論文 84-21-6-8

GRAFCET로 記述된 順序論理 시스템의 Race없는 프로그램으로써의 合成

(Race-Free Programmable Synthesis of a Sequential System Described by a GRAFCET)

禹 廣 俊*

(Kwang Joon Woo)

要 約

本 論文에서는 GRAFCET로 記述된 竝列 sequence를 갖는 順序 論理 시스템을 programmed logic에 依해 實現하는 方法을 提案한다. 이를 위해 먼저 竝列 sequence를 갖는 GRAFCET를 그 本來의 物理的 意味를 변형치 않고 多數의 state graph로 재구성하는 알고리즘을 提案했으며 이 提案된 알고리즘은 모든 형태의 GRAFCET에 適用 可能하며 특히 sub-system으로의 분해 및 그 역과정을 손쉽게 한다. 다음 이와같이 재구성된 state graph를 ROM을 使用한 microprogrammed logic에 依해 構成하는 조직적인 方法을 提示 했으며 이 提示된 構成 方法은 選擇 sequence의 數를 任意로 확장할 수 있다.

Abstract

This paper proposes a programmed logic realization of sequential logic system with parallel sequences which is described by a GRAFCET.

For this purpose, an algorithm is proposed, which decomposes the GRAFCET with parallel sequence into a set of state graph without changing the physical meaning, which is applied to all kinds of GRAFCET, and which devides the system into sub-systems and vice versa. A systematic implementation by microprogrammed logic using ROM is proposed, which expands the number of selection sequence.

I. 序 論

順序 論理 시스템의 合成은 specification에 맞게 일정한 형식에 따라 記述(description)한후 이 記述된 言語 혹은 그라프등으로 부터 構成(materialization; implementation)과정을 거쳐 行해진다. 일반적인 論理 시스템 뿐만 아니라 industrial sequential automation과 같이 多數(수십~수천)의 入出力 變數 및 內部狀態 變數가 要求되는 論理 시스템을 合成하는데는; -願하는 機能을 理解하기 쉽고 명확 간결하게 記述할 수 있어야 하며, -記述된 結果로부터 直接 構成할 수 있는 점이 要求된다.

그러나 古典的인 記述 方法(例로써 Huffman의 方法, flowchart 혹은 timing diagram)으로써는 시스템이 복잡해 질수록 명확하게 記述 할 수 없을 뿐만 아니라,

*正會員, 檀國大學校 理工大學 電子工學科

(Dept. of Electronic Eng., Dan Kook Univ.)

接受日字 : 1984年 8月 9日

(※ 本 論文은 1984年度 文教部 學術研究助成費에 依하여 研究되었음)

記述된 結果로부터 直接 構成하기가 不可能하다. 따라서 industrial sequential automation과 같은 대규모 시스템을 記述하는데 절대적으로 必要한 機能의 同時性까지도 記述할 수 있는 graph에 依한 方法(GRAFCET[1])이 1977년 AFCET[2]에 依해 개발되었다.

이후 GRAFCET의 구조적인 硏究[13 14], 다양한 industrial automation에의 적용에 관한 硏究[10 12] 및 GRAFCET에 依해 記述된 順序 論理 시스템의 構成에 관한 硏究[1-2][10 13]가 중점적으로 이루어져 왔다. 특히 GRAFCET는 시스템의 효율성 증대 및 sub-system으로 처리가 可能한 sequence의 同時性(竝例 sequence)의 記述을 可能하게 함에 따라, 이러한 竝列 sequence를 갖는 GRAFCET의 構成 方法이 硏究되어 왔다. 順序 論理 시스템의 構成 方法은 wired logic과 이 보다 flexibility가 좋은 programmed logic에 依한 方法이 있으며, programmed logic 構成에는 ; -ROM을 使用한 microprogrammed logic과 -기존의 microprocessor를 利用하는 方法등이 있다. 따라서 GRAFCET로 記述된 順序 論理 시스템을 특히 programmed logic에 依한 構成 方法이 중점적으로 硏究되어 왔다.[1-9]

특히 근년에는 IC 技術의 급격한 발달로 인해 全體合成 費用중 material이 차지하는 費用의 相對的인 감소로 인해, 使用 素子數의 最小化에 관한 硏究등으로부터 -記述된 結果로 부터 直接 構成이 可能하고- maintenance가 용이한 構成 方法에 관한 硏究가 관심의 대상이 되었으며, 이러한 추세는 시스템이 복잡해질수록 더욱 절실히 요구되는 과제이다.

