IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 24 June 2024, accepted 22 August 2024, date of publication 28 August 2024, date of current version 10 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3450869

RESEARCH ARTICLE

Digital Twin-Based Optimization of Operational Parameters for Cluster Tools in Semiconductor Manufacturing

JOONICK HWANG[1] AND SANG DO NOH[2]
[1]Department of Semiconductor and Display Engineering, Sungkyunkwan University, Suwon-si 16419, South Korea
[2]Department of Industrial Engineering, Sungkyunkwan University, Suwon-si 16419, South Korea

Corresponding author: Sang Do Noh (sdnoh@skku.edu)

This work was supported by Samsung Electronics Company Ltd.

ABSTRACT Manufacturing and supply chain management are becoming increasingly important in the semiconductor industry. Cluster tools are essential equipment in semiconductor production and are used for microscopic operations, such as the surface treatment of wafers. In particular, the operational parameters of cluster tools directly affect productivity, and various efforts are underway to optimize them at manufacturing sites. However, there are still many challenges in this process. In this study, we propose a framework for optimizing the operational parameters of cluster tools in semiconductor manufacturing using digital twin technology to address these challenges. The framework predicts productivity through manufacturing simulations based on operational parameters in a digital twin environment, analyzes the factors affecting productivity, and determines the optimal operational parameters. The study demonstrates that the cycle time of cluster tools in wafer fabrication can be significantly reduced through the proposed approach based on digital twins. The application of this framework is expected to contribute to cost reduction, productivity improvement, and enhancement of industrial competitiveness at semiconductor manufacturing sites.

INDEX TERMS Cluster tool, digital twin, finite state machine, hill climbing algorithm, semiconductor manufacturing.

I. INTRODUCTION

Semiconductors are essential components of various electronic devices that play crucial roles in information processing and communication. Especially in the era of the Fourth Industrial Revolution, with technologies such as 5G, autonomous vehicles, artificial intelligence, and the Internet of Things (IoT), semiconductor devices are indispensable, and their shortage affects not only individual industries, but also national security [1]. Therefore, the semiconductor industry is making various efforts to meet demand by making efficient decisions in the process of semiconductor product production and supply chain management, and investing in R&D and facilities [2]. Cluster tools are advanced automation equipment used in wafer fabrication in semiconductor manufacturing processes and are essential equipment in the modern semiconductor industry that requires highly sophisticated and precise process conditions. Therefore, research on the efficient utilization of cluster tools has become an important challenge contributing to the advancement of the modern semiconductor industry.

Wafer manufacturing processes using cluster tools face several challenges. Cluster tools use various types of gases and chemicals during wafer processing [3], and because these raw materials can be hazardous to the health of the operators, careful process management is required [4]. Additionally, semiconductor products are highly susceptible to quality defects due to microscopic factors, such as the inflow of fine particles due to airflow changes during the process [5] or patterns being affected by specific wavelengths of light [6], necessitating many operational procedures and quality verifications in managing cluster tools on the manufacturing

The associate editor coordinating the review of this manuscript and approving it for publication was Rahul A. Trivedi.

floor. In particular, the operational parameters of cluster tools directly affect product quality and productivity. However, owing to the complex structure of cluster tools, various models, and differences in target processes, standardized management criteria for these operational parameters are often lacking. Although manufacturing execution system (MES) can be utilized to manage equipment operations, process control, and semiconductor manufacturing processes more efficiently [7], understanding the entire manufacturing process and operational expertise is essential for effective utilization because most of the data provided by these systems do not clearly indicate the causality of various problems occurring during cluster tool operation and manufacturing processes. For these reasons, there is a growing need for systems to predict productivity through the operational parameters of cluster tools and to analyze and optimize factors affecting productivity to improve the productivity of cluster tools.

Research on enhancing the productivity of semiconductor manufacturing processes has been previously conducted. In particular, to enhance the productivity at the wafer fabrication (FAB) level, research has been conducted on managing FAB shipments [8], predicting FAB cycle times [9], and scheduling FAB logistics [10], [11]. Research has also been conducted to enhance the productivity of cluster tools, focusing on the number of process recipes or combinations of process recipes [12], [13], wafer productivity under simplified conditions [14], and module-level productivity of cluster tools [15], [16]; however, the detailed operating units of the entire cluster tool have not been considered. Furthermore, research has been conducted to improve the fundamental scheduling methods of cluster tools [17], [18] and optimize process flows [19]; however, it is difficult to apply these methods to existing cluster tools in mass production. Considering these limitations, this study aims to explore new approaches to improve the productivity of cluster tools and expand their industrial applicability.

Digital twin technology has been proposed as an alternative for solving various problems in semiconductor manufacturing. A digital twin is defined as a digital replica of a physical object, such as a product, facility, or process. It is a virtual model that reflects the real-world situation of the target object elements, such as their properties and status, in a digital space and depicts their characteristics so that they can be monitored during design and operation throughout their lifecycle [20]. Additionally, it provides the ability to collect and analyze data from real-world systems in real time, thereby enabling the simulation and optimization of a series of operations based on these data [21]. Thus, digital twins are gaining attention as innovative solutions not only in the semiconductor industry but also in various industrial fields to enhance the efficiency of production processes and prevent defects.

This study proposes the design and application of a digital twin framework to determine the optimal operational parameters of semiconductor manufacturing cluster tools. Through this framework, manufacturing simulations based on the operational parameters of cluster tools were performed in a digital twin environment to predict productivity, and factors affecting productivity owing to operational parameters in the wafer manufacturing process were analyzed. Heuristic algorithms are utilized to iteratively explore the operational parameters to enhance the productivity of the cluster tools. The proposed method derives optimal operational parameters by considering various variables and complex interactions occurring in the semiconductor manufacturing process. Additionally, it presents technical solutions for minimizing trial and error in actual manufacturing sites, reducing the costs and time associated with process modification and improvement, enhancing the efficiency of the manufacturing process, and optimizing wafer productivity, thereby strengthening the competitiveness of cluster tools in the semiconductor manufacturing industry.

The structure of this paper is organized as follows: Section II offers a comprehensive overview of semiconductor manufacturing, cluster tools, equipment engineering systems, and digital twins. Section III elaborates on the research methodology, with a focus on the digital twin application framework, digital twin modules, and optimization modules. Section IV discusses the scenarios and practical applications of the digital twin framework, including its implementation and the results obtained. Finally, Section V synthesizes the research findings and presents the conclusions derived from the study.

II. RESEARCH BACKGROUND

A. SEMICONDUCTOR MANUFACTURING AND CLUSTER TOOLS

A semiconductor manufacturing facility is commonly referred to as a FAB. In the FAB, semiconductor products are produced by forming integrated circuits using patterns on the surfaces of bare silicon wafers, as shown in Fig. 1. Typically, silicon wafers are extremely thin, fragile to impact, and vulnerable to defects caused by fine particles. Therefore, the interiors of FABs are designed as cleanrooms with very high cleanliness levels, maintaining airborne particle concentrations below a certain level and managing internal temperature and humidity consistently [22]. Additionally, special containers, such as a front opening unified pod (FOUP), are used to safely store and transport wafers via overhead hoist transport (OHT) configured on top of the FAB to minimize physical impact and quality risks [23]. Most semiconductor manufacturing processes are automated for quality assurance and process efficiency, and various unit processes are repeated using multiple cluster tools in a predetermined order.

Cluster tools refer to automated equipment used in semiconductor manufacturing processes, attaching two or more modularized chambers to one or more wafer transfer robots to perform process steps in parallel, representing multichamber processing manufacturing equipment [24]. This concept of multichamber processing has long been

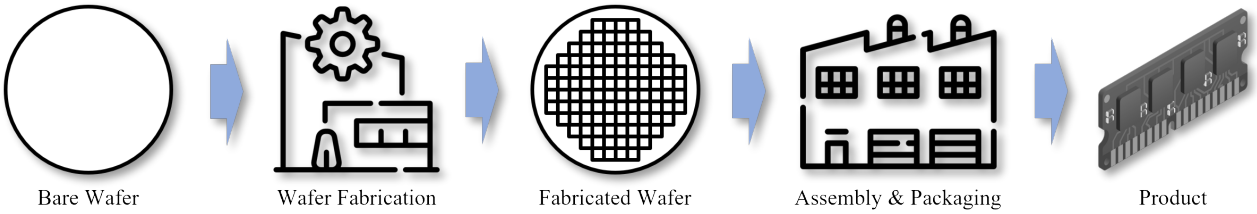

FIGURE 1. Common semiconductor product manufacturing processes using wafers.

