자동 조립 장치를 위한 지능 관리 제어기 개발에 관한 연구

°전 철 항, 이 태 형*, 서 일 홍**, 허 경 무***, 변 증 남***
o 현대전자 * 삼성항공 ** 한양대학교 *** 한국과학기술원

Development of an Intelligent Supervisory Programmable Controller for Automatic Assembly Machine

C.H Jeon°, T.H. Lee*, I.H. Suh**, K.M. Huh***, and Z. Bien***
o HYUNDAI Elec., * SAMSUNG Aero. ** HANYANG Univ. *** KAIST

ABSTRACT

In this paper an intelligent supervisory programmable controller for automatic assembly machine is developed. This is achieved by adding sequence control hardware with input-interrupts to supervisory real time language and also by incorporating an automatic planning method which uses a predicate logic model and an action model.

The designed intelligent supervisory programmable controller is applied to Die Bonding Machine and is to found to work well.

I. 서 론

산업 자동화의 일부인 조립 공정의 자동화에서 주 제어 장치로써 사용되고 있는 P.C.(Programmable Controller)는 자동화의 대상및 조립공정이 점점 복잡해지고 다양해짐에 따라서 좀 더 다양한 기능 및 지능(Intelligence)이 요구되고 있다. 이러한 요구에 부응하기 위해서 한개의 CPU만을 사용하고 단순한 주변장치를 갖는 종래의 PC와는 달리 여러개의 Multi-CPU를 갖는 다기능 PC까지 등장하고 있는 바[1], 이러한 PC들은 전체 시스템의 원활한 운용 및 사용자와의 Man-Machine Interface, Error 진단을 위한 Master 역할을 수행하는 부분인 관리 제어기 (Supervisory Controller)를 갖고 있어야 한다.

본 논문에서는 우선 이러한 Supervisor 역할을 수행할 수 있는 P.C를 고안하고, 자동조립의 전체 작업 순서(Job Sequence)를 기술할 수 있는 Language 및 Sequence제어 구현방법을 제시하기로한다. 또한 P.C로서의 진보된 기능으로 인공 지능을 갖는 Automatic Planning 방법을 제안하고자 하며, 제안된 Supervisory P.C를 반도체 자동 조립공정에 응용하여 실험 검토하고자 한다.

II. Supervisory P.C 의 구성 및 기능

본 절에서는 Supervisory P.C 의 H/W 및 S/W 구조에 대해 논하고 Intelligence 구현 방법을 제안한다.

2-1. Supervisory P.C의 구성개요

Supervisory P.C는 전체 자동 조립 머신의 공정 상황을 감독 및 지시하는 역할을 주목적으로 하는 것으로, 크게는 각 Subsystem의 일의 진행 상황을 지시, 감독을 하며, 사용자로부터 입력된 명령을 해석하고, 결정된 일에 대해 기술된 작업순서대로 운용시키며, 각 Subsystem의 운용상황을 감독한다. 그림 1에 본 논문에서 제안한 전체 공정 제어 시스템의 구조를 보였다. 그림 1. 에서 보듯이 여러개의 CPU는 Time-shared Common Bus 구조[1,2]를 갖도록 연결되어 있으며 시간적으로 Bus를 분할 할당해주기 위한 Bus Arbiter를 갖고, 각 CPU간의 통신 방법은 Common Memory를 통하여 일정한 시간간격으로 Polling함으로써 통신을 수행하도록 하였다.

2.2 Subsystem의 명령 수행 방식 및 Synchronization

각 Subsystem은 Supervisor로 부터 명령을 받아 수행을 하고 각 Subsystem간의 통신은 없는 것으로 하였다. 명령의 종류는 명령의 중요도 및 순서에 따라 3가지 Priority를 갖는다.
Priority 2는 가장 높은것으로 현재 수행중인 명령을 멈추고 바로 이 명령을 수행하고 priority 1은 현재 수행중이던 명령을 멈추고 이 명령을 수행하다가 앞의 명령을 수행한다. Priority 0 는 지금 수행중인 명령을 계속 수행한 다음에 수행한다.

