96 대한기계학회논문집 A 권, 제 29 권 제 1 호, pp. 96~106, 2005

〈 논문 〉

예외상황 처리를 고려한 반도체 통합제조장비 시뮬레이터

김우석* · 전영하* · 이두용†

(2004 년 8 월 17 일 접수, 2004 년 12 월 1 일 심사완료)

Simulator of Integrated Single-Wafer Processing Tools with Contingency Handling

Woo Seok Kim, Young Ha Jeon and Doo Yong Lee

Key Words : Single-Wafer Processor(매엽식 반도체 제조장비), Simulation(시뮬레이션), Contingency Handling(예외상황 처리)

Abstract

An integrated single-wafer processing tool, composed of multiple single wafer processing modules, transfer robots, and load locks, has complex routing sequences, and often has critical post-processing residency constraints. Scheduling of these tools is an intricate problem, and testing schedulers with actual tools requires too much time and cost. The Single Wafer Processor (SWP) simulator presented in this paper is to validate an on-line scheduler, and evaluate performance of integrated single-wafer processing tools before the scheduler is actually deployed into real systems. The data transfer between the scheduler and the simulator is carried out with TCP/IP communication using messages and files. The developed simulator consists of six modules, i.e., GUI (Graphic User Interface), emulators, execution system, module managers, analyzer, and 3D animator. The overall framework is built using Microsoft Visual C++, and the animator is embodied using OpenGL API (Application Programming Interface).

기호설명

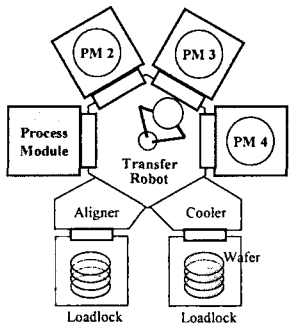
SWP(Single-Wafer Processing): 매엽식 가공
PM(Process Module) : 공정모듈
TM(Transfer Module) : 이송모듈
MTR(Main Transfer Robot) : 주 이송로봇
WTR(Wafer Transfer Robot) : 인덱서 로봇
CTC(Cluster Tool Controller) : 클러스터 장비 제어기
FOUP(Front Opening Unified Pod): 통합 표준 카세트
TMC(Transfer Module Controller) : 이송모듈 제어기
PMC(Process Module Controller) : 공정모듈 제어기
ROR(Ratio of Overtime to Residency Constraint): 웨이퍼 체류시간 초과비율

† 책임저자, 회원, 한국과학기술원 기계공학과
E-mail : lee.dooyong@kaist.ac.kr
TEL : (042)869-3229 FAX : (042)869-3210
* 한국과학기술원 기계공학과

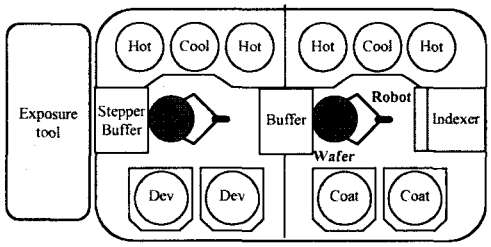
t_i^s : i 번째 작업의 시작시간
t_i^e : i 번째 작업의 종료시간
t_i^d : i 번째 작업의 마감시간
t_i^f : i 번째 작업 후 방출시간
T_i^H : i 번째 작업 시 체류시간
T_i^R : i 번째 작업의 체류시간 제약

1. 서 론

최근의 반도체 팹(fab)은 좀 더 효율적이고 유연한 생산시스템으로 변모하고 있다. 300mm 웨이퍼(wafer) 가공, 장치의 점유면적(footprint) 축소, 매엽식 처리(single-wafer processing)의 도입이 보편화됨에 따라 장비의 효율적인 운영은 더욱 중요시되고 있다. 매엽식 통합제조장비(Integrated single-wafer processing tools)는 다수의 병렬 또는 단일 공정을 수행하는 매엽처리식 공정모듈, 이송로봇, 웨이퍼의 교환이 유기적으로 일어나는 카세트



(a) Cluster tool



(b) Track system

Fig. 1 Examples of single-wafer processing tools

(cassette)로 이루어진 복합 시스템으로 여러 종류 의 웨이퍼를 동시에 처리할 수 있다. 또한 에처(etcher), 트랙장비(track system)와 같은 대부분의 전처리 장비들은 로트(lot) 단위로 생산, 이송되는 기존의 로트 처리 방식 보다는 매엽식 처리방식을 채택하고 있다. Fig. 1은 대표적인 매엽식 통합제조장비인 클러스터 툴(cluster tool)과 트랙장비[1]를 보여준다. 이들은 이송로봇의 형태와 공정모듈의 배열 등으로 구별할 수 있으며 반도체 제조공정에서 핵심적인 역할을 수행한다. 이들 장비에 대한 성능 평가는 해석적인 접근방법을 이용하기 힘들며,[2] 시뮬레이션을 통한 스케줄링은 생산성 향상을 위해 생산원가를 절감하고 단위 웨이퍼 생산시간(cycle time)을 단축시키는데 있어 보다 효과적이다.[3] 이러한 매엽식 통합제조장비의 모델링과 성능평가를 위한 많은 연구들이 이루어져 왔다.[4-6]

시뮬레이션을 위한 도구의 선택에는 크게 두 가지가 있을 수 있는데 하나는 전용 시뮬레이션 언어 혹은 시뮬레이터를 이용하는 것이고 다른 하나는 C++와 같은 일반적인 컴퓨터 언어를 이용하는 것이다. 시뮬레이션 언어는 다시 프로그래밍 시뮬레이션 언어, 생산지향(manufacturing-oriented) 언어, 그리고 생산지향 시뮬레이터로 분류될 수 있으며,[7] Table 1은 시뮬레이션 소프트웨어와 언어들의 예와 특징들을 보여준다. 시뮬레이션 언어는 일반적인 컴퓨터 언어에 비해 적은 프로그래밍 작업으로 쉽게 모델을 구축할 수 있으며 이들 언어에는 Arena, MODSIM III, SIMPLE++ 등이 있다. AutoMod, AutoSched, Quest 등은 생산지향 언어로 시뮬레이션 언어보다 더 빠른 모델링이 가능하다. 생산지향 시뮬레이터는 주로 전형적인 생산시스템의 시뮬레이션을 쉽게 할 수 있도록 하는 것에 초점을 맞춘 소프트웨어로 FACTOR, FactoryFLOW, WITNESS 등이 있으나 시뮬레이션 언어보다 유연성이 떨어진다. 일반적인 컴퓨터 언어의 경우 일정수준 이상의 프로그래밍 숙련도를 필요로 하지만 보다 유연한 모델링이 가능하고, C++와 같은 언어로 구현된 시뮬레이션 소프트웨어는 다른 소프트웨어 시스템이나 모듈과의 쉬운 인터페이스를 제공한다. 이러한 관점에서 매엽식 반도체 통합제조장비의 시뮬레이터를 개발하는 데는 일반적인 컴퓨터 언어가 적합하다고 할 수 있다.