따라서 본 硏究에서는 竝列 sequence를 갖는 GRAFCET를 programmed logic에 依하여 構成하는 方法을 提示했다. 즉 竝列 sequence를 갖는 GRAFCET를 多數의 state graph들로 쉽게 재구성하는 알고리즘을 提案했으며, 이들 state graph를 microprogrammed logic에 依하여 直接 構成하는 方法을 提示했다. 이러한 合成 方法은 sub-system으로 나누어 처리할때 要求되는 sub-system 各各의 獨立性의 보장 및 sub-system간의 synchronization 문제를 해결한다.

Ⅱ. GRAFCET의 基本 槪念

基本的인 定義 및 實例로써 industrial sequential automation을 GRAFCET로 記述하는 方法을 설명한다. 이에 관한 구체적인 硏究는 참고문헌 [11-14]에서 찾아 볼 수 있다.

1. 定義

GRAFCET란 sequential automation의 서로 다른 action을 그라프적으로 記述한 function diagram 으로서, step(정방형으로 표시), transition(bar로 표시)과 이들을 연결시키는 oriented arc로써 구성된다(그림1). 이러한 graph가 sequential automation을 記述할 수 있도록 step은 수행할 일련의 action에 관계되며, transition은 다음 step으로의 evolution을 可能케 하는 logic condition에 관계된다.

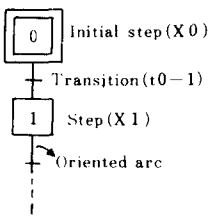


그림 1. GRAFCET의 구조
Fig. 1. Structure of GRAFCET.

Sequential automation을 記述한 GRAFCET는 다음과 같은 evolution 規則을 따른다:

- Initialization은 시스템 최초의 action을 명시한 active step으로서 二重 정방형으로 표시한다(그림1 참조).
- Transition은 直前의 step들이 active되면 有效해진다. Transition이 有效하고 그리고 이 transition에 관계된 logic condition이 眞이 되면 해당 transition을 뛰어 넘는다.
- 뛰어넘은 結果로는 해당 transition 直前의 step들이 inactivation되고, 直後의 step들이 activation된다.
- 同時에 뛰어 넘음이 가능한 多數의 transition 들은 同時에 뛰어 넘는다.
- 同一한 step이 activation과 inactivation이 同時에 이루어질 경우 이 step은 activation되어 머문다.

이와 같이 sequential automation을 GRAFCET로 記述할 경우 하나의 sequence(unique sequence)만으로 記述되는 경우, logic condition에 따라 各各의 sequence를 追求하는 選擇 sequence를 포함하는 경우, 그리고 同時에 2개 이상의 sequence를 追求하는 竝列 sequence을 포함하는 경우가 있다. 특히 竝列 sequence는 獨立的인 sub-system으로 분해 할 수 있는

1. GRAFCET : Le Graphe de Commande Etape-Transition
2. AFCET : L'Association Francaise pour la Cybernetique Economique et Technique

모든 system에 대해 항상 존재하며 이렇게 竝列 sequence로 記述함으로서 system의 효율성을 높인다.

2. 例題[13]

다음과 같은 specification을 갖는 drilling machine을 GRAFCET에 依해 記述한다.

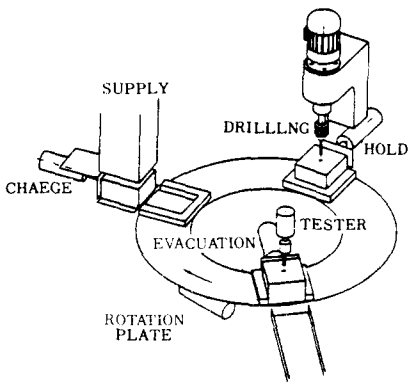


그림 2. 일련 作業의 drilling machine
Fig. 2. Post of drilling machine.

1) Specification

그림 2와 같이 3곳의 作業台을 갖는 回轉 plate의 첫번째 作業台에서는 加工할 물건을 채우고, 두번째에서는 drilling을 하고 세번째에서는 加工된 물건을 검사한 후 비운다. 하나의 piston이 作業台을 받치는 plate를 外部에서 120°씩 정확하게 回轉 시킨다. 검사는 하나의 tester에 의해 行해지며 정확하게 drilling됐으면 이 tester가 low position까지 내려간다. 만일 low position까지 내려가지 못하면 모든 시스템은 정지되며, 이때 tester는 high position에 머문다. 이 사이에 作業者는 不良品을 꺼내고 다시 手動으로 시스템을 재가동시킨다.