Supervisory CPU는 Common Memory의 일정한 약속된 장소를 통해 각 Subsystem에 명령을 내리고 각 Subsystem은 이 장소를 주기적으로 Polling함으로써 전달받으며, 전달된 명령을 해석하여 Subroutine Table을 찾아서 수행하게 된다. 각 Subsystem은 똑같은 구조로 위의 과정을 수행하는데 이 부분을 Background 부분이라하며 Flow Chart는 <그림 2>와 같다. 그리고 Supervisory P.C와 다른 Subsystem들과의 통신은 Common Memory를 통하여 하고 같은 장소에 대해 서로 다른 목적으로 Write하게 되므로 Synchronization Problem이 발생할 수있다. 이러한 문제를 해결하기 위해 Micro Processor(MC 68000)에서 제공하는

Indivisible Read-Modify-Write Cycle Instruction (TAS - Test and Set) [3]을 이용하여 Read-Write Procedure 를 사용함으로써 Reliable한 통신체계를 구성하였다.

2.3 Supervisory P.C의 S/W구조 (Master/slave 구조)[4]

앞에서 전술한 Supervisory의 역할을 수행할 수있는, Multi-CPU 실시간 운용 시스템(Real Time Operating System)의 구조를 갖도록 설계하였다. 특히 Supervisor의 수행시간이 지연될 경우에 Slave의 쉬는 시간이 늘어나게 되어 전체 시스템 Performance가 저하되는 문제점 및, 주기적인 Polling으로 인한 상호 통신의 빈번으로 다음의 식(1)로 주어지는 Processing Power ρ 를 떨어뜨리는 문제점등을 해결함과 동시에, 주어진 Task의 수행을 위해 일을 Slave에 분담시키고 전체상황을 판단하는 기본적인 관리제어 시스템의 작업수행이 가능하도록 설계하였다.

$$\rho = \frac{\text{통신을 위해 소모하는 시간(Idle Time)}}{\text{실제 일을 하는데 소모하는 시간(Action Time)}} \quad \cdots(1)$$

Supervisor CPU의 전체 구성은 <그림 3>과 같이 구분되어 있다. 크게 Condition Interpreter, Task Finder, Polling Period Adjuster 및 Action Executer등의 4개의 소프트웨어 모듈로 구분되며 각각의 기능이 다음과 같은 고유의 기능을 수행토록하였다. Condition Interpreter는 전체 시스템이 가지는 여러가지 상태에 따라 적절한 하나의 일을 찾으며 Table Searching 방법중 빠른 Searching을 위해 Hash Table[5]을 이용 빠르게 Searching하게 하였으며, Task Finder는 앞서 Condition Interpreter가 찾은 일을 일시적으로 해당 우선 순위(Priority)에 따라 해당되는 Queue에 저장한 후, 현재 수행중인 일과 Queue에 저장되어 있는 Task의 우선순위를 비교하여 다음에 수행되어야 할 Task를 정한다. 또한 Polling 주기 조절기(Polling Period Adjuster)는 Polling에 의한 단점을 개선하기 위하여 Slave의 일이 진행되는 상황을 주기적으로 Feedback 받아서 Slave가 현재 수행중이던 일을 완료하고 다음 작업 내용을 Supervisor로 부터 지시받을 때까지의 시간지연을 줄임으로써 Processing Power가 항상되도록 Polling주기 가변에 관한 정보를 미리 만들어 놓기 위한 것이며, Action Executer는 지정된 Task를 순서적으로 수행한다. 이상 기술한 Module들에 의해 이루어지는 Supervisor P.C의 S/W구조의 주 흐름도는 <그림 4>와 같다. 그리고 Queue 구조를 기준으로는 <그림 5>와 같이 구성된다.