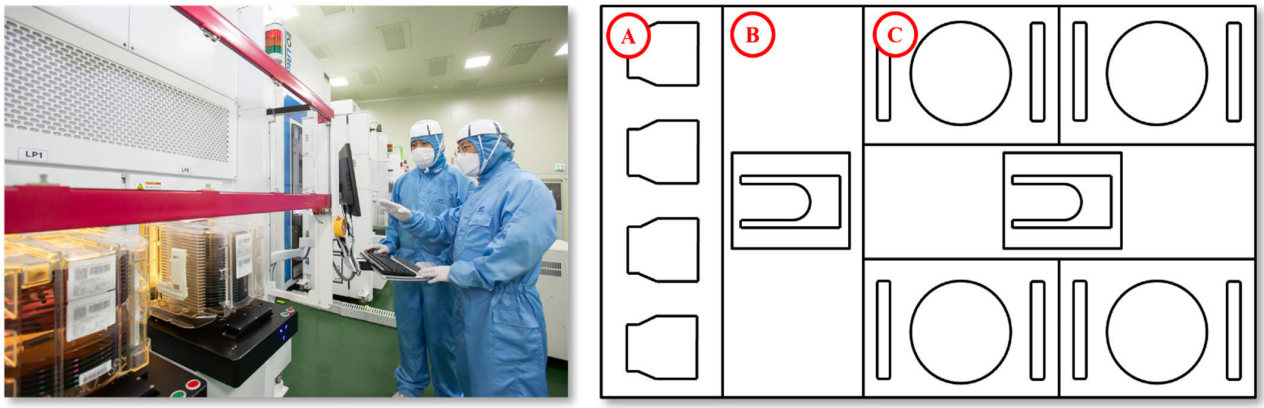

FIGURE 2. Cluster tool with engineers (a) and example configuration for the cluster tool (b).

used in the semiconductor industry, and the technological requirements for cluster tools to meet the miniaturization of circuit patterns owing to continuous advancements in design technology have also increased. Fig. 2 shows a photograph of a cluster tool and a diagram describing its typical configuration, which consists of a cassette module (A), transport module (B), and process module (C) [25]. The roles of these modules are as follows:

- Cassette module: This module is responsible for the input and output of wafers. It opens and closes the door of the FOUP transported through the OHT, transfers wafers stored in the FOUP to the cluster tool using a wafer transfer robot, or transfers processed wafers back to the FOUP.
- Transport module: This module is responsible for transferring wafers between the cassette and process modules using wafer transfer robots. Appropriate operational parameter settings are required because the performance and stability of wafer transfer robots significantly affect the productivity of cluster tools.
- Process module: This module is where the actual wafer processing takes place. It consists of spray nozzles for spraying chemicals or gases and a chuck table for fixing and rotating the wafers.

Fig. 3 [24] illustrates the wafer transfer model between cluster modules. The wafers that are initially input through the cassette module are transported by the transport module to the process module. The process module then performs the sequential steps defined in the process recipe.

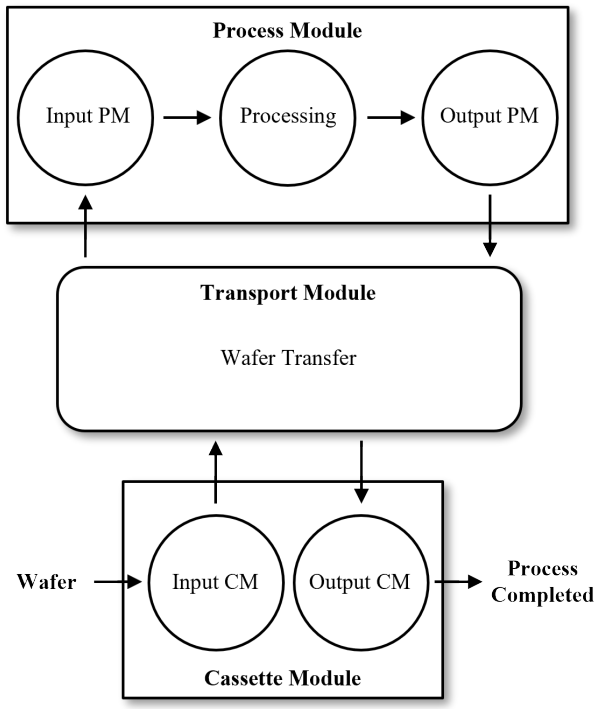

FIGURE 3. Wafer transfer model between cluster modules.

After the completion of the process, the wafers are transferred back to the cassette module via the transport module. Once this transfer process is completed for all the

wafers, the manufacturing sequence of the single-cluster tool ends.

Various studies have been conducted to understand and model the complex operations of cluster tool systems in semiconductor manufacturing. Some of the early work was performed using Petri nets, which are well suited for modeling parallel and distributed systems of cluster tools. In this context, Zuberek implemented the steady-state behavior of cluster tools using an iterative chamber revisiting scheduling [26], and Kim et al. studied scheduling methods within time constraints for cluster tools with dual arms [18]. Lee et al. conducted research on scheduling issues when multiple jobs were performed using batch wafer equipment [27].

However, Petri nets have difficulty analyzing factors that hinder the productivity of increasingly detailed cluster tools; therefore, studies have been conducted using finite state machines, a branch of automata theory. Lee and Lee modeled wafer scheduling techniques considering system time constraints using finite state machines [28], and Sun and Han built a simulation model based on finite state machines to measure the wafer throughput per unit time [29]. Hong and Lee conducted research to improve scheduling by avoiding unnecessary behaviors of the equipment front-end module (EFEM) using finite state machines and tick-based simulations [30].

Studies have also utilized simulations to predict the productivity of cluster tools. Hung and Leachman studied methods to evaluate the productivity of wafer manufacturing equipment in simplified simulation environments [14], and LeBaron and Pool implemented a simulator to predict the productivity of cluster tools using AutoMod [15]. Watanabe implemented a wafer transfer simulator based on spreadsheet applications [16], and Kohn and Rose conducted research to predict cluster tool processing times based on specific recipe combinations and parallel processing methods [13]. In addition, Tu built a productivity prediction model for cluster tools using multiple regression analysis and established regression equations based on the number of process recipes [12].

Previous studies on the productivity of cluster tools have been limited to module-level simulations rather than the detailed operating units of the cluster tool, or have focused on the impact of process recipes. In addition, research focusing on improving the scheduling methods of wafer transfer robots is not applicable to equipment that already operates using existing methods. Thus, existing studies are limited in that they only focus on certain elements of wafer processing using cluster tools rather than comprehensively considering the entire process.

Considering these limitations, this study aims to systematically analyze the factors that hinder productivity throughout the wafer manufacturing process using a deterministic finite state machine methodology and improve the related operational parameters. This approach not only focuses on specific modules or wafer transfer methods, but also comprehensively considers all operating units of cluster tools, providing insights into the complex causality occurring in the manufacturing process and contributing to overall productivity improvement.

B. EQUIPMENT ENGINEERING SYSTEM

The operational parameters of semiconductor manufacturing cluster tools consist of various factors from different perspectives, which are mainly distinguished into aspects concerning quality and those that enhance productivity. In terms of quality, factors that have a decisive impact on the final yield of wafers are relevant, such as the temperature of the operating units and the flow rate of raw materials. In terms of productivity, the set values of operating units, such as the speed of the cylinders and the torque of the motors, are the key components of the equipment. Proper control of these components is essential to optimize the performance of each module during wafer transfer and process progression, and to increase wafer productivity.

Operational parameters are usually detected in real time by sensors located at the end of the equipment or determined by preset values in hardware drivers. These data are reflected in the operational parameters set in the system through a communication module, which assists users in detecting and responding to abnormal situations by defining threshold values for individual items.

To eliminate causes of equipment failure and enhance overall equipment efficiency, an equipment engineering system (EES), one of several MESs, is utilized. The EES consists of various subapplications and provides various functionalities depending on user requirements and the environment. The main subapplications and their roles are as follows [31]:

- Fault detection and classification (FDC): Real time monitoring of equipment operational parameters to detect and control anomalies.
- Recipe management module (RMM): Monitors and manages process recipes.
- Alarm management module (ALM): Monitors and manages errors occurring in equipment.
- Equipment constant module (ECM): Manages the initial values set on the equipment.

With advances in semiconductor manufacturing technology, various studies have been conducted on systems to manage these manufacturing processes efficiently. Chien et al. presented an FDC framework for the anomaly detection and classification of semiconductors to monitor and analyze FAB profiles to reduce defects and unnecessary yield reduction [32]. Yasuda et al. studied FDC indicators that could detect abnormal conditions in equipment without chambers or process recipes [33]. Tsuda et al. analyzed the parameter data related to the product yield using FAB-wide systems [34]. Kim conducted research to improve the processing efficiency owing to increased data volume [35], and Kim et al. conducted research to build an equipment-level FDC system capable of analyzing anomalies on its own [36]. Zhang et al. conducted research to optimize

production planning based on data from a MES [37]; they utilized manufacturing- and logistics-related data rather than cluster tool data. These studies mainly focused on monitoring functions or data processing methods.

Considering these limitations, this study aims to perform manufacturing simulations and determine the optimal operational parameters based on a cluster tool implemented in a digital twin environment by utilizing the operational parameters and process recipe data of the cluster tool stored in the EES. The utilization of data from legacy systems is expected to enhance the reliability of the simulation results, ultimately aiming to provide suitable solutions for actual retrofitting and the improvement of cluster tools at manufacturing sites.

C. DIGITAL TWIN

The concept of a digital twin was initially introduced by Grieves in 2003 [38] and has garnered significant attention in both academia and industry, leading to continuous research efforts [39], [40], [41]. Grieves and Vickers defines a digital twin as a set of virtual information constructs that perfectly describe a real-world physical entity [20], and the terms ''digital twin'' and ''model'' are used distinctly depending on the presence of the actual physical entity [42]. Digital twins can be used to build real physical systems or processes in a digital space, monitor and analyze the status of real systems in real time through IoT [21], and solve problems in the product lifecycle by using data and simulations of real products and systems to predict and analyze their behaviors [43]. Thus, digital twins are expected to significantly impact the future of manufacturing by enabling a shift from analyzing the past to predicting the future.

Digital twin technology is already being utilized in various industries and various studies have been conducted to apply it in the semiconductor industry. Haddod and Dingli researched the implementation of digital twins in semiconductor FABs and the synchronization between actual production lines and developed digital twins [44]. Deenen et al. conducted research based on simulations using digital twins to resolve bottlenecks in wafer manufacturing [45]. Sivasubramanian et al. implemented a digital twin-based discrete-event simulation model to address reticle management issues in the photolithography process [46]. However, these studies either focused on production lines rather than individual equipment units or were driven by specific process data, making it difficult to extend them for application to all cluster tool models.