2) Function GRAFCET의 구성

前述의 specification을 GRAFCET로 記述하면 그림 3과 같다.

그림 3에서 使用한 actions 및 logic conditions의 심볼 및 意味는 표 1과 같다.

표 1. Actions과 logic conditions의 심볼 및 그 意味
Table 1. Natures and symbols of actions and logic conditions.

Actions

Push charger	PUC	Double pilot piston
Pull charger	PLC	
Push holder	PUH	〃
Pull holder	PLH	
Descend drilling	DED	〃
Raise drilling	RAD	
Descend tester	DET	〃
Raise tester	RAT	
Push evacuator	PUE	〃
Pull evacuator	PLE	
Rotation plate	ROP	simple pilot piston

Informations

Piece charged	PIC	End of course
Charger in back	CHB	〃
Piece holded	PIH	〃
Piece released	PIR	〃
Drilling in low	DRL	〃
Drilling in high	DRH	〃
Tester in low	TEL	〃
Tester in high	TEH	〃
Piece evacuated	PIE	〃
Evacuator pulled	EVP	〃
End of rotation	ENR	〃
Start	ST	Push botton
Restart	RE	〃

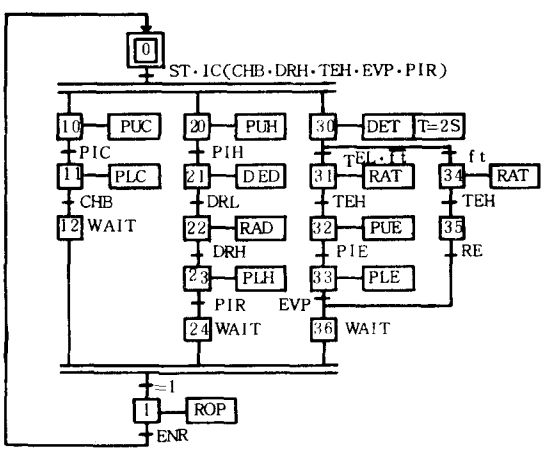


그림 3. Drilling machine의 GRAFCET
Fig. 3. GRAFCET of drilling machine.

"Start"라는 命令과 operation part가 正位置에 있다는 initial condition(IC)의 information에 따라 step 10, 20, 그리고 30이 同時에 activation된다. 즉 竝列 sequence의 synchronization이 이루어진다. 이러한 situation으로 부터 sequence 10-12, 20-24, 그리고 30-36은 各各 獨立的으로 evolution되며, 이러한 3개의 獨立된 sequence가 step 1을 갖는 共通 sequence

로 同時에 evolution되어야 한다. 즉 step12, 24, 그리고 36 모두가 同時에 active되고, 이들 step 直後의 logic condition이 眞이 되어야 한다. 이를 위해 always true를 意味하는 logic condition "=1"를 使用하며, 또한 竝列 sequence들의 마지막 step들 12, 24, 그리고 36은 어떠한 action도 이루어지지 않고 다만 이 竝列 sequence의 re-synchronization 역활만을 하며, 이들을 "WAIT STEP"이라 한다.

Transition t 30-31의 logic condition은 tester가 low position에 있다는 것과 step 30이 activation된 후 2 seconds가 경과하지 않았다는 점을 고려했으며, transition t 30-34의 logic condition과는 security를 보장하기 위해 exclusive시켜 $\overline{ft}$와 ft를 使用했다.

Ⅲ. 竝列 Sequence를 갖는 GRAFCET의 State Graph로의 재구성

1. 재구성 알고리즘

[定義 1] State graph(SG)란 어느 한 時點에 하나의 step만이 sctive되는 GRAFCET 즉, 竝列 sequence를 포함하지 않는 GRAFCET를 말한다.

[定義 2] 竝列 sequence를 구성하는 sequence 들을 sub-sequences라 하며, initial step부터 竝列 sequence 直前까지를 pre-common sequence, 竝列 sequence 直後부터를 post-common sequence라 말한다. 또한 竝列 sequence 直前의 logic condition을 특히 synchronization logic condition(SLC), 竝列 sequence 直後의 logic condition을 re-synchronization logic condition (RLC)이라 한다(그림 4).

