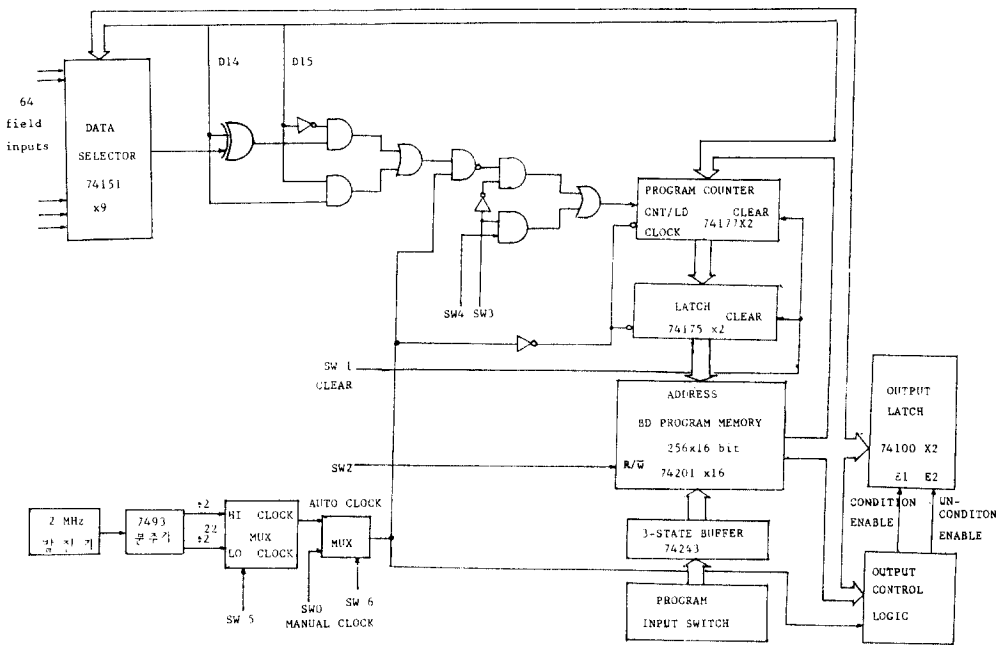

그림 4. BD-PC 모듈의 블럭선도.
Block diagram of BD-PC module.
SW 1 : clear, SW 2 : Read/Write, SW 3 : Auto/manual, Count/load,
SW 4 : BD-PC clock/Computer clock 선택,
SW 5 : High/Low clock 선택, SW 6 : Auto/manual clock 선택)

15	14	13 ... 8	7 ... 0
0	v	INPUT VARIABLE ADDR	NEXT INSTRUCTION ADDR

v⊕ INPUT VARIABLE = 1 : NEXT INSTRUCTION SHOWN BY 0-7.
0 : NEXT INSTRUCTION IN NEXTLOCATION.

15	14	13 ... 8	7 ... 0
1	v	OUTPUT TALUES	OUTPUT VALUES/ NEXT INSTRUCTION ADDR.

v = 1 : NEXT INSTRUCTION SHOWN BY 0 - 7.
OUTPUT VALUES : BITS 8 - 13.
0 : NEXT INSTRUCTION IN NEXT LOCATION
OUTPUT VALUES: BITS 0 - 13.

그림 5 BD 프로그램의 instruction format.
Instruction format of BD program.

의를 위해 설정하였다.

사진 1은 2-bit magnitude comparator의 BD-PC 프로그램을 구동하고 입력으로 서로 90°의 위상차를 갖는 4개의 펄스를 입력하여 얻어낸 출력을 보여 주고 있다. A1, B1, A0, B0 는 구형파 입력이며, X, Y, Z는 출력변수이다.