Table 1 Examples of simulation software and their characteristics

Languages	Examples of Software	Characteristics
Simulation Language	Arena, MODSIM III, SIMPLE++	Flexible
Manufacturing-oriented Language	AutoMod, AutoSched, Quest	Powerful, rapid modeling
Manufacturing-oriented Simulator	FACTOR, FactoryFLOW, WITNESS	Easy to use, limited adaptation area
General Computer Language	C++, Java	Need for expertise, high flexibility

Table 2 The characteristics of recent simulators

Simulator	Target System	Software / Language	Scheduler	Machine Configuration
TrackSim	Track System	AutoMod, Visual Basic (UI)	Rule-Based	impossible
ClusterSim	Cluster Tool	AutoMod	Rule-Based	possible
ToolSim	Various types	AutoMod	Rule-Based	possible
VCT	Cluster Tool	VC++, OpenGL	Off-Line Scheduling	Limited
SWP	**In-line Systems**	VC++, OpenGL	***On-Line* Scheduling**	**possible**
HiFiVE – 300	Wafer Fab.	Java Sevlet, Applet, VRML	Rule-Based	Flexible
FASL	Wafer Fab.	ManSim/X, MS/X OnTime	N/A	Impossible

Table 2는 반도체 장비와 웨이퍼 팹을 대상 시스템으로 하여 개발된 시뮬레이터를 보여준다. TrackSim[1]은 *Brooks-PRI Automations* 사의 AutoMod를 기반으로 구현된 트랙장비의 시뮬레이터로 생산시스템에서 널리 쓰이는 투입과 배분규칙에 따른 장비의 성능평가를 할 수 있다. Joo and Lee[8]는 클러스터 장비의 CTC(Cluster Tool Controller)를 검증하기 위한 가상 클러스터 툴 모델을 제시하였으며, 이를 에뮬레이션(emulation)할 수 있는 VCT (Virtual Cluster Tool)을 개발하였다. FSM(Finite State Machine)에 기반하여 모델을 구성하였고 실제 제어기들간의 원격접속으로 인한 통신지연을 고려하

였다. ClusterSim 은 AutoMod 를 기반으로 클러스터 장비에 개별화 된 시뮬레이터로 각종 통계적 결과를 도시적으로 보여준다. ToolSim 은 ClusterSim 의 개선된 버전으로 몇 가지 반도체 장비의 원형을 가지고 있어 사용자가 쉽게 장비를 특화 시킬 수 있는 반도체 장비용 확장 시뮬레이터이다. Texas Instruments 의 DMOS 5 웨이퍼 팹 내 CVD(Chemical Vapor Deposition)장비의 단위시간당 웨이퍼 생산량(throughput)을 평가하기 위해 사용되기도 하였다.[9] Huang and Hsiao[10]는 클러스터 툴의 예방정비와 원격 진단을 위한 원격 관리 시스템을 제시하였고 페트리 넷(petri-net)을 이용하여 클러스터 툴(cluster tool) 시뮬레이터를 개발하였다.

반도체 팹(fab)을 대상으로 한 시뮬레이터를 살펴보면 Pikett and Zuniga[11]은 단위 웨이퍼 생산시간과 예상 용적에 대한 정보를 제공하는 FASL(Fujitsu AMD Semiconductor Limited)을 제시하였다. Kim 등[12,13]은 자바 웹 서버(java web server)에서 실행되는 서브릿(*servlet*)의 형태로 구현된 가상기계를 제시하였고, 이를 이용하여 300mm 웨이퍼 팹을 위한 웹 기반 시뮬레이터 *HiFiVE – 300* 을 개발하였다.

기존의 반도체 장비에 관한 연구들은 주로 원형 클러스터 툴의 스케줄러 단계에서의 성능평가에 초점이 맞추어져 있고, 시뮬레이터는 생산지향 언어를 이용한 반도체 팹 및 장비의 레이아웃(layout)에 따른 물류흐름 평가를 위해 개발된 것이 대부분이다. 생산지향 언어를 사용하여 시뮬레이터를 개발하는 경우 쉽고 빠른 모델링이 가능하지만 장비의 기계적 움직임을 자세히 모델링 하기에 적합하지 않으며, 단순한 규칙이 아닌 복잡한 스케줄링 알고리즘을 통합하여 구현하는데 어려움이 있다. 또한 로봇의 이송시간을 거리에 상관없이 일정하게 가정하고 있으므로 로봇의 이송시간에 따른 장비의 성능은 실제 결과와 다를 수 있다. 온라인(*on-line*) 스케줄러를 평가하기 위한 시뮬레이터에 관한 연구도 거의 이루어지지 않았다. 따라서 본 논문에서는 스케줄러가 실제 장비에 적용되기 이전에 온라인 스케줄러의 예외처리 성능을 검증하고 매엽식 통합제조장비의 성능평가를 하기 위한 시뮬레이터(SWP simulator)를 개발하였으며, 본 시뮬레이터와 연결된 스케줄러는 장비에서 발생할 수 있는 핫런(hot-run), PM(Process Module)의 고장등과 같은 예외상황과 PM 내 웨이퍼의 체류시간 제약을 고려하는 유연한 온라인 스케줄링 알고리즘을 사용한다.[14] 실제 현장에서 예외상황은 수시로 발생하며 이에 대한 효과적인 처리는 해결해야 할 중요한 문제이다. SWP 시뮬레이터는 이러한 예외상황들을 사용자의 요구에 따라 다양하게 발생시켜 온라인 스케줄러의 대처능력을 평가할 수 있게 하였고 PM 의 고장 발생 시 진행 가능한 잔류 웨이퍼를 위해 작업진행경로에 방치된 웨이퍼를 회수하는 기능을 가지고 있다. 사용자는 다양한 장비의 레이아웃과 공정(recipe)에 따른 시뮬레이션을 수행할 수 있으며 이송로봇의 3 축 운동을 고려, 이송시간을 이송거리에 비례하도록 함으로써 로봇의 이송시간에 따른 장비의 정확한 성능평가가 가능하다. 또한 시뮬레이션이 진행되는 동안 실시간으로 보여지는 애니메이션과 분석기를 통해 보다 직관적으로 시뮬레이션 상황을 파악할 수 있다.

2. 매엽식 통합제조장비

2.1 매엽식 통합제조장비의 구조

매엽식 통합제조장비의 일반적인 구조는 에칭(etching), 세정(cleaning)등의 공정이 이루어지는 공정모듈, 이송로봇, 1~3 개의 FOUP(Front Opening Unified Pod), 버퍼(buffer)로 이루어진다. 장비 내 이송로봇의 갯수와 로봇의 이동경로는 개발되는 장비의 특징에 따라 다양화 될 수 있다. FOUP 은 300mm 웨이퍼 생산 공정에서의 생산 및 이송 단위이며, 하나의 FOUP 은 일반적으로 25 개에서 30 개의 웨이퍼를 저장할 수 있다. FOUP 의 공급은 팹 내의 *OHT(Over Head Transporter)*나 AGV (Automatic Guided Vehicle)가 담당한다.