2.4 시이퀀스 제어의 구현

자동조립공정에는 결정된 하나의 일에 대해서 여러단계의 작업순서로 구성되어 있는 경우가 많은데 이러한 작업 순서(Job Sequence)를 기술하는 방법에는 여러가지가 있으나 본 논문에서는 Real Time Language라하여 <그림 6>과 같은 Language를 이용하여 한개의 일(Task)를 구성하도록 하였다. Supervisor P.C는 이러한 Language로 구성되어 있는 Table을 한개씩 Interprete하여 작업 순서를 만들게 하였다. 또한 이러한 작업 순서를 운용시키는 것외에 Peripheral Device로 부터 들어오는 I/O 신호를 받아 CPU에게 Interrput를 걸어 현재의 Task외에 필요한 I/O신호에 대한 Service를 수행하는, Multitasking이 가능하도록 설계하였다.

2.5 Intelligence 구현 방안

앞서 설명한 Language로 작업 순서를 기술하기 위해서는 사용자가 Language를 이용하여 전체 작업 순서마다 결과로 나타나는 것을 일일이 알아야 하므로 Program하는데는 고도로 숙련된 Operator를 필요로하게 된다. 이러한 단점을 해결하는 한 방법은 작업순서를 일일이 기술하는 방법보다 간단히 목적으로 하는 Task를 수행하면 나타나는 환경 상태를 우리가 이해하기 쉬운 Logic으로 표현하게 하여 Supervisor 스스로 자동계획(Automatic Planning)의 기능을 갖도록 하는 것이다[6]. 본 논문에서는 이러한 인공 지능을 갖는 PC를 다음과 같이 제시하였다.

(1) World Model

현재의 환경 상태에 그러한 상태를 천이시킬 수 있는 Operation(Action)를 가했을 때 결과로 나타나는 지식으로 이루어진다. 즉 실제의 환경 상태를 사용자가 알기 쉬운 용어로써 표현하는데 본 논문에서는 Logic Language의 방식을 이용하여 Predicate Logic으로 Real World를 기술하였다.

(2) Action Model

현재의 상태를 다른 상태로 천이시키는 Operation들의 변형효과를 기술하는 것으로 다음과 같은 Format 으로 표현한다.

```
Operator Name : Arguments
Precondition  : Precondition Literals
Del_list      : Delete_list literals
Add_list      : Add_list literals
```

즉 주어진 환경상태에서 Precondition이 만족되어야 Operator를 적용할 수 있으며, 적용한 뒤의 결과가 Add_list에 명시된 조건들을 환경상태에 더하고, Del_list의 조건들이 현재의 환경상태에서 제거되도록하였다.

(3) Planning 과정의 S/W 흐름

Plan Generation Process가 주어진 초기 상태에서 Goal 상태를 얻기위한 작업 순서를 선택한다. 즉, Plan Generation Process가 Search Tree안에서 Solution Path를 찾아가는 것으로 본 논문에서 구현된 S/W flow는 <그림 7>과 같다.

III. 적용예 : Die Bonding Machine에의 응용

본 논문에서 제시된 방법의 타당성을 보이기 위하여 설계된 관리 제어 시스템을 반도체 자동 Die Bonding 공정

제어에 응용하였다. Die Bonding 시스템의 주된 기능은 Wafer에 있는 Die들중 양호한 것만을 집어 Lead Frame에 붙이는 것인데, 이를 위해 기본적으로 Die를 집어 Package위에 붙이는 일을 하는 Bonding Head와 이 Head가 집어가는 위치로 Die를 옮기는 XY Table, Die의 양 불량을 판단하고 Die의 위치오차, 경사도 등을 계산하는데 쓰이는 비젼 시스템 및 조명 장치등으로 구성된다. 이러한 기능을 갖는 Die Bonding 시스템을 본 KAIST 제어 연구실에서 연구 개발하고 있는 바, 그 제어 H/W 구성은 <그림 8>과 같다. 본 Die Bonding시스템이 가지는 조건은 내부 조건과 외부 조건으로 구분하였는 바, 외부 조건은 Key입력으로 보고 내부 조건은 State, Subflg, Page number, Menu Number 등의 종류로 구분하였다. 따라서 Key의 갯수를 줄이면서 Menu라는 User선택 S/W기능을 두어 편리하게 사용할 수 있도록 더 필요한 기능이 생기면 계속 S/W에 첨가시킴으로써 확장이 용이하게 하였다. 참고로 내부조건중 시스템이 가지는 상태인 State의 이름 및 그 Transition Diagram은 <그림 9>와 같다. 위의 <그림 9>에서 Bonding State내의 Bonding Task에 대해서 외부 입력신호 및 출력 신호에 대해 즉각적으로 반응해야 하므로 Interrupt를 이용하여 원하는 Timing Chart를 만들 수 있었다. 또한 구현된 Task의 종류는 각 Priority Level에 따라 구분하여,