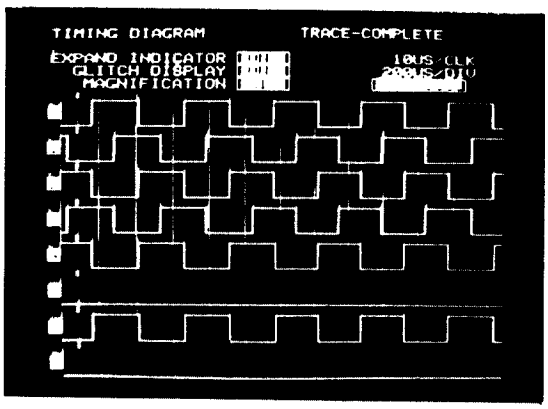

사진 1. 2 비트크기 비교기의 예
Example of 2-bit magnitude comparator.

다. 인터페이스 장치

인터페이스 장치는 ZDS-1/40 MDS의 RAM 4000-41FF H에 저장된 BD-PC 목적프로그램을 BD-PC 모듈의 RAM (256×16 bits)에 다운로딩하기 위한 장치이다. BD-PC 목적프로그램의 인스트럭션은 16bits이므로 MDS의 2 bytes가 하나의 BD-PC 인스트럭션에 해당된다. BD-PC 모듈에서 프로그램카운터를 클리어(clear)하고 DOWN LOADING 요구신호를 보내면 PIB (Parallel Interface Board) 내의 PIO (A)를통해 Z-80 CPU의 MODE II 인터럽트를 요구하게 되며, 이때 프로그램카운터의 클럭은 컴퓨터에서 만들어지는 클럭으로 스위치된다. Z-80 CPU는 인터럽트 요구에 따라 그림 7의 인터럽트 서비스루틴을 구동하여 다운로딩을 수행하고 완료시에는 완료신호를 PC 모듈에 보낸다. 그림 6에는 인터페이스 장치의 회로도를 도시하였고, 그림 7에는 인터럽트 서비스루틴의 순서도를 도시하였다.

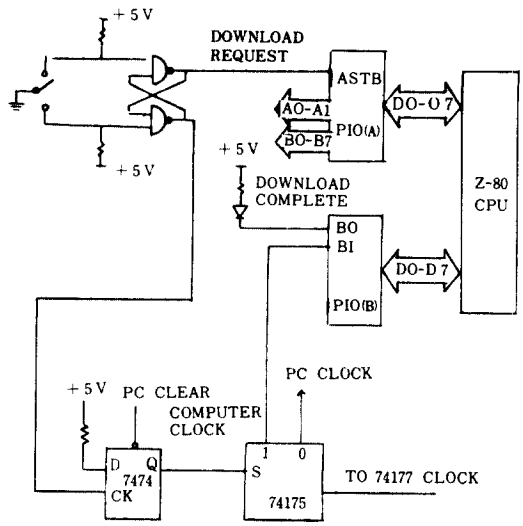

그림 6. 인터페이스 장치의 회로도
Circuit diagram of iterface unit.

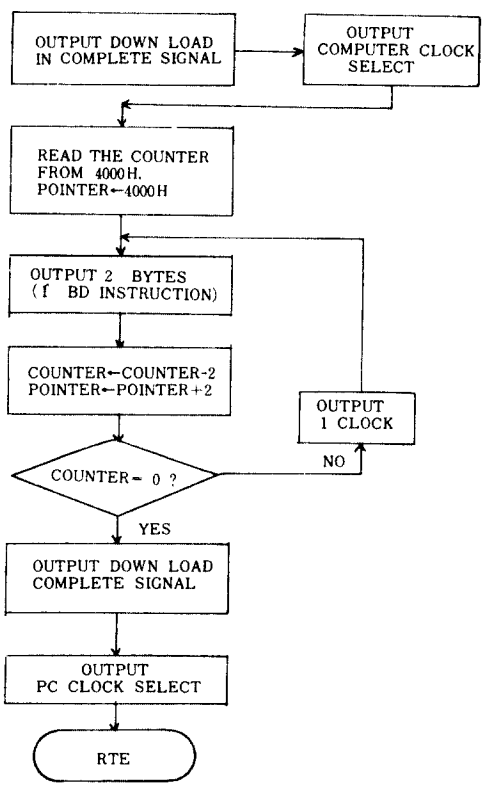

그림 7. 인터럽트 서비스루틴의 순서도
Flow chart of interrupt service routine.