Ⅱ-2의 例題에서 본바와 같이 竝列 sequence를 使用함으로서 竝列 sequence를 구성하는 多數의 sub-sequence의 獨立性 및 이들 相互間의 同時性(synchronization및 re-synchronization)을 보장한다. 따라서 竝列 sequence를 포함한 GRAFCET를 多數의 SG로 재구성하려면, 竝列 sequence를 구성하는 多數의 sub-sequence의 獨立性 및 이들 사이의 同時性을 보장해야 한다. 따라서 竝列 sequence를 포함하는 GRAFCET를 多數의 SG로 재구성하는 알고리즘의 節次는 다음과 같다.

(節次 1) Sub-sequence 1 및 共通 sequence로써 순서에 변화없이 SG 1을 구성한다(그림 5 (a) : 그림 4를 例로함). 이와 같이 함으로서 sequence 1의 獨立性을 보장한 SG 1이 구성되며 이 SG 1의 action 및 logic condition의 內容에는 原 GRAFCET와 변화가 없다. 다만 sub-sequence간의 同時性 즉 SG 간의 同時性을 보장하기 위한 SLC와 RLC의 內容은 節次 3에 의한다.

(節次 2) Sub-sequence 2 및 pre-common sequence에 대응하는 initial step(synchronization memory step이라 함)과 post-common sequence에 대응하는 또 다른 step(re-synchronization memory step이라 함)으로 獨立性이 보장된 SG 2를 구성한다(그림 5 (b) : 그림 4를 例로 함). 이때 SG간의 同時性을 보장하기 위한 SLC 및 RLC의 內容은 節次 3에 의한다. 이와 같이 竝列 sequence를 구성하는 모든 sub-sequence에 대해 各各 SG를 구성한다. 但 re-syn·mem·step은 경우에 따라 생략할 수 있다.

(節次 3) 이와같이 구성된 SG간의 同時性을 보장하기 위해서 SLC에 이 transition 直前의 step들을 相互 logic AND 시킨다(그림 6). 또한 RLC에도 이 transition 直前의 step들을 相互 logic AND 시킨다(그림7).

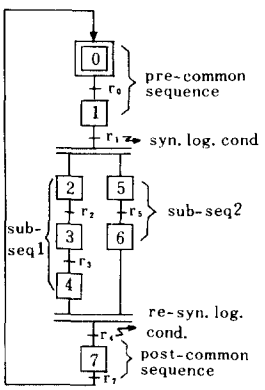
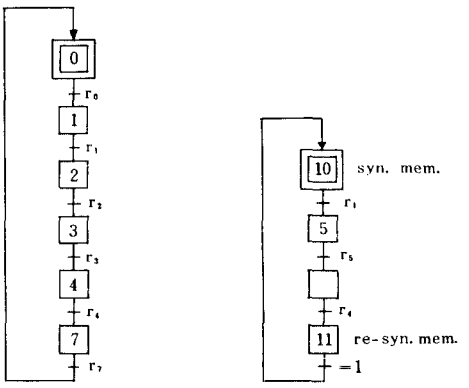


그림 4. 竝列 sequence를 갖는 GRAFCET의 例
Fig. 4. Exemple of GRAFCET with parallel sequence.



(a) SG 1 (b) SG 2

그림 5. 그림 4에 대한 state graphs의 재구성
Fig. 5. Construction of state graphs for Fig. 4.

但 共通 sequence를 포함하는 SG1의 SLC로 logic AND되는 step은 생략 할 수 있다.

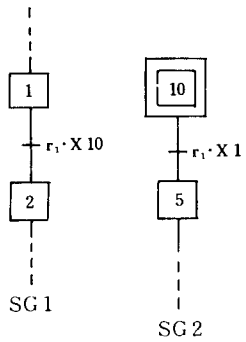


그림6. 그림5에 대한 state graph간의 synchronization
Fig. 6. Synchronization of state graphs for Fig.5.

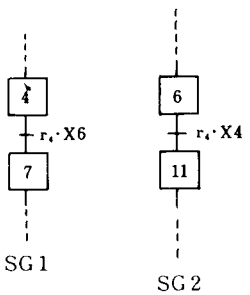


그림7. 그림5에 대한 state graph간의 re-synchronization
Fig. 7. Re-synchronization of state graphs for Fig. 5.