본 논문에서는 매엽식 통합제조장비인 한국디엔에스㈜의 SWP-3004 를 모델로 하여 시뮬레이터를 개발하였으며 Fig. 2 와 같이 모델화 하였다. 이 모델은 인 라인(in-line) 형태의 매엽식 통합제조장비로, 일반적인 원형 클러스터 장비와 달리 웨이퍼 운반용 이송로봇이 장비 내에서 모듈 사이를 수평운동 하는 인 라인(in-line)형식을 취하고 있다. 이송로봇은 두 개의 팔(dual-arm)을 가지고 있어 하나의 공정모듈에서 웨이퍼의 탈착(unloading)과 장착(loading)이 동시에 이루어지는 스와핑(swapping) 작업이 가능하다. 공정모듈은 공정계획

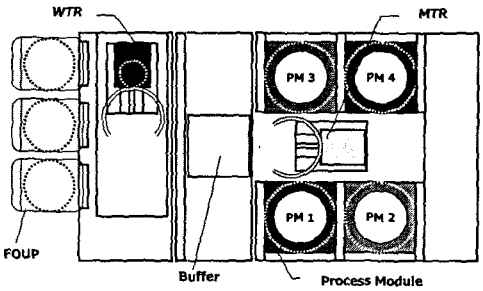


Fig. 2 Typical layout of SWP

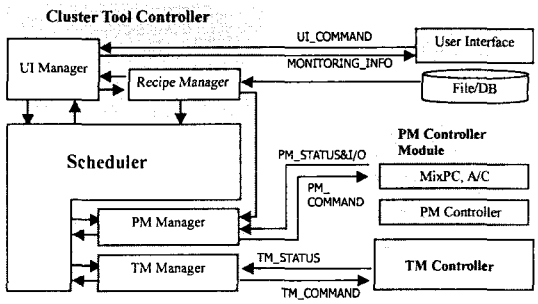


Fig. 3 Framework of SWP control system

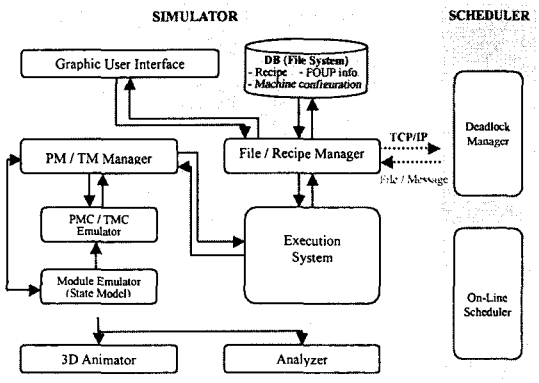


Fig. 4 Architecture of SWP simulator

에 따라, 동일한 작업을 수행하는 병렬모듈을 추가 또는 교체할 수 있고 실제 장비에서는 현재 보통 4 개가 있으나, 본 시뮬레이터에서는 8 개까지 추가가 가능하도록 하였다. 버퍼의 경우 2 개에서 6 개까지 확장이 가능하다.

2.2 관리제어 시스템

매엽식 통합제조장비는 모듈 제어기, 아날로그 제어기, 그리고 약액 주입과 관련된 MixPC 로 구성된 CTC(Cluster Tool Controller)에 의해 통합적으로 관리제어 된다. CTC 는 이송로봇과 공정모듈의 실제 하드웨어를 직접적으로 관장하고 공정모듈과 이송모듈이 적절한 작업을 수행할 수 있도록 각 하위 제어기에게 명령을 내림과 동시에 모듈의 상태를 감시함으로써 컴퓨터에서의 CPU 와 같은 중추적 역할을 한다.

본 논문에서는 SWP 의 관리제어 시스템을 크게 CTC(Cluster Tool Controller)와 모듈 제어기로 구성하였다. 스케줄러를 CTC 에 포함시켰으며 스케줄러의 기능이 상위단계에서의 스케줄 생성에 효과적으로 집중되도록 하기 위해 기존 관리제어 시스템에서 CTC 가 직접 관리하고 있던 공정모듈의 약액 주입과 아날로그 제어에 관련된 반계층적인 부분을 PMC(Process Module Controller)의 관리하에 두는, 새롭게 정의한 제어구조를 Fig. 3 에 나타냈다. 스케줄러 모듈은 Yoon and Lee[14] 가 제시한 온라인 스케줄링 방법을 사용하며, 공정모듈의 고장 등과 같은 예외상황을 고려하여 실시간으로 스케줄을 생성할 수 있는 효과적인 알고리즘을 사용한다. 또한 공정모듈 내 웨이퍼의 체류시간에 대한 제약조건을 만족시키면서 생산성을 향상시키기 위한 교착회피(deadlock-free) 방법[15]을 이용한다. 본 논문에서의 시뮬레이터는 Fig. 3 에 보이는 것과 같이 CTC 내에 장착될 스케줄러를 검증하고 평가하는 것을 하나의 주요한 목적으로 가지며, 이를 위해 장비와 제어기를 에뮬레이션 할 수 있는 모듈들을 포함한다.

3. 시뮬레이터의 구성

3.1 시뮬레이터 구조

시뮬레이터는 크게 GUI(Graphic User Interface), 에뮬레이터(emulator), 실행시스템(execution system), 모듈 관리자(manager), 3 차원 애니메이터(animator), 분석기(analyzer)를 포함한 6 개의 모듈로 구성되며 Fig.4 는 시뮬레이터의 구조를 보여준다. 에뮬레이터 모듈은 로봇과 공정모듈의 상태 모델(state model)을 가지고 있고 장비와 제어기를 모사하는 역할을 한다. 매니저 모듈은 스케줄러와의 통신을 담당하고 시뮬레이션 객체를 생성하며, 실행시스템으로부터 전달되는 로봇과 공정모듈의 작업시작 및 종료 명령을 수행하기 위해 모듈 제어기 에뮬레이터에 세부명령을 전달한다. 실행시스템은 이벤트 시간에 로봇과 공정모듈의 작업명령을 매니저에게 하달하고 예외상황처리를 위한 기능을 가지고 있다. 시뮬레이터의 전반적인 부분은 Microsoft Visual C++를 이용하여 구현되었으며 애니메이터 부분은 OpenGL API(Application Programming Interface)를 이용하였다. 스케줄러와 시뮬레이터는 독립적인 소프트웨어로 구현되었으며 소켓(socket)을 이용한 TCP/IP 통신으로 메시지와 파일을 주고받는다. 이는 스케줄러와 시뮬레이터간 인터페이스를 간결하게 하여 독립적으로 유지 보수가 가능하며 스케줄러의 변경이나 수정 시 시뮬레이터의 소스코드(source code)가 크게 변경되는 것을 방지할 수 있다. 또한 서로 다른 컴퓨터에서 접속을 유지하며 시뮬레이션을 할 수 있기 때문에 예외상황 발생 시 스케줄러는 실시간으로 스케줄링을 생성하는데 컴퓨터의 자원을 보다 집중 시킬 수 있다.