Research has also been conducted on the implementation of the operating units of cluster tools in virtual environments. Bratovanov developed a wafer pick-and-place robot simulator using SolidWorks and Visual Basic.NET, which simulates wafer handling by setting parameters such as the valve open/closed states of the end-effector/pre-aligner during the process [47]. Nicolescu et al. implemented a robot in a virtual environment using CATIA and DMU Kinematics, allowing the teaching of wafer transfer robots by setting parameters such as movement and velocity/acceleration of each axis of the robot via a teaching pad [48]. However, these studies mainly focused on implementing wafer transfer robots and did not reflect all the components of the entire equipment or were not based on actual manufacturing site data, resulting in insufficient simulation reliability.

Considering the limitations of the existing research, this study aims to model cluster tools in a virtual space using digital twins and perform manufacturing simulations in the same sequence as the actual processes. By visually representing the manufacturing simulation process in a three-dimensional (3D) environment using a base model implemented in a digital twin, we aim to enable users to identify and respond more effectively to potential risks that may occur during the process. Through this, we propose a new approach to cluster tool management at manufacturing sites and contribute to the digitalization and automation of manufacturing.

Table 1 presents a comparison of the objectives, methodologies, and tools used in previous studies, which were discussed in the research background and are closely related to this study.

III. RESEARCH METHODOLOGY

A. DIGITAL TWIN APPLICATION FRAMEWORK

The framework of existing MES is designed to collect data from manufacturing equipment via communication based on the SEMI equipment communications standard (SECS)/generic equipment model (GEM) protocol, process it, and present the results in visual forms such as tables and charts, enabling users to monitor and analyze data on equipment operational parameters in real time [49], [50]. However, these frameworks are limited to simply displaying data visually and do not provide advanced features, such as predictive analytics and optimization, which can limit the ability to increase the efficiency of manufacturing processes and proactively anticipate and respond to problems.

Therefore, based on the existing framework, the framework of the proposed digital twin application was designed as shown in Fig. 4. Digital twin applications utilize operational parameters and process recipe data stored in legacy systems to iteratively perform simulations and optimizations in a digital twin environment, and explore the optimal operational parameters to enhance the productivity of cluster tools. The data generated in this process are stored in the MES and utilized by users to improve the actual cluster tool.

The framework of the digital twin application proposed in this study consists of a digital twin module and an optimization module, where the digital twin module consists of a base model and a simulation engine, and the optimization module consists of an optimization engine and an interface. First, the base model is implemented in the digital twin environment as a representation of the cluster tool based on

TABLE 1. Literature review.

Reference	Objective	Methodology	Tool used
LeBaron *et al*. (1994)	Develop a flexible simulation model for cluster tools	General-purpose simulation language for model building	AutoMod simulation software
Lee *et al*. (2000)	Schedule and determine feasible process times for cluster tools	Modeling scheduling techniques considering system time constraints	Finite state machine-based custom-built scheduler
Kim *et al*. (2003)	Analyze scheduling for time-constrained cluster tools with diverse wafer flow patterns	Systematic modeling and analysis using timed Petri nets and linear programming	Custom-built software
Zuberek (2004)	Analyze cluster tools with chamber revisiting	Modeling and analysis using timed Petri nets	Timed Petri nets
Sun *et al*. (2005)	Measure wafer throughput and simulate sequence of cluster tools	Implementation of finite state machine-based simulation system	Custom-built simulation system
Lee *et al*. (2007)	Optimize scheduling for wet stations in cluster tools	Petri net modeling and mixed integer programming	Custom-built Petri net model
Nicolescu *et al*. (2009)	Develop virtual prototypes of robots for wafer manipulation and applications	3D CAD modeling, real-time robot programming and simulation	CATIA, DMU Kinematics
Kohn *et al*. (2011)	Develop an automated analytical process time models for cluster tools	Analytical modeling combining throughput and discrete event simulation	Real-world data analysis and simulation software
Watanabe (2015)	Develop a wafer transfer simulator for efficient wafer handling	Cellular automata-based modeling and simulation	Custom-built spreadsheet application
Tu, (2019)	Develop a throughput estimation model for cluster tools	Multiple regression analysis and simulation	eM-Plant simulation software
Hong *et al*. (2020)	Model, simulate, and control cluster tools	Extended finite state machine-based modeling and supervisory control	Tick-based simulation framework
Bratovanov *et al*. (2021)	Develop pick-and-place wafer handling robotics simulations in virtual environment	3D CAD modeling, proximity calculation, states monitoring	SolidWorks API, VB.NET
Haddod *et al*. (2021)	Develop a digital twin system for semiconductor manufacturing	Real-time data synchronization and 3D visualization	KAFKA, Unity3D
Kim *et al*. (2022)	Develop a system for cluster tool optimization and real-time fault detection	Data analysis and context mapping	.NET framework, PostgreSQL, JavaScript
Deenen *et al*. (2022)	Built a digital twin of semiconductor processes to validate improved production control	Discrete-event simulation based on real-world data, development of dispatching heuristic	Custom-built simulation model using C#
Kim *et al*. (2022)	Optimize throughput of cluster tool	Data collection and analysis, derivation of optimization parameters, ML-based algorithms (KNN)	Lam Research simulation software
Sivasubramanian *et al*. (2023)	Develop a digital twin-based approach for semiconductor processes	Digital twin and simulation-based rollout algorithm	Python, SimPy
This study	Propose a framework for determining the operational parameters of cluster tools using digital twin	Digital twin, deterministic finite state machine-based simulation, and hill climbing algorithm	Inventor, C#, Unity3D, Python, Anaconda3

data, such as equipment specifications, operation sequences, and operational parameters. The simulation engine performs actual manufacturing simulations by automating detailed state transitions based on the base model, event handling based on operational logic, and scenario execution using automation settings. The optimization engine utilizes the data generated by the simulation engine to create neighborhoods for multiple models based on metaheuristic algorithms, iteratively exploring and improving the operational parameters. While running these applications, visualization of the

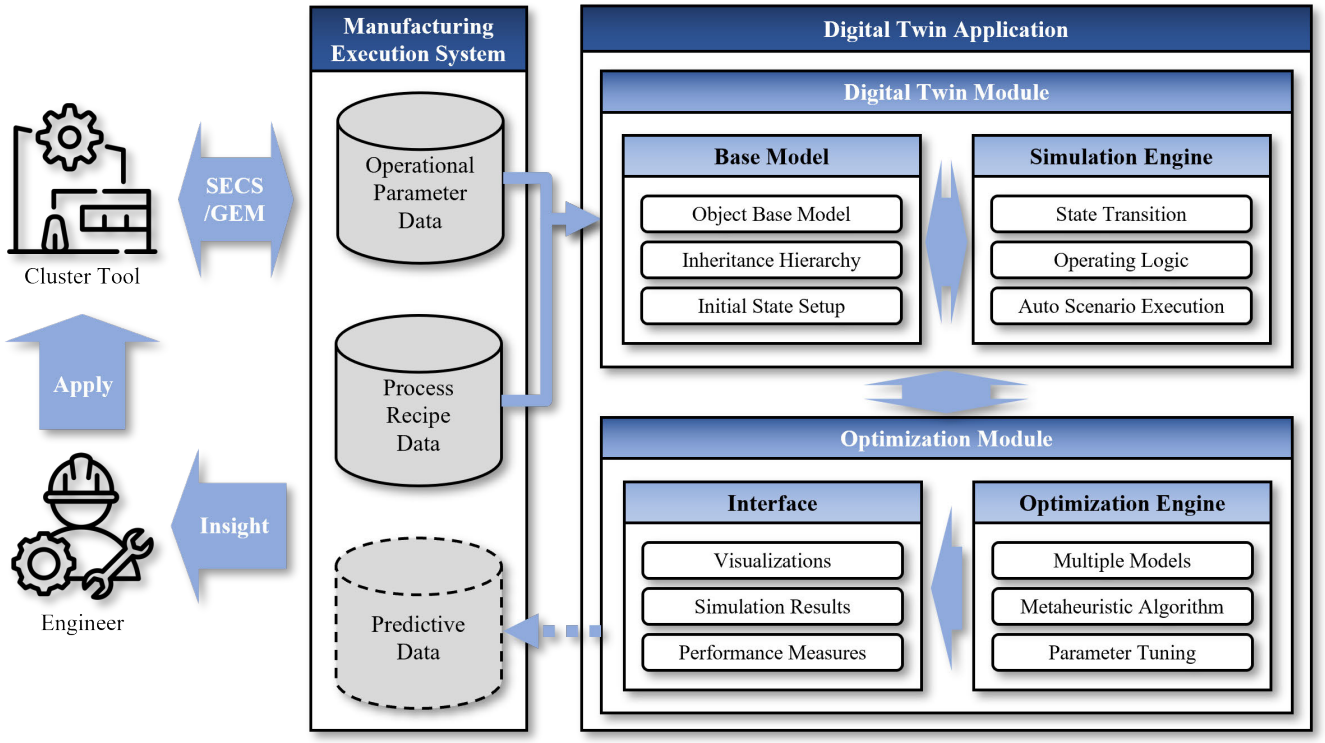

FIGURE 4. Framework of digital twin application.