이와같이 獨立性을 보장하기 위해 竝列 sequence內의 sub-sequence들을 各各 갖는 SG들을 구성한 후 이 재구성된 SG사이의 同時性을 보장하기 위해 crossed step을 logic condition으로 使用한다. 따라서 이러한 알고리즘은 모든 형태의 GRAFCET에 적용 가능하며 原 GRAFCET가 갖는 物理的 意味를 保存함으로서 maintenance등에 용이하다. 또한 이 알고리즘은 sub-system으로 나누거나 sub-system을 합성하는 方法으로 유용하게 使用 될 수 있다.

2. 알고리즘 적용 例

Ⅱ-2에 記述된 GRAFCET를 上記 알고리즘을 적용하여 3개의 SG로 재구성하면 그림8과 같다.

Ⅵ. State Graph의 Programmed Logic에 依한 構成

竝列 sequence를 갖는 GRAFCET를 前述한 알고리즘에 따라 SG들로 재구성함으로서 竝列 sequence의 programmed logic에 의한 構成時 제기되는 program race 문제를 해결 할 수 있으며, 竝列 sequence를 포함하는 GRAFCET의 構成은 단지 SG의 構成 문제로 귀착된다. 따라서 本 chapter에서는 SG를 ROM을 使用한 microprogrammed logic에 의한 構成 方法을 提示한다.

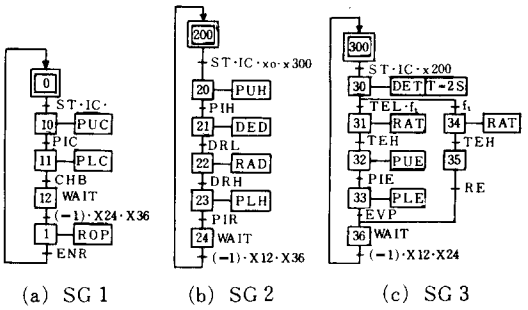


그림8. State graph로의 재구성
Fig. 8. Construction of state graphs.

1. Microprogrammed logic의 Hardware 구조

SG는 어느 한 時點에 하나의 step만이 active 되므로 이러한 active된 step은 automation에서 present state에 對應한다. 따라서 이와 같은 SG를 microprogrammed logic에 依한 構成時 그 hardware 구조는:

- Present state의 code를 나타내는 register
- Action에 對應하는 output와 next state의 address 등으로 구성된 micro-instruction을 저장하는 ROM
- Present state에서 next state로 상태 천이를 위한 logic condition의 출현을 감지하는 logic condition matrix(LCM)
- 이상의 block간의 information 교환을 보장하며 選擇 sequence를 포함하는 경우 정확한 next state

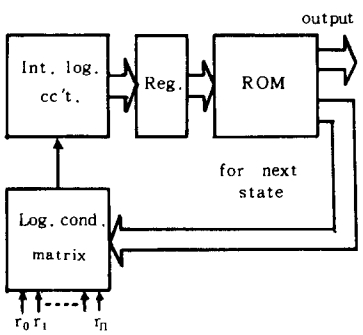


그림9. Microprogrammed logic 構成의 block diagram
Fig. 9. Block diagram of microprogrammed logic implementation.

를 결정시켜주는 internal logic circuit

以上의 block들로 구성된 block diagram은 그림9와 같다.

以上의 構成에서 고려해야 할 점은 選擇 sequence를 처리하는 데 있다. 즉 選擇 sequence의 경우(그림10 참조) step N의 next step은 step(N+1) 혹은 step(N+i)로서 해당 logic condition(r_n 혹은 r_{n+i})에 따라 그 sequence가 정해진다. 따라서 logic condition r_n과 r_{n+i}는 서로 다른 LCM에 各各 入力되며, step(N+1)이나 step(N+i)를 選擇하기 위해서 이들 各各의 step의 next step으로 指定되도록 서로 다른 增分을 지정하는 값이 各各의 LCM의 出力과 各各 AND gate로 연결됨으로서 r_n 혹은 r_{n+i}에 따라 next step이 선택된다. 따라서 以上에서 論한 unique sequence 및 選擇 sequence를 처리 할 수 있는 hardware구조를 그림11에 나타낸다. 이때 logic condition r_n과 r_{n+i}는 各各의 LCM의 同一한 번호에 入力되어야 한다. 또한 同一한 logic condition이 選擇 sequence를 구성하는 各各의 sequence에 존재할 경우는 各各의 LCM의 서로 다른 번호에 入力되어야 한다.