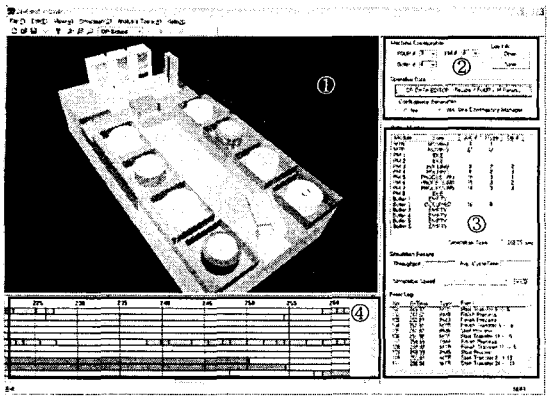


Fig. 5 Graphic user interface

사용자는 GUI 의 데이터 입력패널(Fig. 5 ②)을 통해 장비의 레이아웃, 작업계획, FOUP 에 저장되어 있는 웨이퍼의 종류에 대한 파일을 저장 또는 불러올 수 있고 예외상황관리자를 선택하여 다양한 예외상황을 편집할 수 있다. 시뮬레이션이 시작되면 애니메이션 화면(Fig. 5 ①)은 실시간으로 3 차원 애니메이션을 보여주며, 사용자는 애니메이션과 함께 모니터링 창(Fig. 5 ③), 간트 차트(Gantt Chart)(Fig. 5 ④)를 통해 시뮬레이션 상황을 보다 직관적으로 파악할 수 있다.

3.2 에뮬레이터

Fig. 6 에서 보이는 바와 같이 에뮬레이터는 모듈 제어기 에뮬레이터와 모듈에뮬레이터로 구성된다. 모듈 제어기 에뮬레이터는 공정모듈 제어기(Process Module Controller, PMC) 에뮬레이터와 이송모듈 제어기(Transfer Module Controller, TMC) 에뮬레이터로 나뉘며 모듈 에뮬레이터는 PM(Process Module)과 TM(Transfer Module)의 상태모델을 가지고 있다. 웨이퍼 이송작업을 위해 TMC 는 로봇으로 하여금 장착, 탈착 등의 단위동작을 수행하게끔 하는 제어함수들을 가지고 있으며 PM 의 경우 하나의 PMC 가 여러 PM 들을 관리한다.

3.2.1 모듈 에뮬레이터 및 기본 데이터

모듈 에뮬레이터는 자원들과 웨이퍼의 물리적 객체와 웨이퍼의 작업계획을 나타내는 추상적 객체 등으로 이루어지며 이들은 C++ 클래스의 형태로 구현되고 각 모듈들의 현재 상태를 저장하는 속성들과 시뮬레이션 결과수집에 필요한 변수들을 포함한다. Fig. 7 은 자원들과 웨이퍼의 상태 다이어그램을 보여준다. 공정모듈의 경우 모듈 내 웨이퍼를 고정하는 척(chuck)의 움직임이나 약액 주입을 위한 밸브의 개폐는 고려하지 않았으나 이송로봇의 경우 단위동작을 수행하기 위한 세부 상태

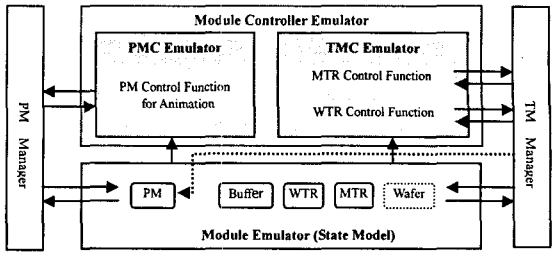


Fig. 6 Framework of emulator module

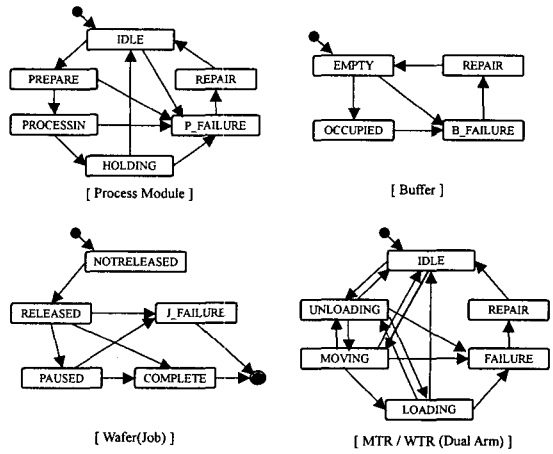


Fig. 7 State diagram of resource and wafer

태들을 정의하였다. 웨이퍼의 상태에는 COMPLETE, NOTRELEASED, RELEASED, PAUSED, J_FAILURE 가 있으며 통계를 위한 자료로만 사용되고 상태들의 순환(cycle)을 갖지 않는다.

3.2.2 모듈 제어기 에뮬레이터

모듈 제어기 에뮬레이터는 TM 의 단위동작과 PM 의 작업을 수행하며 TMC 에뮬레이터는 WTR(Wafer Transfer Robot)과 MTR(Main Transfer Robot)의 제어함수들을 가지고 있다. 이들 제어함수들은 3 축의 위치변수들을 조정하여 로봇을 회전시키거나 팔의 슬라이딩 등의 동작을 행하게 되며 3 축 방향의 속도에 따라 작업수행 시간이 결정되게 된다. TMC 에뮬레이터는 서른 개의 제어함수들로 이루어 지며 Find Position, Unload, Load, Move, Rotated 의 다섯 개의 비슷한 기능적 그룹으로 분류될 수 있다. MTR 의 경우 Swap 그룹이 추가된다. Fig. 8 은 MTR 의 제어함수 중 Find Position 그룹에 속하는 FindULPosMTR() 함수의 흐름도를 보여준다. FindULPosMTR()는 탈착을 하게 되는 모듈의 z 방향 웨이퍼 투입구의 위치로 MTR 의 팔을 이동시키는 제어함수로 먼저 두 개의 팔 중 어떤 팔을 사용할 지를 결정해야 하며 두 개 다 비

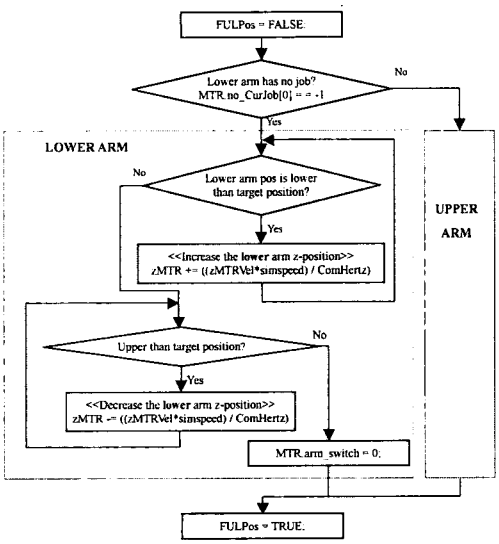


Fig. 8 Subroutines of FindULPosMTR() control function

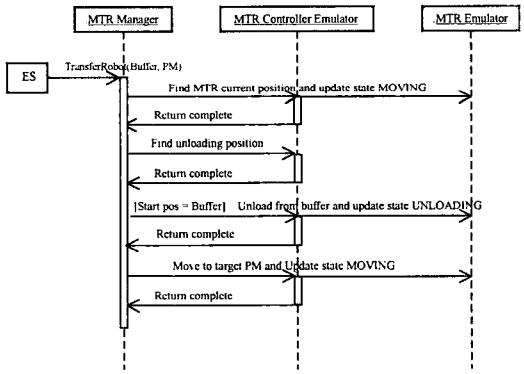


Fig. 9 Sequence diagram of MTR command

어 있을 경우 아래팔이 기본으로 사용된다. 다음 선택된 팔의 z 위치가 탈착하게 되는 웨이퍼의 투입구의 z 위치에 이를 때 까지 MTR(Main Transfer Robot) 팔의 z 방향 변수인 m_zMTR 을 변화시킨다. 팔이 원하는 위치에 다다르면 제어함수는 작업을 완료했다는 반환값을 상위 제어기인 MTR 매니저에 반환하게 된다. PM(Process Module)의 경우 ProcessPM() 제어함수가 모든 PM 의 작업시작과 종료명령을 실행하며 공정계획에 따른 일정시간을 소비하고 나면 PM 매니저로 완료되었다는 메시지를 전달한다.