라. BD-PC 프로그램 컴파일러의 설계

진리표로부터 입력된 결정법칙을 그대로 BD-PC 프로그램으로 변환한다면, 입력수가 증가함에 따라 식(1)과 같이 결정 node가 기하급수적으로 많아지게 되며, 그 만큼 프로그램을 저장하기 위한 메모리도 많이 필요로하게 된다. 따라서 최소한의 메모리를 차지하도록 트리를 재구성하여 최적화된 트리를 얻어내야 한다.[3] 그림 8은 입력된 결정법칙으로부터 컴파일러를 구동하여 BD-PC 프로그램을 얻어내는 과정을 도시하였다.

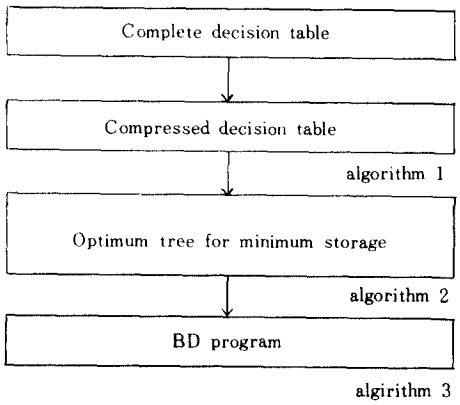

그림 8. BD-PC 컴파일러의 블럭 다이어그램
Block diagram of BD-PC compiler.

첫번째 과정에서는, 진리표로부터 입력과 출력의 비트수를 읽어들이며 해당 출력벡터에 따른 모든 입력벡터를 읽어들일 때까지 과정을 반복한다. 두번째 과정은, 결정표를 압축해가는 과정으로 Quine-Macclusky의 tabulation 방법을 이용하여 하나의 출력벡터에 대한 입력벡터들의 essential prime implicant를 찾아내고 minimum cost를 갖는 해를 채택한다.

최소크기의 메모리를 갖는 최적트리를 얻어내기 위해서 pollack의 알고리즘을 채택하였다.[3],[10] 그림 9 에는 이 알고리즘으로 부터 트리의 root에 해당하는 node를 찾아내는 방법을 순서도를 이용하여 도시하였다.

압축된 결정표에서 CCj (Colume Count), DCi (Dash Count), Di(Delta Count) 등을 계산한 다음 DCi가 가장 작은 값을 가진 입력변수를 제일 상위의 결정 node로 한다.DCi가 동일한 것이

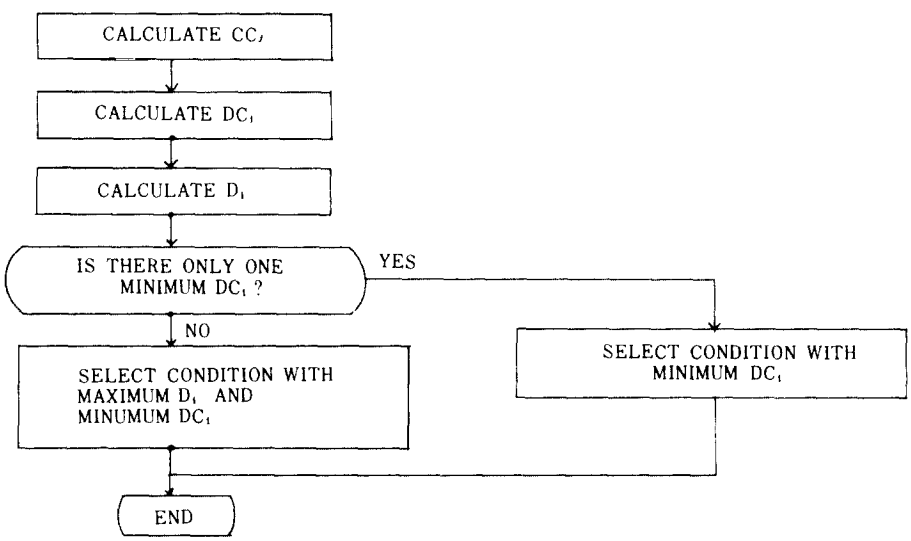

그림 9. node를 찾기 위한 순서도
Flow chart for a node.