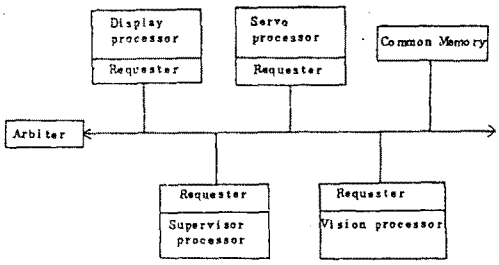
Automatic Planning을 위한 Operator table을 <표 1>과 같이 구성하여 원하는 Task에 대해 각 명령의 순서 (Action Sequence)를 만들어 보았다.

IV. 결론

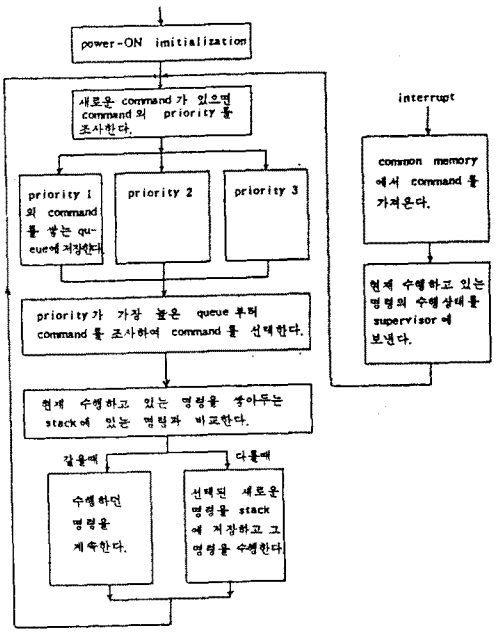
본 논문에서는 멀티프로세서 구조를 갖는 자동 조립 기계장치의 제어기에 적합한 실시간용 Supervisory P.C를 제안하여 그 구성을 Module화 함으로써 좋은 Expandibility를 갖는 구조로 설계하였고, 그리고 고도의 지능을 갖는 방법으로써 Automatic Planning의 방법을 제시하였다. 또 제안된 구조로 갖는 자동 조립 시스템의 관리 제어기를 구성하고, 이를 KAIST 제어 연구실에서 개발중인 반도체 Die Bonding용 자동 조립장치에 응용하여 그 운용이 효율적이고 원활하게 됨을 보였다.

V. 참고문헌

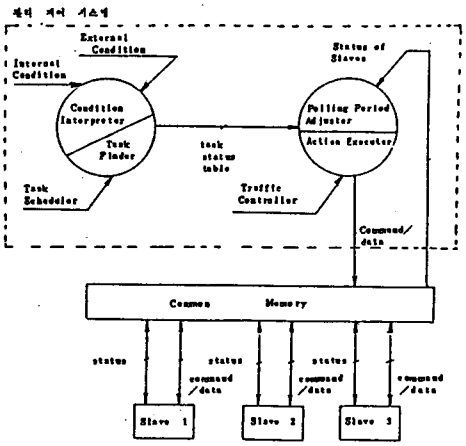
1. P.H. Enslow JR., "Multiprocessor Organization - A survey", Computing Survey, Vol.9, No.1, March 1977.
2. G.A. Anderson and E.D. Jensen, "Computer Interconnection Structures : Taxonomy, Characteristics and Examples" Computing Surveys, Vol. 7, No.4, DEC. 1975
3. "MC 68000 16 bit Microprocessors", April, 1983
4. "P.H. Enslow JR., "Multiprocessor and Parallel Processing", John Wiley & Sons, 1984
5. W.D Maurer and T.G. Lewis "Hash Table Method", Computing Surveys, Vol.7, No.1, pp. 5 - 19, March 1975
6. Michael J.P. Show, Andrew B., "Automatic Planning Flexible Scheduling : A knowledge- Based Approach", 1985