3.3 모듈 매니저

기능적인 관점에서 볼 때 모듈 매니저는 PM 매니저, TM 매니저(MTR/WTR 매니저), File/Recipe 매니저로 구성된다. TM 매니저는 실행시스템으로부터 전달된 이송명령을 수행하기 위해 TMC

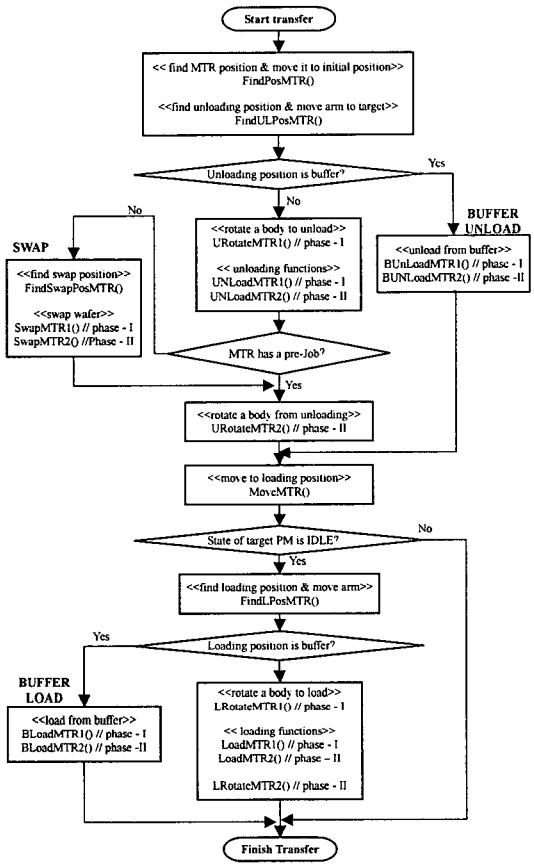


Fig. 10 Subroutines of MTR manager

에뮬레이터의 제어함수들을 호출하게 되며 Fig. 9 는 버퍼에서 PM 으로의 웨이퍼 이송명령 시 MTR 매니저, MTRC 에뮬레이터(Main Transfer Robot Controller Emulator), MTR 에뮬레이터간 주고받게 되는 제어입력의 전달과정 예를 보여준다. MTR 의 경우 스와핑 작업과 버퍼와 PM, PM 간의 이송 등 WTR(Wafer Transfer Robot)과 달리 비교적 복잡한 움직임을 가지게 되며 MTR 매니저가 MTRC 에뮬레이터의 제어함수들을 호출할 때 여러 가지 조건에 따라 다른 루틴을 가지게 된다. Fig. 10 은 주어진 조건들을 검토하여 MTR 의 이송명령을 수행하는, MTR 매니저의 작업흐름을 나타낸다. 블록 안의 함수들은 MTRC 의 제어함수들이다. 장착, 탈착 모듈이 버퍼인 경우 BUFFER LOAD/UNLOAD 루틴을 실행하며, 탈착을 시작하기 전 웨이퍼를 이미 가지고 있는 경우 탈착이 끝난 상태에서 이동을 하지 않고 바로 장착을 하는 스와핑을 하게 된다. 또한 MTR 이 장착위치로 이동했으나 해당 PM 이 작업을 수행중인 경우 현 상태에서 대기하며, 그렇지 않고 PM 내 웨이퍼가 없는 경우 즉,

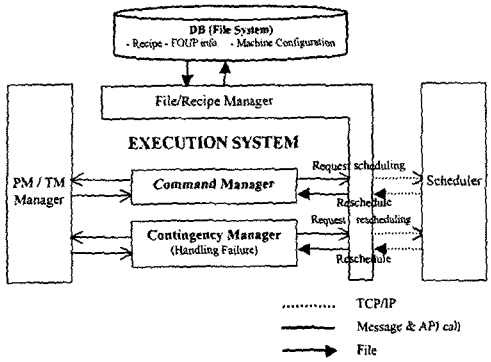


Fig. 11 Framework of execution system

PM 의 상태가 IDLE 인 경우엔 웨이퍼를 장착한다. MTR 의 상태가 바뀌게 되는 일련의 단위작업이 끝나면 MTR 매니저는 MTR 의 현재 상태를 업데이트 하며 통계를 위한 각종 시뮬레이션 자료들을 수집한다. 이에 대한 부분은 Fig. 10 에서 생략되었다. PM 매니저는 실행시스템의 공정 명령에 따라 PMC 에뮬레이터(Process Module Controller Emulator)의 ProcessPM()을 호출하며 PM 의 상태를 PREPARE → PROCESSING, PROCESSING → HOLDING 으로 업데이트 한다. PM 의 나머지 상태 천이는 MTR 매니저에 의해 이루어진다. File/Recipe 매니저는 장비의 레이아웃(*.lay), 공정계획(*.rec), FOUP 정보(*.fup), 스케줄 된 이벤트 리스트(report.txt)등의 파일을 관리하며, 이들 파일로부터 얻어진 자료를 기반으로 동적으로 시뮬레이션 객체들을 생성하는 역할을 한다. 또한 스케줄러와 시뮬레이터간 메시지 및 파일통신을 담당한다.

3.4 실행시스템

SWP 시뮬레이터의 실행시스템은 생산시스템에서의 MES(Manufacturing Execution System)와 유사한 역할을 하며 스케줄러에 의해 생성된 이벤트 리스트에 따라 모듈들의 작업명령을 매니저에 전달하는 명령 처리기(command manager)와 예외상황이 발생했을 때의 후속처리를 담당하는 예외상황 처리기(contingency manager)로 구성된다. Fig. 11 은 실행시스템의 처리기들과 매니저, 스케줄러 사이의 입출력 관계를 보여준다. 시뮬레이션이 시작되기 전에 실행시스템은 레이아웃, 공정계획, FOUP(Front Opening Unified Pod)정보를 담은 파일을 스케줄러에 전송하고 최초 스케줄링을 요청하며, 스케줄러로부터 받은 report.txt 파일을 읽어 들여 ETM[].TEVENT[]와 EPM[].PEVENT[]의 이벤트 리스트를 생성, 명령 처리기가 WTR(Wafer Transfer Robot)에 작업시작 명령을 전달함으로써

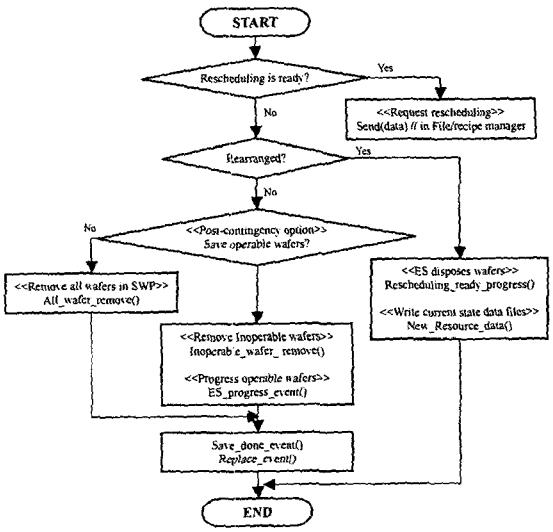


Fig. 12 Operation procedure of contingency manager

시뮬레이션이 시작된다.