하나 이상 있을 경우에는 Di가 가장 큰 값을 갖는 입력변수를 선택한다. 여기서, 출력 Yi를 위한[3] K번째 열의 CC_{jk} 는

$$CC_{jk} = 2^{d} \text{(단, d는 j번째행의 dash 수)} \quad (2)$$

이며 i번째 행의 DCi는,

$$DCi = \sum_{k} CC_{jk} \; A_{jk} \quad (3)$$

이다. 여기서, A_{jk} 는

A_{jk} = 0 (Yi를 위한 k번째행에 있어서 Xi∈{0, 1}인 경우)
= 1 (Xi가 don't care인 경우) (4)

이다. 또한 i번째행의 Di는,

0	0	0	0	0	0	0	1
0	1	0	0	0	0	1	1
1	0	0	0	0	1	1	0
0	0	0	0	0	0	1	1
0	1	0	0	0	1	1	0
1	0	0	0	1	0	0	1
0	0	0	0	0	0	1	1
0	1	0	0	1	0	0	1
1	0	0	0	0	0	1	1
0	0	0	0	0	0	1	0
0	1	0	0	1	0	0	1
1	0	0	0	1	1	0	0
1	1	0	0	0	0	0	1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

0	0	0	0	0	1	0	0
0	1	0	0	1	0	0	1
1	0	0	0	0	0	1	1
1	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

b_7	b_8	CONTENTS
0	0	INPUT VARIABLE
0	1	0 BRANCH
1	0	1 BRANCH
1	1	OUTPUT VARIABLE

그림10. 2비트 크기 비교기의 브랜치 테이블
Branch table of 2-bit magnitude comparator.

$$Di = \sum_{K} CC_{ik} \ \beta_{jk} \qquad (5)$$

이며, 여기서 β_{jk} 는,

$$\begin{aligned}\beta_{jk} &= 0 \text{ (k 번째 열의 Xi = 1 인 경우)} \\ &= 1 \text{ (Xi = don't care 인 경우)} \\ &= -1 \text{ (Xi = 0 인 경우)}\end{aligned} \qquad (6)$$

과 같다.

그림 8 에 도시된 방법을 이용하여 root-node를 찾은 뒤, "0"또는 "1" 브랜치에 해당하는 subtree를 만들어 내고, 같은 방법으로 순차적으로 결정node 들을 찾아내어 최후에 출력node들만 남을 때까지 계속하며 그 결과로 모든 연결관계가 기록된 테이블을 얻어낸다.

그림 10 에는 2-bit magnitude comparator의 예에 대한 브랜치 테이블을 도시하였고, 여기서 비트 7, 비트 6은 입력과 출력변수의 번호를 구별하고, "0" 브랜치, "1"브랜치의 오프셋(offset)값을 구별하기 위해 사용하였다. 또한, 출력변수의 브랜치 오프셋은 0으로 만들었다.

BD-PC 프로그램 알고리즘에서는, 알고리즘 2에서 얻어진 테이블을 이용하여 연속적인 메모리 공간에 BD-PC 인스트럭션 포멧에 맞도록 엔코드하는 작업이다[3]. 임의의 입력변수에 대한 "0" 브랜치 또는 "1" 브랜치 중, 반드시 하나는 연속적인 메모리를 가져야 하는 인스트럭션상의 제약때문에 일단 root node에서 시작하여 "0" 브랜치 node만 연속적으로 찾아가며 입력 인스트럭션에 엔코드하여 연속적인 메모리를 차지하게 하며 이때 "1" 브랜치 어드레스는 비워둔다. 이런 일련의 탐색과정은 출력 node를 만나면 출력 인스트럭션을 엔코드하고, 이 출력 인스트럭션의 다음 어드레스는 프로그램의 맨 처음이 된다.