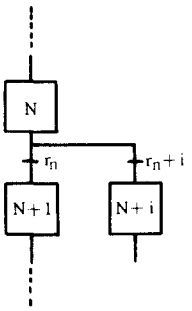


그림10. 選擇 sequence가 2인 경우
Fig. 10. Case of two selection sequences.

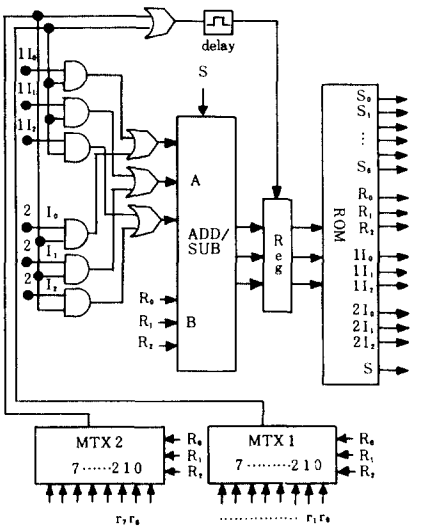


그림11. Microprogrammed logic의 hardware 구조 (선택 sequence가 2인 경우)
Fig. 11. Hardware organization of microprogrammed logic (case of two selection sequence).

2, Microprogram의 작성

그림12와 같이 GRAFCET의 step number를 memory address와 一致시키면, 이 address가 지정될때 즉 stepN이 present state로 되면 그 內容인 micro-instruction은 present action을 나타내는 micro-action field($S_0S_1\cdots S_b$: 그림11)와 next state인 step(N+1) 혹은 step(N+i)를 지정하는 field 즉 next state (next step)로의 천이 조건인 logic condition r_n(혹은 r_{n+i})의 LCM에서의 入力번호를 나타내는 logic condition field($R_0R_1R_2$: 그림11)와 選擇 sequence 別로 next state(next step)의 지정을 위한 increment field ($1I_01I_11I_22I_02I_12I_2S$: 그림11)로써 구성된다(그림13). 이렇게 함으로서 state graph로 부터 直接 microprogram을 作成 할 수 있다.

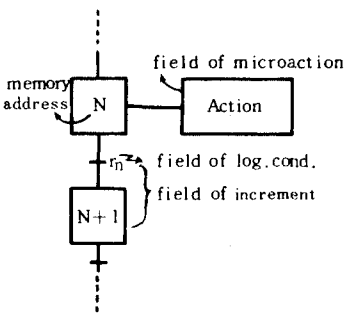


그림12. GRAFCET와 micro-instruction field와의 관계
Fig. 12. Relationship between GRAFCET and field of microinstruction.

field of microaction	field of log. cond.	field of increment

그림13. microinstrsction의 format
Fig. 13. Format of microinstruction.

3. 例題

Ⅲ-2의 그림8로 記述된 例題中 SG3을 microproprogrammed logic에 의한 構成을 위해 hardware 구조 및 microprogram을 작성한다.

前述한 바와 같이 [next step number]=[next step

直前의 logic condition의 MTX 入力 number]±[increment field]이므로 step number를 "0"으로 부터 올림순으로 하고, 또한 logic condition의 MTX의 入力 number는 가능하면 直前의 step number와 같게 하면 microinstruction의 各 field의 bit數를 줄일 수 있다.

따라서 그림8의 SG3은 그림14와 같이 step number가 새로이 주어지며, 그림15처럼 information이 연결된다.

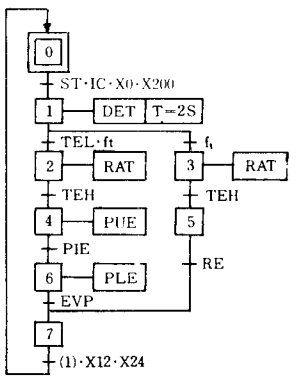


그림14. 새로이 step number한 state graph 3
Fig. 14. State graph 3 with renumbered steps.