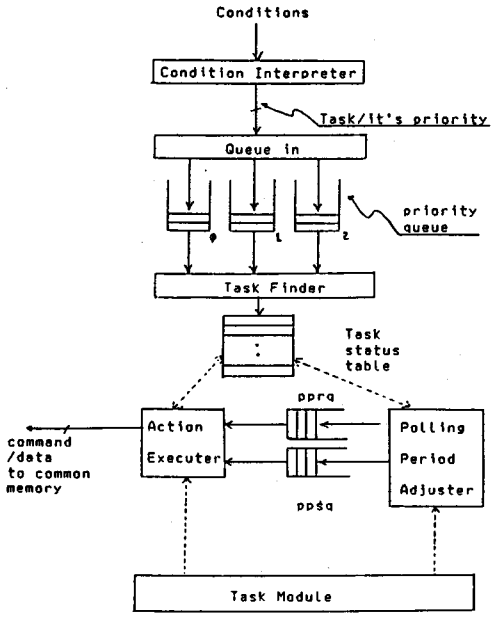
<그림 1 Supervisory P.C.의 Multiprocessor Hardware Structure>



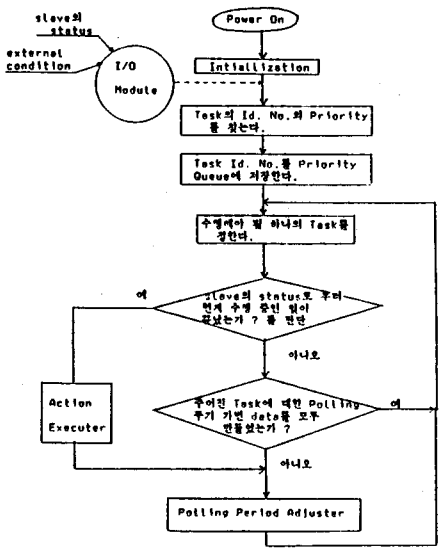
<그림 2 Subsystem의 Background Program>



<그림3 관리 제어 시스템의 구성도>



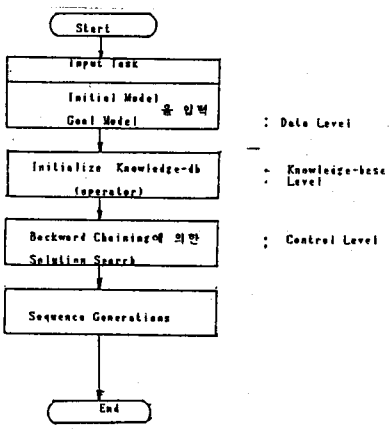
<그림 5 > Queue를 중심으로 한 관리 제어 시스템의 구성도



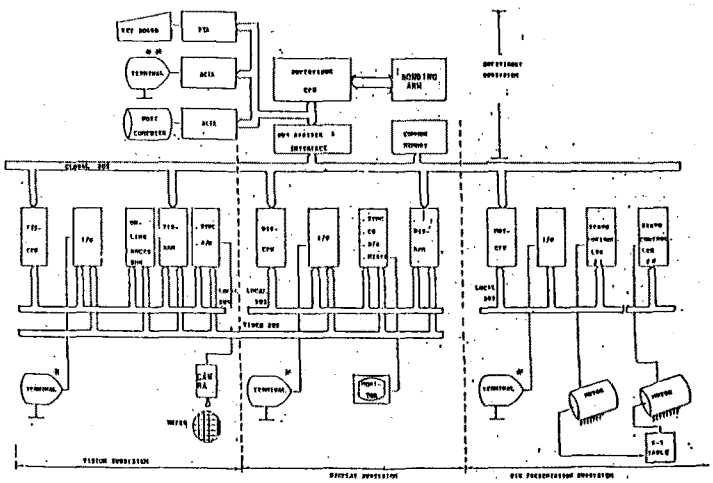
<그림 4 > 관리 제어 시스템의 Main Flowchart

Action	내 용
MVCOM i	i 번째 Slave 에게 command를 전달한다.
MVDAT	Data 를 common memory 에 전달한다.
TABCC	Data 내용을 검사후 cc(condition)를 만족하면 branch
ADDAT	destination 에서 source를 더해 desti. 에 저장
SBDAT	destination 에서 source를 뺀 desti. 에 저장
MLDAT	destination과 source를 곱해 desti. 에 저장
DIDAT	destination을 source로 나누어 desti.에 저장
GOTO	go to —
CMP	두 data 내용을 비교하여 condition을 만듦
EOT	end of task

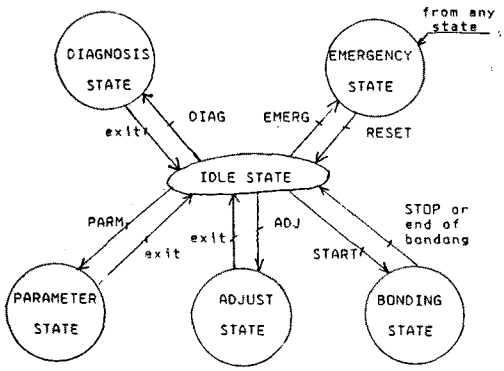
<그림6 Supervisory R.T. Language 종류>



<그림7 Planning 과정의 S/W Flow Chart>



<그림 8 Die Bonding 제어장치의 구성도>



<그림 9 > State Transition Diagram

```
**********      transition operator table **********
operator        :  SIZE(DIE)                    A
precondition    :  IN(MONITOR, THRESH)MONITOR(CLEAR)
del-list        :  null                         A
edd-list        :  IN(MONITOR, RETICLE)MONITOR(DIRTY)

operator        :  LEVEL(THRESH)
precondition    :  MONITOR(CLEAR)
del-list        :  null                         A
add-list        :  IN(MONITOR, THRESH)MONITOR(DIRTY)

operator        :  SLOPE(WAFER-SLOPE)           A
precondition    :  IN(MONITOR, THRESH)MONITOR(CLEAR)
del-list        :  null                         A