예외상황 처리기는 다양한 예외상황을 발생시키기 위한 고장리스트를 작성하고 고장발생 시 잔류웨이퍼의 처리를 선택할 수 있는 예외상황 편집기능을 제공하며 Fig. 12 는 예외상황 처리기의 작업흐름을 나타낸다. 고장리스트의 생성은 두 가지 방법이 있는데, 하나는 고장이 발생할 웨이퍼의 특정공정이 진행된 시간에 의한 것이고, 다른 하나는 특정 PM(Process Module)의 고장발생시간을 지정하는 것이다. 잔류웨이퍼의 처리에는 고장발생시 이송로봇을 포함한 장비내의 모든 웨이퍼를 제거하는 것과 잔류웨이퍼의 작업을 진행시키는 선택이 가능하며, 후자의 경우 공정순서 중 이미 고장 PM 을 거쳐 나머지 공정을 수행하는데 영향이 없는 웨이퍼와 고장 PM 이 공정순서에 없어 나머지 PM 으로 작업이 진행 가능한 종류의 웨이퍼를 포함하며 더 이상 진행 가능한 웨이퍼가 없어 재 스케줄링이 불가능한 경우 자동으로 시뮬레이션을 종료한다. 시뮬레이션이 진행되는 동안 예외상황이 발생하게 되면 실행시스템은 예외상황 처리기를 실행시키게 되는데 예외상황 처리기는 편집기로부터 선택된 잔류웨이퍼 처리방식에 따라 후속처리를 실시하며 추가로 진행된 이벤트와 상태들을 저장한 뒤 스케줄러에 재 스케줄링을 요청하고 작업을 종료한다.

3.5 분석기

시뮬레이터의 분석기는 일반적으로 반도체 생산에서 통합제조장비의 단일 성능을 평가하기 위해 널리 사용되는 다양한 성능지수들을 평가한다.[15,16]

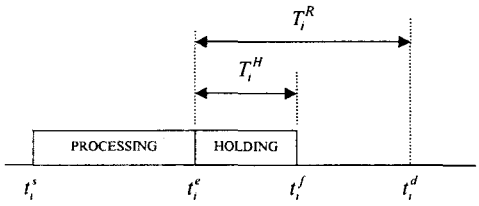


Fig. 13 An i[th] processing operation with post-processing residency constraint of execution system

성능지수에는 총 생산시간(makespan), 단위시간당 웨이퍼 생산량(throughput rate), 단위 웨이퍼 생산시간, 총 생산시간(total operation time), 모듈들의 이용률(utilization)등이 있으며 본 논문에서는 추가로 웨이퍼의 품질을 가늠할 수 있는 척도로서 웨이퍼 체류제약 시간에 대한 실제 체류시간의 절대비율을 나타낸 ROR(Ratio of Overtime to Residency constraint)를 제안한다. Fig. 13 은 PM(Process Module)내 웨이퍼의 체류시간 제약을 가지는 i 번째 공정유닛을 보여준다. t_i^e 는 i 번째 작업의 종료시간을, t_i^f 는 방출시간으로 i 번째 작업을 마친 웨이퍼가 MTR(Main Transfer Robot)에 의해 실제로 PM 에서 꺼내지는 시간을 의미한다. 따라서 웨이퍼가 실제로 PM 내에서 머물게 되는 체류시간인 T_i^H 는 t_i^f 와 t_i^e 의 차로 나타난다. t_i^d 는 웨이퍼의 작업이 끝난 후 반드시 꺼내져야 하는 마감 시간을 나타내고 t_i^d 와 t_i^e 의 차이가 웨이퍼의 체류시간 제약 T_i^R 이 된다. 일반적으로 T_i^H 가 T_i^R 를 넘게 되면 불량이 발생하고, 넘지 않는 경우라도 화학공정이 많은 통합제조장비의 경우 T_i^H 가 증가함에 따라 웨이퍼의 품질이 떨어지게 되는데 T_i^H 의 T_i^R 에 대한 비율은 웨이퍼 품질의 정도를 판단하는 척도가 될 수 있으며 R_j^{HC} (j 번째 웨이퍼에 대한 ROR)은 식 (1)과 같다.

$$R_j^{HC} = \frac{1}{n}\sum_i^n \left(\frac{T_i^H}{T_i^R}\right) \quad (1)$$

n은 j 번째 웨이퍼의 총 공정수를 나타내며 R_j^{HC} 는 0 부터 1 사이의 값을 가지게 된다. 또한 분석기는 이들 성능지수들을 볼 수 있는 통계 창과 함께 실시간으로 보여지는 간트 차트, 이벤트 로그, 모니터링 창을 제공한다.