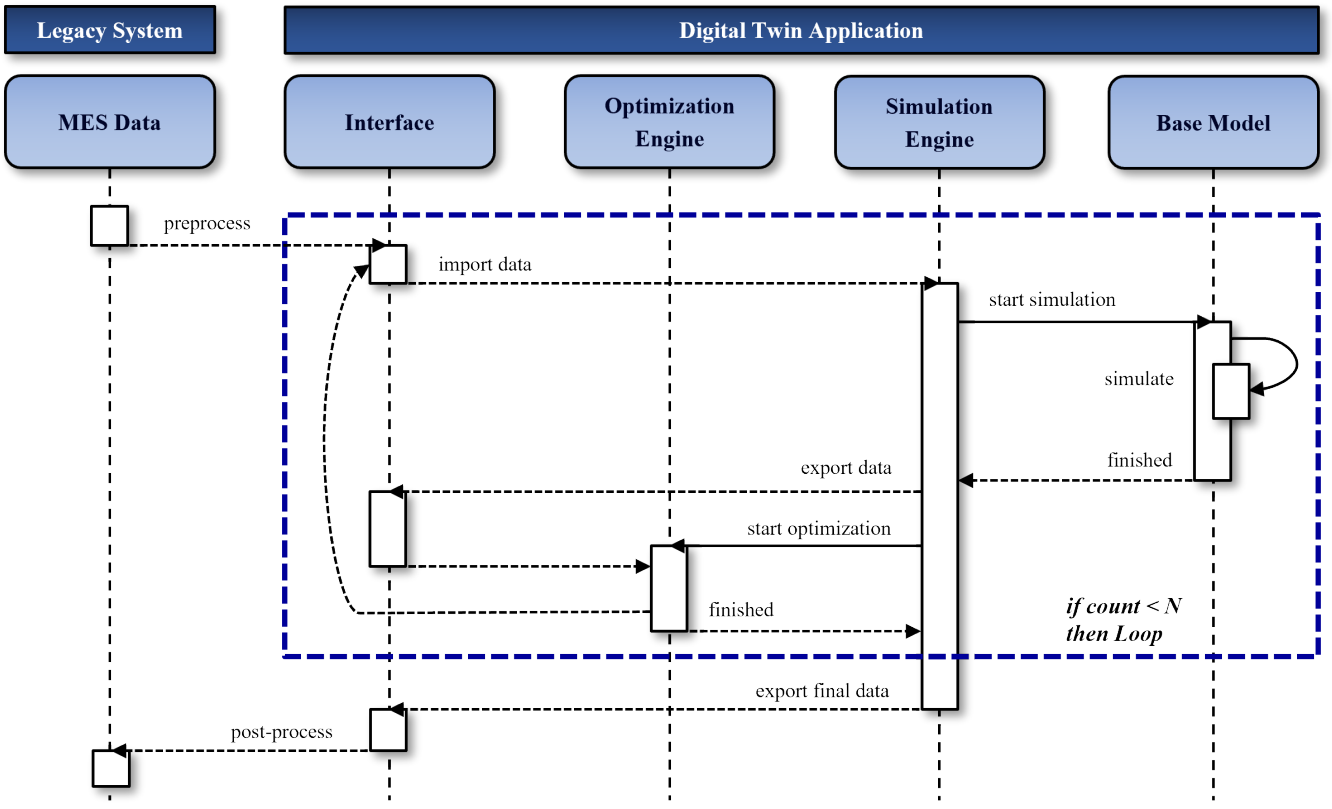

FIGURE 5. Operational sequence of digital twin applications.

manufacturing simulation and the input and output of the data are performed through interfaces.

Fig. 5 illustrates the operational sequence of this digital twin application. Initially, the operational parameter data of the cluster tool stored in the MES and the target process recipe data are processed and set as initial input values for the simulation to drive the base model. The simulation engine synchronizes these data, assigns them as the parameters of the subobjects that constitute the base model, and performs the simulation. During the simulation, users can observe virtual manufacturing activities in a digital twin environment from various perspectives through the interface. Upon completion of the simulation, the resulting data are output, and the optimization engine utilizes them to perform optimization tasks on the operational parameters. Once the optimization is completed, the simulation engine is initialized, and the optimized operational parameters are reassigned to the base model for simulation. This process is defined as an episode that is repeated as many times as the user sets during the application run. Upon reaching the specified count, the application operation is terminated, and the final resulting data is processed and stored in the MES. Subsequently, users can use these data for real-world business purposes.

B. DIGITAL TWIN MODULE

1) BASE MODEL

In the base model constituting the digital twin module, the cluster tool is implemented in the digital twin environment through an inheritance hierarchy structure, which is a hierarchical combination of modeled component-level objects. The inheritance hierarchy is a concept used in object-oriented programming to represent hierarchical relationships between parent and child objects [51]. A parent object has one or more child objects, and the child objects have one parent object. This hierarchical structure enables the representation of relationships such as inheritance, composition, and grouping among objects.

The hierarchical structure of this base model offers several advantages for effectively performing manufacturing simulations based on finite state machines. Firstly, the hierarchical structure allows for a clear definition of interactions between objects, enabling detailed modeling of the complex systems and state transitions within a cluster tool. This facilitates the implementation of realistic semiconductor manufacturing processes, thereby enhancing the accuracy and reliability of the simulations. Additionally, the hierarchical structure permits the independent definition and management of the state and behavior of each component within the cluster tool. This capability minimizes the impact on other components when a specific issue arises, enabling rapid identification and resolution of problems. Lastly, the hierarchical structure provides flexibility when adding new components or functions to the cluster tool. This flexibility allows for the expansion and modification of functionalities using the existing hierarchy, thereby enhancing the scalability and reusability of the simulation environment.

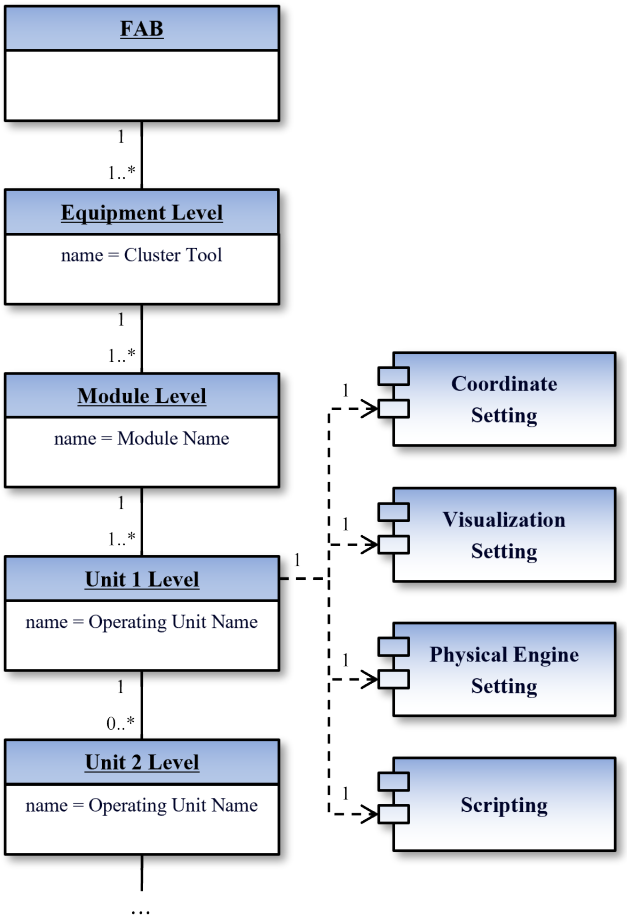

FIGURE 6. Example of the hierarchy of objects and the components assigned to them.

Fig. 6 illustrates the hierarchical structure of the objects constituting the cluster tool and examples of the assigned components. This hierarchy is organized into three levels: equipment, module, and unit, which are described as follows:

- Equipment level: Objects representing the cluster tool units that constitute the FAB are assigned. Objects at the equipment level control objects at the module level sequentially perform the processes from start to end of the manufacturing simulation. Multiple objects were deployed at the equipment level to enable the simulation of multiple operational parameters simultaneously.
- Module level: Objects representing the cassette, transport, and process modules constituting the cluster tool are assigned. The objects at the module level control multiple subunit-level objects to perform a series of operations, enabling each module to fulfill its role.
- Unit level: Assign component-level objects that constitute each module, perform actual operations, and distinguish between parent and child objects by assigning a sequence of orders. The operational parameters input to equipment-level objects are assigned to the

lowest unit-level objects, enabling the execution of operations based on the state transitions of objects during the manufacturing simulation.

Each object constituting this hierarchical structure was endowed with appropriate attributes through component-based software engineering (CBSE). CBSE is a methodology for building software systems by assembling reusable components [52], enhancing development productivity and maintainability through software modularization, and increasing code reusability to improve system flexibility and scalability [53]. Utilizing CBSE in digital twin applications can significantly enhance development productivity, maintainability, and system flexibility. Firstly, reusable components of the base model can be consistently applied across various simulation scenarios. Additionally, components can be easily updated and replaced independently, making maintenance easier. Lastly, the system's flexibility is augmented as new components can be added or modified in the base model without extensive redesign. These components define the characteristics and functionalities of the objects and coordinate the interactions between them to configure the operation of the entire system. Descriptions of each component assigned to the base model are as follows:

- Coordinate Setting: X, Y, and Z coordinates, along with rotation values, are configured to determine the position and orientation of objects within the 3D space. For instance, this ensures that specific modules of the cluster tool or wafers are accurately positioned and correctly oriented for operation.
- Physical Engine Setting: Physical properties such as collision, reaction, motion, and rotation are configured to simulate interactions between objects. For example, during the wafer transfer process, the actions of the cluster tool modules are simulated considering various physical factors affecting the wafer to achieve realistic simulations.
- Visualization Setting: Mesh rendering is used to determine how objects are displayed on the screen, improving the visual representation of objects within the simulation environment. For instance, each module of the cluster tool is configured to appear with textures and materials similar to the real ones.
- Scripting: Logic to control the behavior and state of objects is defined, programming the interactions and actions among objects. For example, the operational sequences of the objects comprising the cluster tool are implemented to ensure that the defined states transition appropriately during interactions.