add-list        :  IN(MONITOR, WAFER-SLOPE)MONITOR(DIRTY)

operator        :  INSPECT(DIE)                 A
precondition    :  IN(MONITOR, THRESH)IN(MONITOR, RETICLE)
del-list        :  null                         A
add-list        :  IN(MONITOR, INSPECT)DIE(GOOD)

operator        :  MONITOR(DIE(GOOD))
precondition    :  DIE(GOOD)
del-list        :  DIE(GOOD)                    A
add-list        :  IN(MONITOR, GOOD-DIE)POS(XYTABLE, NEXTDIE)

operator        :  MONITOR(DIE(BAD))
precondition    :  DIE(BAD)
del-list        :  DIE(BAD)                     A
add-list        :  IN(MONITOR, BAD-DIE)POS(XYTABLE, NEXTDIE)

operator        :  MONITOR(DIE(NO))
precondition    :  DIE(NO)
del-list        :  DIE(NO)                      A
add-list        :  IN(MONITOR, NO-DIE)POS(XYTABLE, NEXTDIE)

operator        :  CLRC(MONITOR)
precondition    :  MONITOR(DIRTY)
del-list        :  null
add-list        :  MONITOR(CLEAR)

operator        :  BOND(DIE)                    A
precondition    :  ON(DIE, CENTER)DID(GOOD)
del-list        :  DIE(GOOD)                    A
add-list        :  POS(XYTABLE, NEXTDIE)ON(DIE, LEADFRAME)

operator        :  MOVE(XYTABLE)
precondition    :  IN(LIMIT)
del-list        :  null
add-list        :  ON(DIE, CENTER) POS(XYTABLE, NEXTDIE)

operator        :  CORRECT(XYTABLE)
precondition    :  IN(MONITOR, INSPECT)
del-list        :  null
add-list        :  ON(DIE, CENTER)
```

A : condition AND

< 표 1 Die Bonding 의 Operator Table)