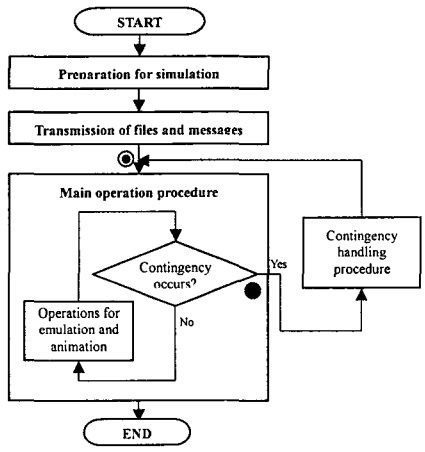


Fig. 14 Three sequential stages of operating procedure

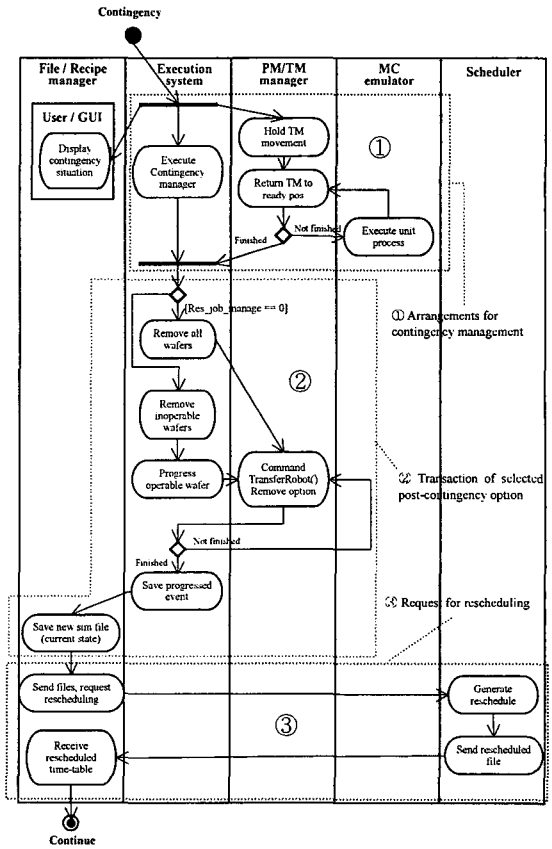


Fig. 15 Activity diagram of contingency handling procedure

4. 예외상황 처리

4.1 예외상황 처리과정

예외상황 처리를 포함한 시뮬레이터의 운영은

Table 3 Recipe information for eight process modules

Op.	Type A			Type B			Type C		
	Alternative PMs	PT	RC	Alternative PMs	PT	RC	Alternative PMs	PT	RC
1	1 or 2	40	20	3	30	20	4 or 5	40	20
2	5	30	20	4	40	20	6	30	20
3	7	20	10	7	20	10	7	20	10
4	8	20	10	8	20	10	8	20	10

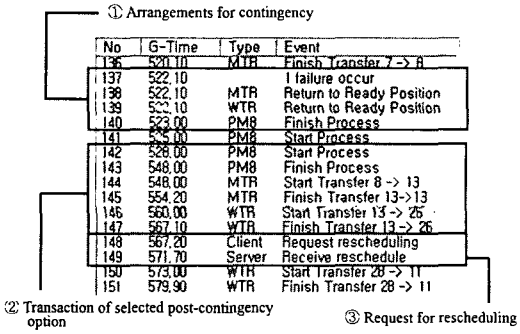


Fig. 16 Event logs after the first failure occurs

크게 시뮬레이션을 위한 준비단계, 파일과 메시지의 전송, 주 운영과정의 세 단계로 이루어 지며 Fig. 14 는 운영과정의 흐름도를 보여준다. 시뮬레이션이 진행되는 동안 실행시스템은 고장리스트로부터 예외상황을 발생시키게 되며 예외상황이 발생하면 그 처리는 주 운영과정에서 예외상황 처리기로 넘어가게 된다. Fig. 15 는 예외상황 처리과정의 세부적인 단계와 시뮬레이터의 각 모듈간의 상호작용을 보여준다. 예외상황 처리과정은 예외처리를 위한 정리(Fig. 15 ①), 후속조치의 실행(Fig. 15 ②), 스케줄링 재 요청(Fig. 15 ③)의 세 단계로 이루어 진다. 예외처리를 위한 정리단계를 보면 먼저 예외 처리기가 실행됨과 동시에 PM/TM 매니저가 공정모듈의 작업과 로봇의 움직임을 일시 정지시킨다. 그런 다음 이어질 웨이퍼 제거 작업을 실행하기 위해 로봇을 기본위치로복귀시키게 된다. 이 작업이 완료되면 예외상황 처리기는 시뮬레이션이 시작되기 전 입력된 후속 처리에 따라 모든 웨이퍼를 제거하거나 작업 가능한 웨이퍼의 작업을 진행시킨 뒤 예외상황 처리기에 의해 추가적으로 발생한 이벤트와 모듈들의 상태들을 파일에 저장한다. 이 파일들은 스케줄러로 순차적으로 보내지며 스케줄러는 고장 PM 을 제외한 잔류 PM 들로 나머지 웨이퍼에 대한 새로운 스케줄을 생성하고 시뮬레이터로 전송함으로써 예외상황 처리는 종료된다. Fig. 15 의 종료상태 Continue 는 Fig. 14 의 ⊙로 표시된 주 운영과정의 시작단계로 이어지게 되며 다시 예외상황이 발생

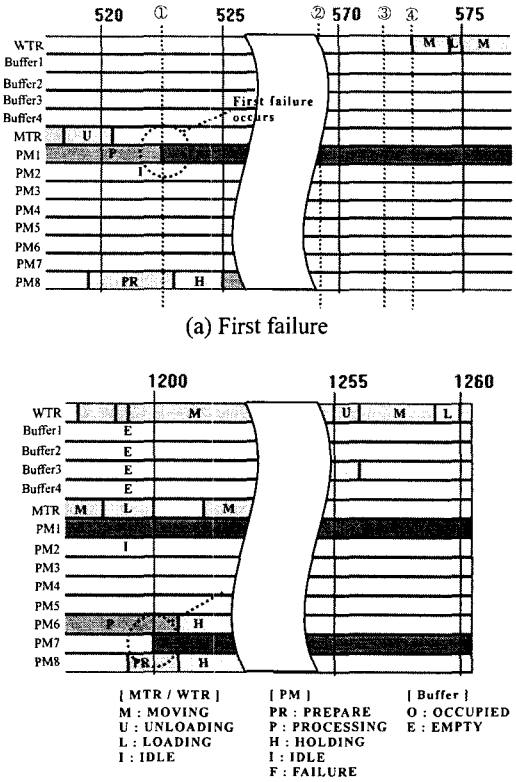


Fig. 17 Gantt chart where failures occur

할 때까지 정상적인 운영과정을 실행하게 된다. 일단 예외상황이 발생하면 FOUP(Front Opening Unified Pod)으로부터 새로운 웨이퍼의 투입은 정지되며 예외상황 처리기는 새로운 스케줄에 의해 명령 처리기가 작업명령을 내릴 때까지 장비 내 잔류웨이퍼들의 처리를 맡게 된다.

4.2 예외상황 처리 시뮬레이션

온라인 스케줄러를 평가하기 위해 예외상황을 발생시켜 시뮬레이션 해 보았고 스케줄러와 시뮬레이터는 하나의 컴퓨터에서 실행되었다. 예외상황 시뮬레이션을 위한 SWP 구성은 8 개의 공정모듈과 4 개의 버퍼로 하였고 두 번의 고장을 일정간격을 두고 발생시키도록 하였다. 첫 번째는 7 번째 웨이퍼의 첫 공정이 시작된 뒤 10 초 후에, 두 번째는 7 번 공정모듈에서 전역시간 1200 초에 각각 고장이 발생하도록 하였고 후속처리는 실행시스템이 작업 가능한 잔류웨이퍼를 진행시키도록 선택하였다. 웨이퍼는 Table 3 에 보이는 것과 같이 A, B, C 의 세가지 종류를 각각 5 개씩 번갈아 가