2) SIMULATION ENGINE

Cluster tools for semiconductor manufacturing involve complex interactions between multiple process steps and equipment components. To effectively manage this complexity and implement the operational flow of the system, a finite state machine was utilized as the core methodology for designing the simulation engine. A finite state machine is a modeling technique that allows a system to transition between multiple states and perform specific behaviors in each state. This can be useful for tracing the operational flow of a system, identifying and solving problems by representing all possible situations, and determining how the system reaches these situations [54]. Moreover, finite state machines are highly scalable, and existing states and transitions can be extended to reflect the requirements as new sequences or components are added. In particular, deterministic finite state machines are finite state machines in which the next state is uniquely determined given discrete states and inputs, making them ideal for expressing the behavior of a system simply and clearly [55]. The elements constituting such deterministic finite state machines are defined by 5-tuples, as follows:

$$M = (Q, \Sigma, \delta, q_0, F) \tag{1}$$

The M stands for machine, where Q is the set of all possible states of a finite state machine, expressed as $Q = \{q_0, q_1, q_2, \ldots\}$, and Σ is the set of all possible input values that a finite state machine can recognize. $\delta : Q \times \Sigma \rightarrow Q$ is a state transition function, which determines the next state based on the current state and inputs. $q_0 \in Q$ is the initial state, the first state existing when the finite state machine starts, and $F \subseteq Q$ is the set of final states.

To model the cluster tool and the wafer to be manufactured in the simulation engine, five components of the deterministic finite state machine were defined as listed in Table 2. The operating unit of the cluster tool defines all the states in which it can act and performs wafer fabrication by changing its operational state in a prescribed sequence and considering the current state of the wafers. The wafers also define all the states that occur during the manufacturing process, transition through the wafer states based on the current position of the wafers and the behavior of the operating units, and the simulation ends when all wafers have transitioned to the end state. Fig. 7 illustrates an example of the state transition process between the operating units of the cluster tool and the wafers with each other's states as input.

The primary functions of the modeled simulation engine can be categorized into state transitions of each object, the execution of operating logic for modules and units comprising the equipment, and the automation of these processes through defined scenarios. The scenario is configured as the top-level class of the simulation engine, controlling subordinate classes to ensure that data input and output, manufacturing simulation, and optimization of operational parameters are executed automatically in a sequential manner. Under the control of this top-level scenario class, during the execution of a series of manufacturing simulations, the subordinate classes that make up the simulation engine perform their functions through scripting components assigned to each object of the base model. This process enables state transitions according to the defined operating logic and generates simulation data. Fig. 8 depicts the scope of implementing the main functions of the simulation engine based on the base model.

TABLE 2. Definition of deterministic finite state machine elements for modeling.

Finite state machine element	Operating unit of cluster tool	Wafer
Finite nonempty set of states	All operational states	All states during manufacturing progress
Initial state	Preoperational states	Located on the carrier before manufacturing
Final state	Postoperational states	Located on the carrier after manufacturing
Input alphabet	States of the wafers	Current position, states of the operating units
State transition function	Operational sequence	Behavior of operating units

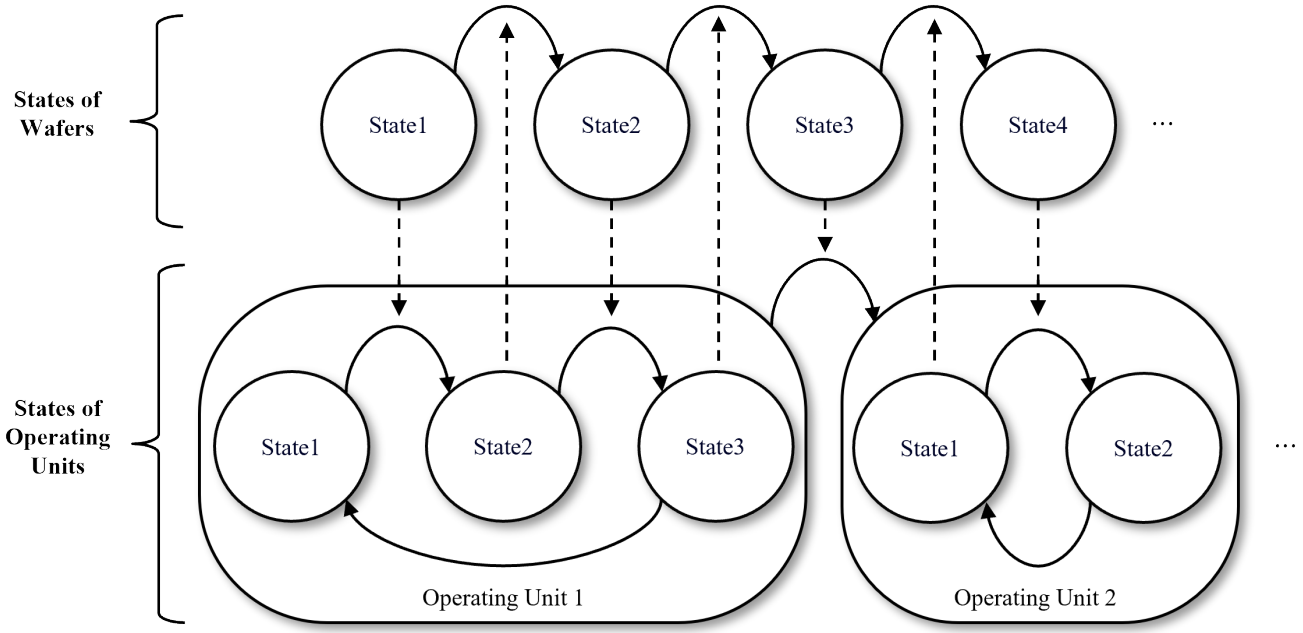

FIGURE 7. Example of state transitions between operating units and wafers.

During the simulation, the time required by each wafer in every state of the manufacturing process was measured and stored. The time required for each state is calculated based on the coordinates setting assigned to the object on the wafer, dividing the time for the object to move and the time for the object to rotate according to the state. First, to calculate the time for the object to move, the 3D vector of the i-th wafer in the j-th state is expressed as:

$$\mathbf{P}_{ij} = (x_{ij}, y_{ij}, z_{ij}) \tag{2}$$

Based on this, the Euclidean distance the i-th wafer travels from the j-th state to the $(j+1)$-th state is calculated as:

$$d_{ij} = \sqrt{(x_{i,j+1} - x_{ij})^2 + (y_{i,j+1} - y_{ij})^2 + (z_{i,j+1} - z_{ij})^2} \tag{3}$$

The driving speed of the actuator when the i-th wafer is in the j-th state is defined as v_{ij}, and the time taken for the i-th wafer to move in the j-th state is calculated as:

$$t_{ij} = \frac{d_{ij}}{v_{ij}} = \frac{\sqrt{(x_{i,j+1} - x_{ij})^2 + (y_{i,j+1} - y_{ij})^2 + (z_{i,j+1} - z_{ij})^2}}{v_{ij}} \tag{4}$$

To calculate the time for the object to rotate, the rotational Euler angles for each axis of the i-th wafer in the j-th state are expressed as:

$$\mathbf{R}_{ij} = (\theta_{ij}^x, \theta_{ij}^y, \theta_{ij}^z) \tag{5}$$

Based on this, the change in the rotational angle for each axis from the j-th state to the $(j+1)$-th state of the i-th wafer is calculated as:

$$\Delta\theta_{ij}^x = \left|\theta_{i,j+1}^x - \theta_{ij}^x\right| \tag{6}$$

$$\Delta\theta_{ij}^y = \left|\theta_{i,j+1}^y - \theta_{ij}^y\right| \tag{7}$$

$$\Delta\theta_{ij}^z = \left|\theta_{i,j+1}^z - \theta_{ij}^z\right| \tag{8}$$

The rotational speed of the actuator when the i-th wafer is in the j-th state is defined as ω_{ij}, and the rotational time for each axis is calculated as:

$$t_{ij}^x = \frac{\Delta\theta_{ij}^x}{\omega_{ij}} \tag{9}$$

$$t_{ij}^y = \frac{\Delta\theta_{ij}^y}{\omega_{ij}} \tag{10}$$

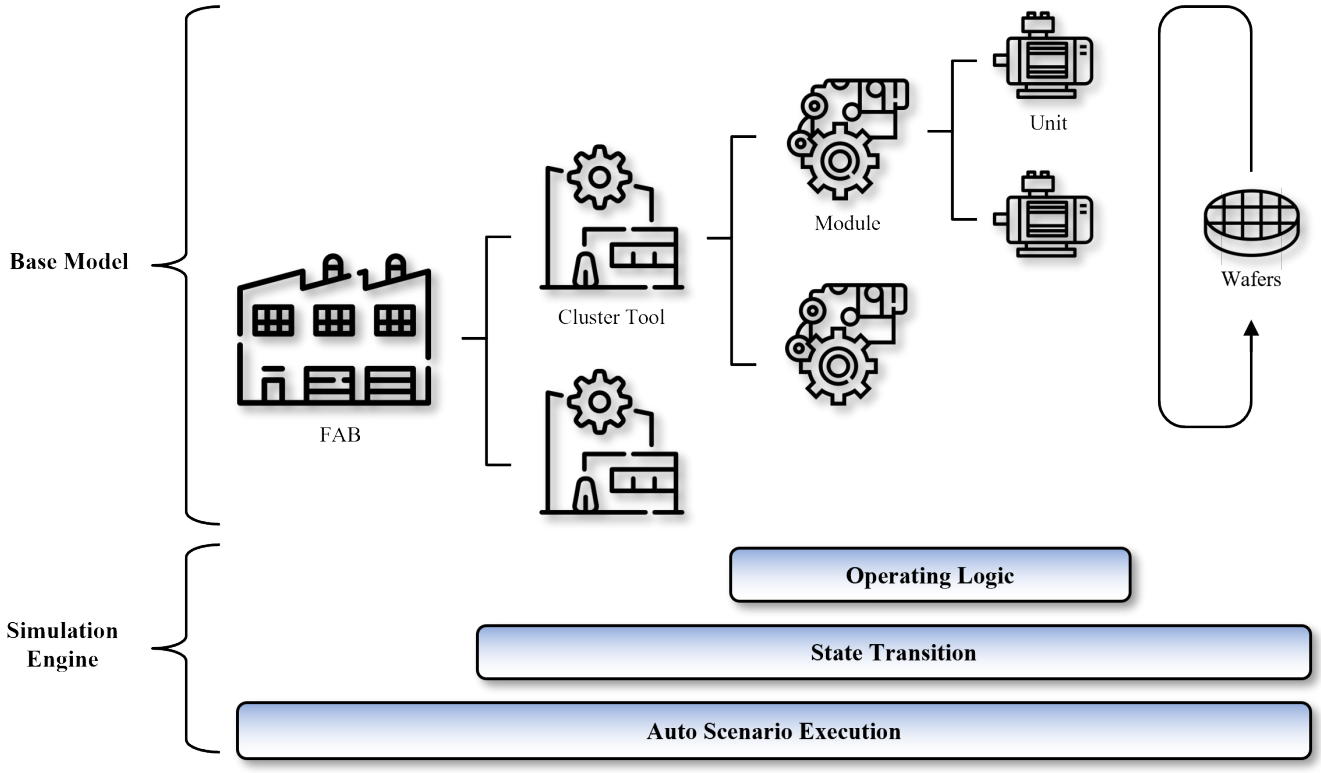

FIGURE 8. Scope of implementing the main functions of the simulation engine based on the base model.