다음에는 root node의 하위레벨에 있는 고려되지 않는 "1" 브랜치 node를 알고리즘 2에서 얻어진 테이블에서 찾아내어 엔코드하며, 이 때 엔코드된 인스트럭션에 점유되는 메모리의 어드레스를 비워 두었던 root node의 "1" 브랜치 어드레스 필드에 기입하고, 계속하여 해당 node로부터 "0" 브랜치 node를 찾아 엔코드한다. 상기한 과정을 모든 node들이 엔코드될 때까지 계속하여 컴파일을 끝내게 된다. 그림 11에는 2-bit magnitude comparator의 예에서 상기한 방법으로 엔코드되는 순서를 도시하였다.

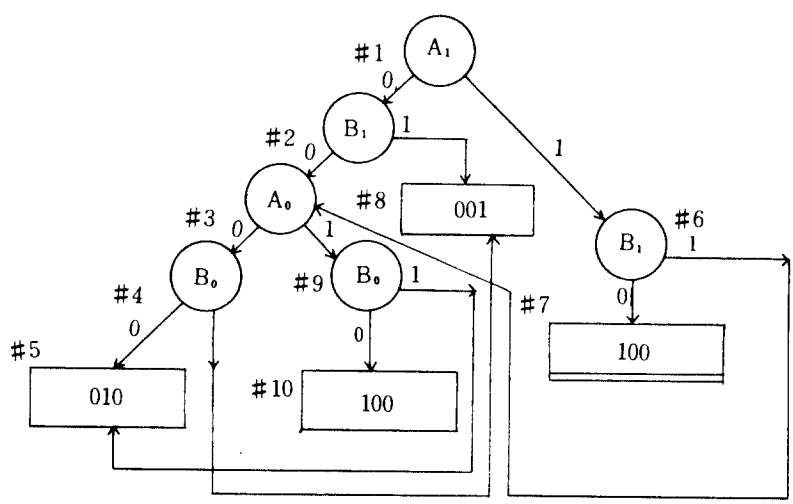

그림11 2 비트 크기 비교기의 BD트리 다이어그램
BD tree diagram of 2-bit magnitude comparator.
#1, #2, ···, #9, #10 node 탐색순서.

Ⅲ. 결 론

종래의 Boolean 방식을 PC에 적용할 경우 입력수의 증가에 따라 스캔 스피드가 기하급수적으로 늘어나는 단점이 따르게 된다. 따라서, 이러한 단점을 개선하기 위해 결정단계수가 최대로 입력수를 넘지 않는, BD방식을 적용한 BD-PC 모듈을 표 2의 규격과 같이 설계 제작하여 스캔 스피드를 개선하였다.

또한, 진리표로부터 입력된 결정법칙을 압축하여 최소크기 메모리를 갖는 최적트리를 만들고, 이 트리로부터 BD-PC 프로그램을 얻어내기 위한 컴파일러를 Z-80 어셈블리어로 개발하였다. 개발된 BD-PC 모듈을 병렬 인터페이스 장치를 통하여 ZDS-1/40 MDS에 연결하여 계층적인 BD-PC 시스템의 가능성을 보였다.

앞으로의 과제는 분산제어 시스템에 적합한 OS 개발과 산업용 제어에서 적용할 수 있는 프로그램 개발등이 있고, MDS를 host로 하는 multi BD-PC 시스템도 연구 개발해야 할 과제이다.

본 연구는 '85년도 전반기 한국과학재단 지원으로 이루어졌음.

參 考 文 獻

(1) 권욱현, 김원철"프로그램형 제어기의 기술현황," 전기학회지. pp. 474-485, vol. 32 no. 8, 1983.

(2) P. J. Zsomber-Murray, L. J. Vroomen, R-Hudson, Le-Ngoc and P. Holck "A Binary-Decision Based Programmable Controller." IEEE Micro, August, October, December. 1983.