그림15와 같이 informations 이 연결될 때 logic condition field의 code는 표2와 같다. 이때 MTX1의 logic condition "TEL·$\overline{f_t}$"와 MTX 2의 logic condition "f_t"는 同一한 入力 number를 가지며, MTX 1의 logic condition TEH와 MTX 2의 logic condition

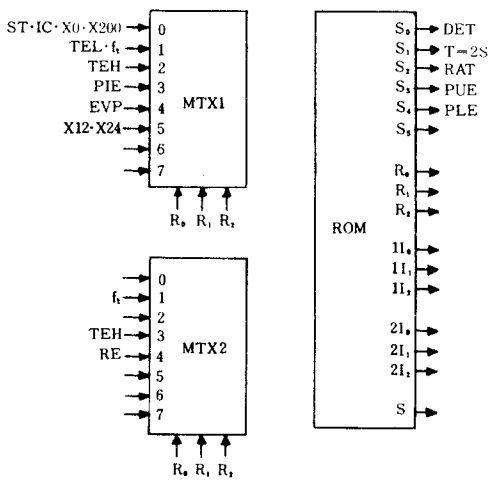


그림15. State graph 3의 hardware연결(그림11中 일부)
Fig. 15. Hardware connection of state graph 3 (a part of Fig. 11)

표 2. Logic condition field의 code
Table 2. Code of field of logic condition.

Log. cond.	Input MTX No.	Field of log. cond. ($R_0R_1R_2$)
ST·IC·×0·×200	0 (MTX 1)	0 0 0
TEL·$\overline{f_t}$	1 (MTX 1)	0 0 1
TEH	2 (〃)	0 1 0
PIE	3 (〃)	0 1 1
EVP	4 (〃)	1 0 0
×12·×24	5 (〃)	1 0 1
ft	1 (MTX 2)	0 0 1
TEH	3 (〃)	0 1 1
RE	4 (〃)	1 0 0

표 3. State graph 3의 microprogram
Table 3. Microprogram of state graph 3.

ROM address	Micro Instructions																
	S_0	S_1	S_2	S_3	S_4	S_5	R_0	R_1	R_2	$1I_0$	$1I_1$	$1I_2$	$2I_0$	$2I_1$	$2I_2$	S
0 0 0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0 0 1	1	1	0	0	0	0	0	0	1	0	0	1	0	1	0	0
0 1 0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0
0 1 1	0	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0
1 0 0	0	0	0	1	0	0	0	1	1	0	1	1	0	0	0	0
1 0 1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0
1 1 0	0	0	0	0	1	0	1	0	0	0	1	1	0	0	0	0
1 1 1	0	0	0	0	0	0	1	0	1	1	0	1	0	0	0	1

TEH는 서로 다른 入力 number를 갖는다.

따라서 그림14, 그림15 및 표2로 부터 손쉽게 microprogram을 작성할 수 있다(표3).

그림11에서 보는 바와 같이 [next step no.]=[$R_0R_1R_2$]±[$1I_01I_11I_2$ 혹은 $2I_02I_12I_2$]가 되며 여기서 S bit가 0이면 addition을 1이면 subtraction을 나타낸다. 따라서 microprogram의 작성은 그림14에서 present state가 step 0일 경우 ROM address는 "0 0 0"이 되며, 그 내용인 microinstruction에서는 action이 없으므로 microaction field $s_0s_1\cdots s_6$는 "0 0 … 0"로, next step으로의 천이조건인 "ST·IC·×0·×200"은 MTX1의 入力 번호 0에 연결됐으므로 logic condition field $R_0R_1R_2$는"0 0 0"로, 그리고 next step number는 1이므로 上記 관계에서 $1I_01I_11I_2$는 "0 0 1"이 되며, 한편 $2I_02I_12I_2$는 "0 0 0"로 그리고 S는 addition을 나타내는 "0"가 된다(표3의 제1 line).

Step 1으로 evolution됐을 때 ROM address는 "0 0 1", microinstruction中 step 1에서의 action은

"DET" 및 "T=2S"로서 그림15에서 各各 s_0s_1 에서 출력되므로 $S_0S_1S_2\cdots S_6$는 "1 1 0 … 0"으로 되며, "TEL·$\overline{f_t}$"와 "f_t"가 MTX1과 MTX2에서 各各 同一한 入力 number 1에 연결됐으므로 $R_0R_1R_2$는 "0 0 1", 그리고 "TEL·$\overline{f_t}$"가 MTX1에서 감지되면 step 2로 evolution 되므로 $1I_0 1I_1 1I_2$는 "0 0 1"이 되고 만일 "f_t"가 MTX2에서 감지되면 step 3으로 evolution되므로 $2I_0 2I_1 2I_2$는 "0 1 0"으로 되며 S는 "0"이 된다(표3의 제 2 line).

이와 같이 各各의 모든 step에 대해서 microprogram을 S.G로 부터 直接 작성할 수 있으며 표3과 같다.