$$t_{ij}^{z} = \frac{\Delta\theta_{ij}^{z}}{\omega_{ij}} \tag{11}$$

Based on the rotational time for each axis, the time taken for the i-th wafer to rotate in the j-th state is calculated by distinguishing between the cases where the rotations of the three axes occur simultaneously and sequentially:

$$t_{ij} = \begin{cases} \max(t_{ij}^{x}, t_{ij}^{y}, t_{ij}^{z}) & \text{if in simultaneous rotation} \\ t_{ij}^{x} + t_{ij}^{y} + t_{ij}^{z} & \text{if in sequential rotation} \end{cases} \tag{12}$$

To output these state-specific times as data, assume that a FOUP containing a total of n wafers is docked in the cluster tool and that each wafer goes through a total of m states before being processed and undocked. Through this process, all calculated wafer status times during the manufacturing of one FOUP by the cluster tool can be represented as an n row by m column matrix, as:

$$\textit{Wafer State Times} = \begin{bmatrix} t_{11} & t_{12} & t_{13} & \cdots & t_{1m} \\ t_{21} & t_{22} & t_{23} & \cdots & t_{2m} \\ t_{31} & t_{32} & t_{33} & \cdots & t_{3m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{n1} & t_{n2} & t_{n3} & \cdots & t_{nm} \end{bmatrix} \tag{13}$$

Let CT_i denote the time it takes for the i-th wafer to be fed into the cluster tool and complete manufacturing. This time is calculated to determine the duration of the manufacturing process for each wafer as:

$$CT_i = \sum_{j=1}^{m} t_{ij} \tag{14}$$

The time required to complete the manufacturing of all wafers is cycle time, which is the same as the time when the manufacturing of the last wafer (n-th wafer) ends and is calculated as:

$$CT = CT_n = \sum_{j=1}^{m} t_{nj} \tag{15}$$

In addition, the average time that n wafers spend in the j-th state of the cluster tool is termed ST_j, and it can be calculated to determine which state of the wafer manufacturing process requires the most time on average as:

$$ST_j = \frac{1}{n} \sum_{i=1}^{n} t_{ij} \tag{16}$$

Table 3 lists the notations of simulation results data.

C. OPTIMIZATION MODULE

In semiconductor manufacturing, the yield, which is the number of healthy chips actually produced as a percentage of the maximum number of chips designed on a wafer, is also affected by subtle factors such as errors in setting process parameters, poor facility management, and operator error [56]. Consequently, modifications and improvements to cluster tools require rigorous validation procedures, and production in the absence of such validation can

TABLE 3. Notation of simulation result data.

Notation	Definition
$\mathbf{P}$	3D vector of the wafer
$\mathbf{R}$	Euler angles of the wafer
θ	Rotational angle for each axis
d	Euclidean distance
v	Driving speed of the actuator
ω	Rotational speed of the actuator
i	Number of wafer ($i = 1,2,\dots,n$)
j	States of the wafer ($j = 1,2,\dots,m$)
m	Total number of all states the wafer has
n	Total number of wafers
t_{ij}	Time spent by the i-th wafer in the j-th state
CT_i	Cycle time of the i-th wafer
ST_j	Average time for n wafers to spend in the j-th state

critically affect the quality of an entire production line or product. To minimize such quality risks, this study aims to explore optimal operational parameters through incremental improvements based on existing operational parameters. This approach ensures the stability of the existing validated operational parameters while gradually enhancing equipment performance. Therefore, in this study, an optimization engine was designed based on the hill climbing algorithm, which is a metaheuristic algorithm.

The hill climbing algorithm searches for an optimal solution by exploring neighboring solutions relative to the current solution. Although it does not guarantee a global optimal solution, it effectively searches for local optimal solutions [57]. The hill climbing algorithm operates as follows:

1. Set the initial solution.
2. Generate neighboring solutions from the initial solution according to specific rules.
3. Compare the generated neighboring solutions with the initial solution, and update the current solution if a better solution is found.
4. Repeat this process until it converges to the optimal solution.

The key to this algorithm lies in the generation of neighboring solutions and the criteria for selecting a better solution. Neighbors can be generated in various ways, including by randomly selecting neighboring solutions or incrementally modifying the initial solution. The criterion for selecting a better solution varies depending on the nature of the problem; however, in most cases, it involves comparing objective function values of the solutions and selecting the solution with the smallest value.

To design an optimization engine based on the hill climbing algorithm to explore the optimal operational parameters for the cluster tool, the main components were defined as listed in Table 4. Because it is difficult to determine exactly how each component of the initial solution affects the objective function, various adjustments were made to the initial solution to generate multiple neighboring solutions. Subsequently, the same objective function was applied to the generated neighboring solutions to compare the results and expand the search process for the optimal solution in various ways to determine how the components of the initial solution affect the results of the objective function, which is expected to help us find a high-quality solution.

The optimization engine sets the existing operational parameters stored in the MES as the initial solution and searches for the optimal operational parameters. For this purpose, the base model of the digital twin module was set as the objective function, and a manufacturing simulation was performed based on the generated neighboring operational parameters.

To achieve this, the base model of the digital twin module is set as the objective function, and the generated neighboring solutions, which are the operational parameters, are used as simulation input data for manufacturing simulation. The output data from the simulation includes the input operational parameters and resulting cycle time values. The optimization engine selects the neighboring solution with the shortest cycle time from these results, i.e., the improved operational parameters and updates the current solution accordingly. When configuring the input data for the next simulation episode based on the improved operational parameters, analyzing the impact of each operational parameter item on the simulation results is possible. To achieve this, multiple neighboring operational parameters are generated by sequentially adjusting the values of each item in the current and newly generated operational parameters. The generated data are then utilized as input data for subsequent simulation episode. Fig. 9 illustrates an example of the data configuration for updating the current solution and generating neighboring solutions.

This series of processes is defined as one episode, and the optimal operational parameters are derived by repeating a specified number of episodes. Fig. 10 illustrates the process of the optimization engine searching for the optimal operational parameters based on the hill climbing algorithm.

IV. APPLICATION APPROACH

A. SCENARIO AND APPLICATION OF DIGITAL TWIN FRAMEWORK

The scenario configuration for optimizing the operational parameters of the cluster tool defined in this study is shown in Fig. 11. During the scenario execution, the cumulative episode count increased for each episode. At the beginning of an episode, the simulation engine assigns the initial input data as parameters to the base model and performs virtual manufacturing. After completing all manufacturing activities, the simulation engine merges the assigned parameters with the simulation results and outputs them. Subsequently, based

TABLE 4. Definition of hill climbing algorithm elements for optimization.

Algorithm element	Optimization engine design element
Initial solution	Existing operational parameters stored in the manufacturing execution system
Current solution	Improved operational parameters
Neighboring solution	Operational parameters generated by current solution
Optimal solution	Optimal operational parameters
Objective function	Manufacturing simulation using a cluster tool implemented based on a digital twin
Objective function result	Cycle time of manufacturing for 1 FOUP