(3) J. L. van Driel "Binary Decision Compiler Algorithms.", Dept. of Mechanical Engineering. Sep. 1. 1982.

(4) 김원철, 변대규, 권욱현"프로그래머블 콘트롤러의 설계및 제작에 관한 연구." 대한전자공학회. vol 6-2, 238-241, 1983.

(5) 김용수, 김영헌, 최영규, 유준, 변증남 "프로그래머블 콘트롤러의 개발에 관한 연구." 대한전자공학회. 추계 종합학술대회 논문집. vol. 5-2, 278-281, 1982.

(6) 윤재우 "적응제어이론과 마이크로 프로세서를 이용한 온도 제어에 관한 연구." 전북대학교 석사학위 논문. Feb. 1985.

(7) D. A. Huffman "A Method for the Construction of Minium Redundancy Codes." Proc. IRE 40, vol 10, pp. 1098-1108, Oct. 1952.

(8) Richard W. Hamming "Coding and Information Theory." Tower Press, pp. 64-68.

(9) R. T. Boute "The Binary-Decisision Machine as programmable Controller." Euromicro News-Letter, vol 1, no. 2, pp. 16-22, 1976.

(10) S. L. Pollak "Conversion of Limited Entry Decision Tables to Computer Programs." Comm. ACM, vol 8, Nov. 1965.

(11) S. B. Akers "Binary-Decision Diagrams." IEEE Trans. Computers, C-27, no. 6, 1978.

(12) F. Cerny, D. Mange, and E. Sabcgez "Multiplexers and Binary-Decision Trees." IEEE Trans. Computers, C-28, 1979.

(13) Eric J. Lerner "Computer Aided Manufacturing." IEEE Spectrum, Nov. 1981.

(14) R. Hudson, L. J. Vroomen, P. J. Zsombor-Murray and T. Lengoc "A Hybrid, Binary-Decision/Microprocessor Programmable Controller." Second Int'l Symp. on Mini and Microcomputers, Fort Lauderdate, Fla., pp. 60-65, Dec. 1979.

(15) R. Hudson "A MC 6809 Based Optimizing Compiler for Binary Decision Programs." Dept. of Mechanical Engineering, August. 21. 1981.

(16) 노창주 "해설 Sequence 제어." 아성출판사. 1977.

(17) 대한전기협회 출판부 "프로그래머블 콘트롤러를 활용하는 Sequence 제어." 대한전기협회. 1984.

(18) 전병실 외 4인 "Binary-Decision Based Programmable Controller에 대한 연구." 대한전자공학회, 하계종합학술대회 논문집, vol 1, no. 1, July. 1984.

田 炳 實(Byoung Sil CHON) 正會員
1945年 2月 14日生
1967年 2月 : 全北大學校 工科大學 電氣工學科 卒業
1969年 2月 : 全北大學校 大學院 碩士過程(1970. 8. 工學碩士)
1974年 2月 : 全北大學校 大學院 博士過程(1976. 8 工學博士)
1971年 3月~1987現在 : 全北大學校 工科大學電子工學科 教授
1979~1980年 : 美國 Univ. of Notre Dame 객원교수

李 俊 煥(Joon Whoan LEE) 正會員
1957年 7月 26日生
1980年 2月 : 漢陽大學校 電子工學科 卒業
1982年 2月 : 韓國科學技術院 電氣및電子工學科 卒業(工學碩士)
1982年 3月 : 全北大學校 工科大學 電子工學科 助教
1985年 4月~1987年 現在 : 全北大學校 工科大學 電子工學科 專任講師

嚴 景 培(Kyoung Bae EUM) 正會員
1956年 9 月10日生
1984年 2 月：全北大學校 工科大學 電子工學科 卒業
1986年 2 月：全北大學校 大學院 電子工學科 卒業(工學碩士)
1986年 3 月～1987年現在：全北大學校 大學院 博士過程 在學
1985年 5 月～1987年現在：全北大學校 工科大學 電子工學科 助教