V. 結 論

GRAFCET로 記述된 竝列 sequence를 갖는 順序論理 시스템을 programmed logic에 의해 實現하는 方法을 提示했다.

대규모(수십-수천)의 入出力 變數 및 內部狀態 變數가 필요한 industrial automation은 sub-system의 개념 및 竝列 function을 記述 할 수는 있는 기법이 요구되는 바, 이러한 기능을 記述하기 위해서는 GRAFCET外의 古典的인 方法(例로써 Huffman의 方法등)으로는 不可能하다. 따라서 竝列 sequence를 갖는 GRAFCET를 構成하는 方法이 관심의 대상이 되어왔다. 더욱이 automation 대상이 점점 더 복잡해지고 또한 IC기술의 발달로 인해 使用 素子 費用이 상대적으로 낮아짐에 따라, 記述 結果로 부터 손쉽게 直接 構成할 수 있는 方法 즉 transcription할 수 있는 構成 方法이 추구되어온 바 本 논문에서는 이러한 요구 조건에 맞는 構成 方法을 提示했다. 특히 提示된 multiprocessor에 의한 構成 方法은 공간적으로 떨어져 있는 시스템의 automation에 적합한 方法으로서, 産業 공정의 자동화등에 적용될 수 있으리라 기대된다.

이 논문은 1983년도 문교부 학술연구 조성비에 의하여 연구되었으며 후원해준 문교부에 감사한다.

參 考 文 獻

[1] B. Taconet et B. Challot, *Programmation du GRAFCET Sur Automate a langage Logique, a Relais ou Booléen.* Le Nouvel Automatisme, pp. 44-45, Jan.-Feb., 1979.

[2] T. Maurin et M. Robin, *GRAFCET une Traduction Micro-Programmée.* Mesure-Regulation-Automatisme, pp. 75-80, Mai, 1979.

[3] J.P. Cocquerez et J. Devars," Synthése d'automatismes séquentiels à lâide de circuits logiques programmables," *L'onde Electrique,* vol.59, no.4, pp. 72-78, 1979.

[4] L. Tourres, "Une methode nouvelle détude des systèmes logiques et son application à la réalisation d'automatismes programmés," *RGE,* Tome 85, no. 3, pp. 215-219, Mars, 1976.

[5] J.M. Toulotte; *Réseaux de Pétri et Automates Programmables.* Automatisme, pp. 200-211, Juillet-Aout, 1978.

[6] M. Blanchard; "Approche unitaire de la conception des systèmes logiques. *I.T. E.T.* no. 215, pp. 7-13.

[7] C. André, F. Boeri et J. Maurin, *Synthèse et réalisation des systemes Logiques à Évolutions Simultaneés.* R.A.I.R.O., pp. 67-86, Avril, 1976.

[8] Mosilva et R. David, "Synthése programmée des automatismes logiques décrit par réseaux de petri: une méthode de mise eu oeuvre sur microcalculateur," *R.A. I. R.O.,* vol. 13, no.3, pp. 369-393, 1979.

[9] E. Daclin et M. Blanchard, *Synthese des Systemes Logiques.* Cepadues-Edition, Toulouse, 1976.

[10] G. Michel, C. Laurgeau et B. Espiau, *Les Automates Programmables Industriels.* Dunod technique Paris, 1979.

[11] S. Thelliez et J.M. Toulotte, *GRAFCET et Logique Industrielle Programmée.* Eyrolles, Paris, 1982.

[12] S. Thelliez et J.M. Toulotte, *Applications Industrielles du GRAFCET.* Eyrolles, Paris, 1983.

[13] J.C. Bossy, P. Brard, P. Faugere et C. Merlaud, *Le GRAFCET a Pratique et ses Applications. Educalivre,* Paris, 1979.

[14] M. Blanchard, *Comprendre Maîtriser et Appliques le GRAFCET.* cepadues-éditions, Toulouse 1979.

[15] J.C. Pruvost, *Point en Automatique vol. 1* Technique & Documentation, Paris, 1981.

[16] J.M. Bernard et J. Hugon, *De la logique Cablée aux Microprocesseurs Tome 1,2,3, 4. Eyrolles,* Paris, 1979.

[17] M.M. Mano, *Digital Logic and Computer Design.* Prentice-Hall, 1979.

[18] Hill & Peterson, *Digital Systems: Hardware Organization & Design* John Wiley & Sons, 1978. *