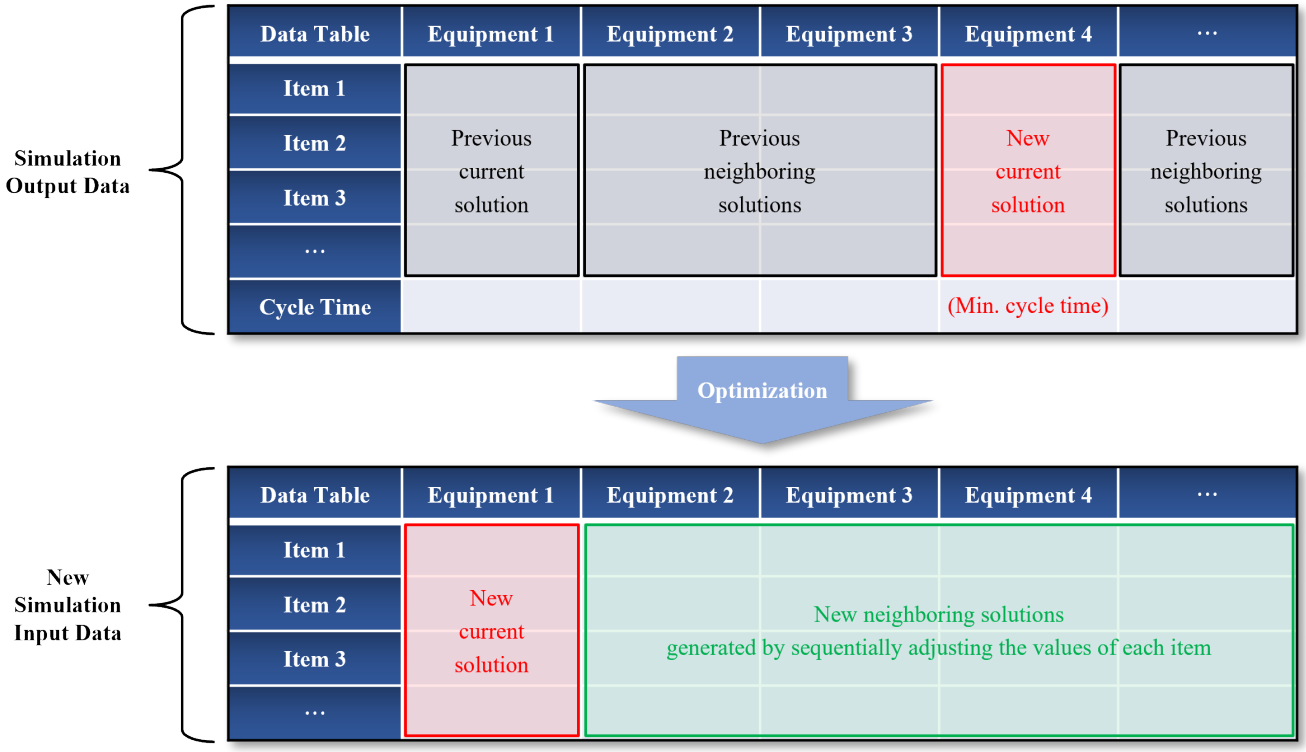

FIGURE 9. Data configuration example for updating the current solution and generating neighboring solutions.

on the simulation results, the optimization engine performs the optimization tasks, initializes the existing simulation environment for the next episode, and ends the episode. The simulation engine determines whether the current episode has reached the number of times specified by the user; if not, it returns to the step before starting the episode and performs the next episode; if it has, it outputs the final result to end the optimization scenario of the operational parameters.

This simulation-optimization iterative structure is implemented based on the repetitive search process of the hill climbing algorithm. It is designed to perform a sufficient number of iterations, as set by the user, to evaluate the impact of changes in each component of the operational parameters on the cycle time and to identify the key operational parameters.

The implementation environment of the digital twin application proposed in this study is listed in Table 5. Each unit component constituting the cluster tool was modeled using Autodesk Inventor Professional 2024, CAD software, and saved in STL file format. Subsequently, Blender, a 3D graphics production software, was used to render the STL files to match the texture and color of the real-world objects, converting them into DAE files, which were then stored in a Unity environment. Unity implements the base model through the inheritance hierarchy and component settings of the stored objects, and the simulation engine is implemented through object-oriented programming using C# in Visual Studio 2022 based on the base model. The optimization engine was implemented using Python in the Visual Studio Code environment and executed in the Anaconda3 virtual environment. The input and output files were in the form

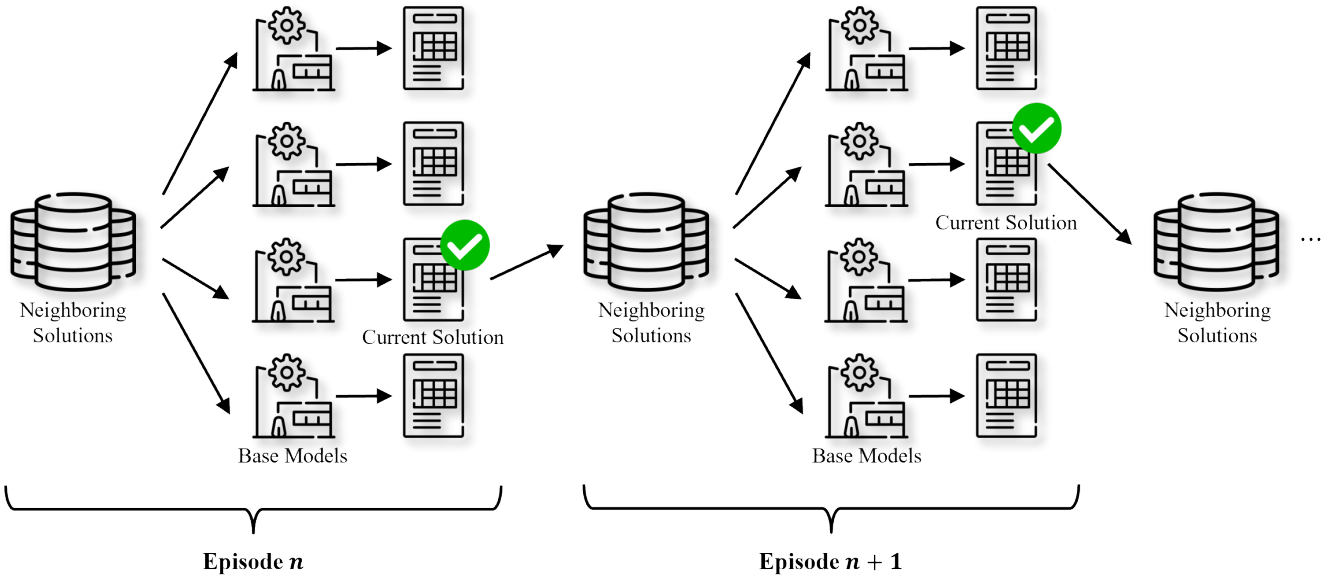

FIGURE 10. Sequence to explore optimal operational parameters based on a hill climbing algorithm.

TABLE 5. Implementation environment for digital twin applications.

Module	Component	Item	Contents
Digital twin module	Base model	CAD software	Autodesk Inventor 2024
		CAD file format	STL file
		Modeling software	Blender V3.4.1
		Modeling file format	DAE file
	Simulation engine	Development software	Unity 2022.3.10f1
		Development environment	Visual Studio 2022 V17.7.34031
		Programming language	C#
Optimization module	Optimization engine	Development environment	Visual Studio Code V1.87.2
		Programming language	Python V3.11.5
		Virtual environment	Anaconda3 V23.7.4
	Interface	Input/Output data format	CSV file
		User interface	Unity UI

of CSV files, and the user interface of the application was implemented using Unity UI.

1) IMPLEMENTATION APPROACH FOR THE DIGITAL TWIN MODULE

To construct the backbone of the digital twin module, which is the base model, each modeled object was arranged according to the appropriate hierarchy and the components required for the object were clearly defined. First, to adjust the position and rotation of the objects, the location and rotation information of the components in 3D space are input based on actual equipment specifications. Additionally, the visualization settings were set to consider the material, color, shadows, and reflections of real objects. Setting up the physics engine involves specifying details such as gravity, air resistance, collision detection, and reaction to enable interactions between objects, thereby enhancing the reliability of the simulation. Finally, object-oriented programming is used to implement the behavior, state, and detailed settings for the interaction between objects to control their behavior and state. Fig. 12 shows an example of the hierarchy and component setting information of the base model implemented in this study. Tables 6–9 show the details of the information model and the contents used to explain each item.

The simulation engine performed manufacturing simulations based on this implemented base model. To achieve this, we implemented a previously defined scenario and combined classes to perform the detailed procedures of the scenario. The class configuration of the simulation engine

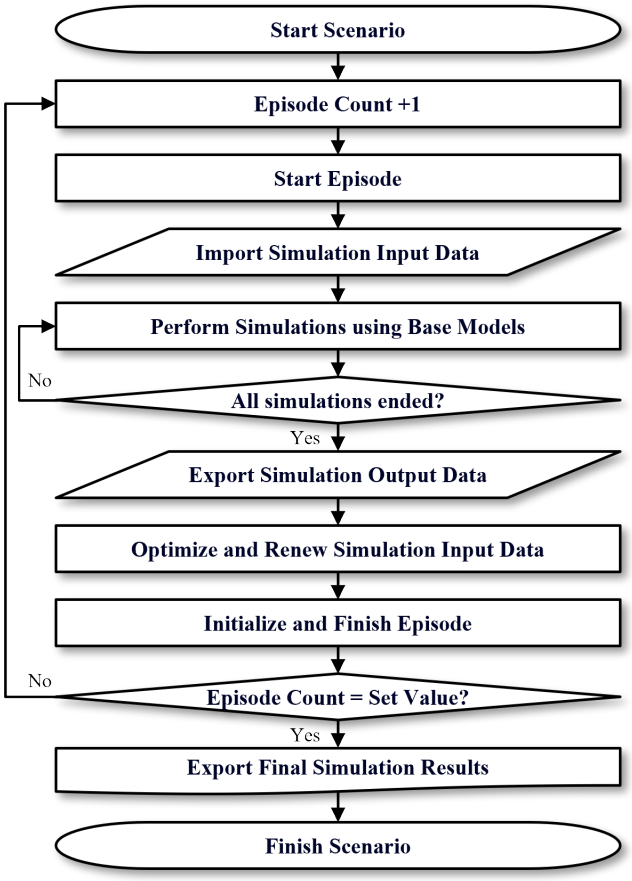

FIGURE 11. Scenario diagrams to optimize operational parameters.

TABLE 6. Schema of coordinates component in the base model.

Element	Unit	Type	Definition
X coordinate	mm	Float	Coordinate value on the X-axis
Y coordinate	mm	Float	Coordinate value on the Y-axis
Z coordinate	mm	Float	Coordinate value on the Z-axis
X rotation	...°	Float	Rotation value from the X-axis
Y rotation	...°	Float	Rotation value from the Y-axis
Z rotation	...°	Float	Rotation value from the Z-axis

implemented in this study is shown in Fig. 13. The four main classes and their roles are as follows:

- Scenario Manager: A class that performs the predefined simulation scenario.
- Job Manager: A class that performs manufacturing simulations based on the base model of the cluster tool.
- Data Input/Output Manager: A class that inputs and outputs the data required for the simulation.
- Optimization Manager: A class that calls the optimization engine.