며 투입하였으며 A 와 C 타입 웨이퍼의 경우 첫 번째 공정이 두 개의 PM(Process Module)으로 동일하게 이루어질 수 있다. Op.는 공정순서를 나타내며 PT 와 RC 는 각각 공정시간과 공정모듈에서의 체류시간 제약조건을 초단위로 나타낸다. Fig. 16 은 첫 번째 고장이 발생한 이후의 이벤트들을 보여주며, 앞서 언급한 세 단계의 예외상황 처리 과정이 나타나 있다. 522 초에 1 번 공정모듈에서 고장이 발생(Fig. 17(a) ①)하였으며 이 때 8 번 공정모듈에 있던 웨이퍼의 작업을 완료한 뒤 다시 FOUP 으로 원위치 시키는 후속처리를 마친 다음 567.2 초에 재 스케줄링을 요청(Fig. 17(a) ②), 571.70 초에 스케줄 파일을 받아(Fig. 17(a) ③) 573 초에 실행시스템의 명령처리기가 WTR(Wafer Transfer Robot)의 이송명령을 내렸음(Fig. 17(a) ④)을 알 수 있다. 통신시간과 파일생성, 명령 처리시간을 포함해서 스케줄러가 고장 PM 을 제외한 나머지 7 개의 PM 들을 가지고 남은 68 개의 웨이퍼에 대한 스케줄을 생성하는데 약 3.5 초 정도 소요되었으며 이는 비교적 짧은 시간으로, 실시간으로 스케줄을 생성할 수 있는 효율적인 온라인 스케줄러임을 알 수 있다. Fig. 17(b)는 7 번 PM 에서 두 번째 고장이 발생하였을 때의 간트 차트이며 6 번 PM 에는 작업진행중인 웨이퍼가, 8 번 PM 에는 준비중인 웨이퍼가 각각 들어있다. Table 3 의 공정계획을 보면 세 종류의 웨이퍼가 3 번째 공정에서 모두 7 번 PM 을 거쳐야 한다. 따라서 6 번 PM 의 웨이퍼는 제거되고 8 번 PM 의 웨이퍼는 공정을 완료한 뒤 더 이상 진행 가능한 웨이퍼가 없으므로 실행시스템은 더 이상 웨이퍼를 투입하지 않고 자동으로 1260.6 초에 시뮬레이션을 종료하게 된다.

5. 결 론

본 연구에서는 온라인 스케줄러의 예외상황 처리능력을 검증하고 다양한 장비구성과 공정계획에 따른 성능을 평가하기 위한 매엽식 통합제조장비의 시뮬레이터를 개발하였다. 이송로봇과 공정모듈들의 상태모델을 정의하였고 예외상황 발생과 시험을 위한 시뮬레이터의 구조를 제시하였다. 또한 고장발생 시뮬레이션을 통해 온라인 스케줄러의 실시간 스케줄링 능력을 확인할 수 있었다. SWP 시뮬레이터의 주요 특징은 다음과 같다.

(1) 스케줄러로부터 독립된 시스템으로 구현되었고 TCP/IP 기반의 메시지 통신을 이용하여 입출력을 간략화 하였다. 이는 스케줄러의 수정 또는 교체 시 시뮬레이터의 소스코드가 크게 수정되는 것을 방지할 수 있다.

(2) 온라인 스케줄러의 예외상황 처리능력을 검증하기 위한 구조를 가지고 있다. 실행시스템의 예외상황 처리기를 통해 다양한 예외상황들을 발생시키고 후속처리에 따른 결과를 시뮬레이션 할 수 있다.

(3) 이송로봇의 3 축 방향 속도와 이송거리에 따른 이송시간을 반영하였고 장비의 구성과 가공법에 따른 세밀하고 다양한 시뮬레이션이 가능하다.

(4) 실시간으로 보여지는 3 차원 애니메이션을 통해 사용자는 보다 직관적으로 시뮬레이션 상황을 이해할 수 있다.

참고문헌

(1) Hong, S.B., Lee, D.Y. and Yoon, H.J., 2001, "Semiconductor Track System Simulator," *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, Tucson, Arizona, USA, October 7-10, Vol. 2, pp. 1036~1040.

(2) Dummler, M.A., 1999, "Using Simulation and Genetic Algorithms to Improve Cluster Tool Performance," *Proceedings of the Winter Simulation Conference,* Phoenix, Arizona, USA, December 5-8, Vol. 1, pp. 875~879.

(3) Kunnathur, A.S., Sundararaghavan, P.S. and Sampath, S., 1996, "Dynamic Rescheduling of a Job Shop: A Simulation Study," *Proceedings of the Winter Simulation Conference,* Coronado, California, USA, December 8-11, pp. 1091~1098.

(4) Wood, S.C., 1996, "Simple Performance Models for Integrated Processing Tools," *IEEE Transactions on Semiconductor Manufacturing*, Vol. 9, No. 3, pp. 320~328.

(5) Nehme, D.A. and Pierce, N.G., 1994, "Evaluating the Throughput of Cluster Tools Using Event-Graph Simulations," *Proceedings of IEEE/SEMI Advanced Semiconductor Manufacturing Conference and Workshop*, Cambridge, Massachusetts, USA, November 14-16, pp. 189~192.

(6) Akcalt, E., Nemeto, k. and Uzsoy, R., 2001, "Cycle-Time Improvements for Photolithography Process in Semiconductor Manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, Vol. 14, No. 1, pp. 48~56.

(7) Law, A.M. and McComas, M.G., 1997, "Simulation of Manufacturing Systems," *Proceedings of the Winter Simulation Conference*, Atlanta, Georgia, USA, December 7-10, pp. 86~89.

(8) Joo, Y.J. and Lee, T.E., 2004, "Virtual Control," *IEEE Robotics and Automation Magazine*, Vol. 11, issue. 3, pp. 33~49.

(9) Potti, K. and Aybar, M., 2003, "Employing a Simulation

Technique to Predict and Improve Equipment Productivity," *Technical report, Texas Instrument.*

(10) Huang, H.P. and Hsiao, Z.Y., 2002, "Development of Remote Control System of a Semiconductor Cluster Tool," *IEEE International Conference on Systems, Man and Cybernetics*, Hammarnet, Tunisia, October 6-9, Vol. 4, pp. 6~9.

(11) Pickett, B. and Zuniga, M. 1997, "Modeling, Scheduling, and Dispatching in the Dynamic Environment of Semiconductor Manufacturing at FASL, Japan," *Proceedings of the IEEE/SEMI Advanced Semiconductor Manufacturing Conference*, Cambridge, Massachusetts, USA, September 10-12, pp. 448~450.

(12) Kim, H.S., Zhou, C. and Du, H.X., 2000, "Virtual Machines for Message Based, Real-Time and Interactive Simulation," *Proceedings of the Winter Simulation Conference*, Orlando, Florida, USA, December 10-13, Vol. 2, pp. 1529~1532.

(13) Kim, H.S., Park, J.H., Sohn, S., Wang, Y., Reveliotis, S., Zhou, C., Bodner, D. and McGinnis, L., 2001, "A High-Fidelity, Web-Based Simulator for 300mm Wafer Fabs," *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, Tucson, Arizona, USA, October 7-10, Vol. 2, pp. 1288~1293.

(14) Yoon, H.J. and Lee, D.Y., 2003, " On-Line Scheduling of Robotic Cells with Post-Processing Residency Constraints," *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, Washington D.C., USA, October 5-8, Vol. 3, pp. 2785~2790.

(15) Yoon, H.J. and Lee, D.Y., 2004, "Deadlock-Free Scheduling of Photolithography Equipment in Semiconductor Fabrication," *IEEE Transactions on Semiconductor Manufacturing*, Vol. 17, No. 1, pp. 42~54.