During virtual manufacturing activities performed through the Job Manager class, the wafer state transitions through the current wafer position and operation state of the cluster tool components, and the time spent in each state is measured and stored as the simulation results. For this purpose, as shown in Fig. 14, the wafer transitions its state by determining whether it interferes with trigger cubes installed in each component of the cluster tool.

2) IMPLEMENTATION APPROACH FOR THE OPTIMIZATION MODULE

The optimization engine of the optimization module was run at the end of each episode to optimize the operational parameters. This optimization process is based on the hill climbing algorithm, which sets the operational parameter with the smallest cycle time from the simulation result data as the optimal solution. The optimal solution is determined based on the simulation input data, and neighboring solutions are generated by multiplying the optimal solution by a certain value. During the generation of neighboring solutions, the ratios of the optimal and neighboring solutions are sequentially adjusted to assess how each operational parameter affects productivity enhancement. Table 10 lists the pseudocode of the hill climbing algorithm for performing the optimization process. The next episode of the manufacturing simulation was performed based on the simulation input data generated by the optimization engine. Fig. 15 illustrates the data input and output processes during the simulation and optimization processes.

B. IMPLEMENTATION AND RESULTS OF THE DIGITAL TWIN APPLICATION

To demonstrate the applicability and validity of the proposed framework, an example of implementing the digital twin application is presented. For security reasons, instead of using actual semiconductor manufacturing equipment, virtual equipment simulating the manufacturing process was implemented based on publicly available patent data. During the simulation process, each module of the cluster tool executed only one operation at a time, and unexpected events such as equipment failures were not considered. In addition, a manufacturing simulation of a single process recipe was performed for 25 wafers, and the subunits of the parallel process module were assigned the same operational parameters. The reason for assigning identical operating parameter values to the sub-units of the parallel process module is that performance differences between chambers can affect wafer yield. To ensure consistent product quality, chamber matching is essential, which involves minimizing parameter differences between chambers [58]. In manufacturing environments, permissible ranges for these operating parameters are established, and sensors are used to maintain consistent parameter values to ensure pattern quality [59]. Additionally, standardization through chamber matching contributes to improved equipment management and operational efficiency.

An example screen of the digital twin application implemented in this study is shown in Fig. 16. A represents the

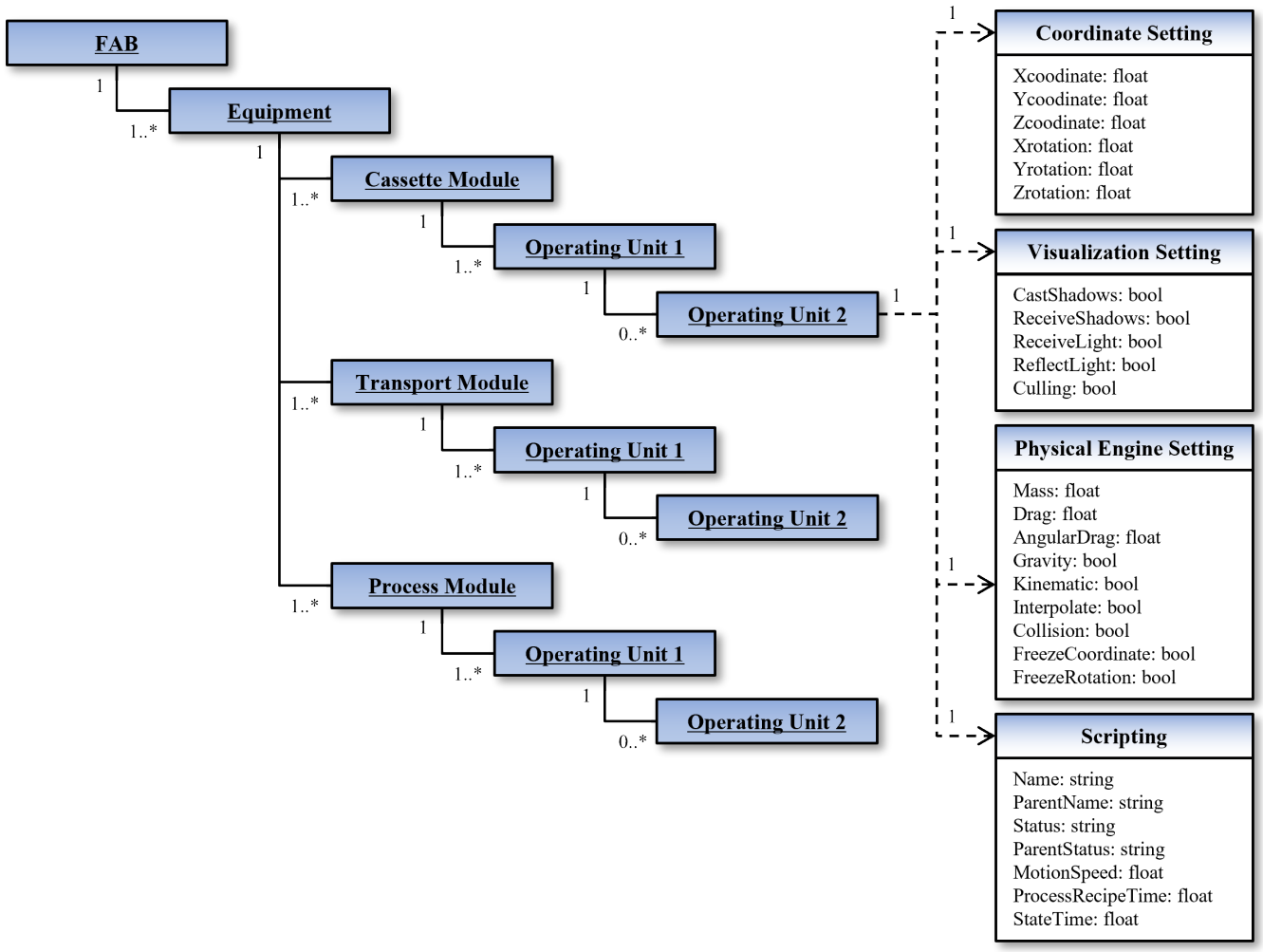

FIGURE 12. Example hierarchy and component setting for implementing a base model.

TABLE 7. Schema of visualizations component in the base model.

Element	Unit	Type	Definition
Cast shadows	-	Bool	Whether shadows are cast by the object
Receive shadows	-	Bool	Whether an object shows the shadow of another object
Receive light	-	Bool	Whether the object is affected by other light sources
Reflect light	-	Bool	Whether the object reflects light from other light sources
Culling	-	Bool	Whether to invalidate visualization when an object is blocked from the view

digital twin module that performs manufacturing simulation based on the base model of the cluster tool. In this example, 16 machines are deployed to consider the graphics resources of the computer. B is the user interface screen, where users can control the camera position to visualize the digital twin environment through buttons. The current progress of the episode, simulation status of each piece of equipment, and optimization status are visualized using LED indicators. C is the optimization module that optimizes the operational parameters considering the cycle time.

When the digital twin application was executed, operational parameters and process recipes were set as simulation input data, and the manufacturing simulation was conducted by assigning these parameters to each base model of the cluster tool. Subsequently, the optimization module explores the operational parameters with the highest productivity based on the stored simulation result data and generates operational parameters for the next episode based on this, updating the simulation input data. This process was repeated until a user-defined number of episodes was reached, and

TABLE 8. Schema of physics component in the base model.

Element	Unit	Type	Definition
Mass	kg	Float	Mass of the object
Drag	N	Float	Set air resistance or friction for vertical section
Angular drag	N	Float	Set air resistance or friction for rotation section
Gravity	-	Bool	Whether to apply gravity to objects
Kinematic	-	Bool	Whether the physics engine controls the object
Interpolate	-	Bool	Whether to interpolate physics updates using the physics engine
Collision	-	Bool	Whether collisions with other objects are detected
Freeze X coordinate	-	Bool	Whether the X coordinate is fixed
Freeze Y coordinate	-	Bool	Whether the Y coordinate is fixed
Freeze Z coordinate	-	Bool	Whether the Z coordinate is fixed
Freeze X rotation	-	Bool	Whether the X rotation is fixed
Freeze Y rotation	-	Bool	Whether the Y rotation is fixed
Freeze Z rotation	-	Bool	Whether the Z rotation is fixed

TABLE 9. Schema of scripting component in the base model.

Element	Unit	Type	Definition
Name	-	String	Name of the object
Parent name	-	String	Name of the parent object
Status	-	String	Status of the object
Parent status	-	String	Status of the parent object
Motion speed	mm/s	Float	The speed at which an object performs an action
Process recipe time	s	Float	Time to perform the process
State time	s	Float	The amount of time an object spends in a particular state

TABLE 10. Pseudocode for a hill-climbing algorithm to optimize operational parameters.

Hill climbing algorithm for operational parameters

```
procedure HillClimbing(CandidateSolutions)
    MinCycleTime ← infinity
    CurrentSolution ← null
    NeighboringSolutions ← empty list
    for each CandidateSolution in CandidateSolutions do
        if CycleTime(CandidateSolution) < MinCycleTime then
            MinCycleTime ← CycleTime(CandidateSolution)
            CurrentSolution ← CandidateSolution
    for each Parameter in CurrentSolution do
        NewNeighboringSolution ← GenerateNeighboringSolution(CurrentSolution, Parameter)
        append NewNeighboringSolution to NeighboringSolutions
    return NeighboringSolutions
```

the accumulated simulation results and optimal operational parameters were utilized for business purposes.

An experiment was conducted to determine the operational parameters using a previously implemented digital twin application. In this experiment, 100 simulation-optimization episodes were repeated to select the operational parameters with the smallest cycle time, and the accumulated simulation data were analyzed. The initial operational parameters

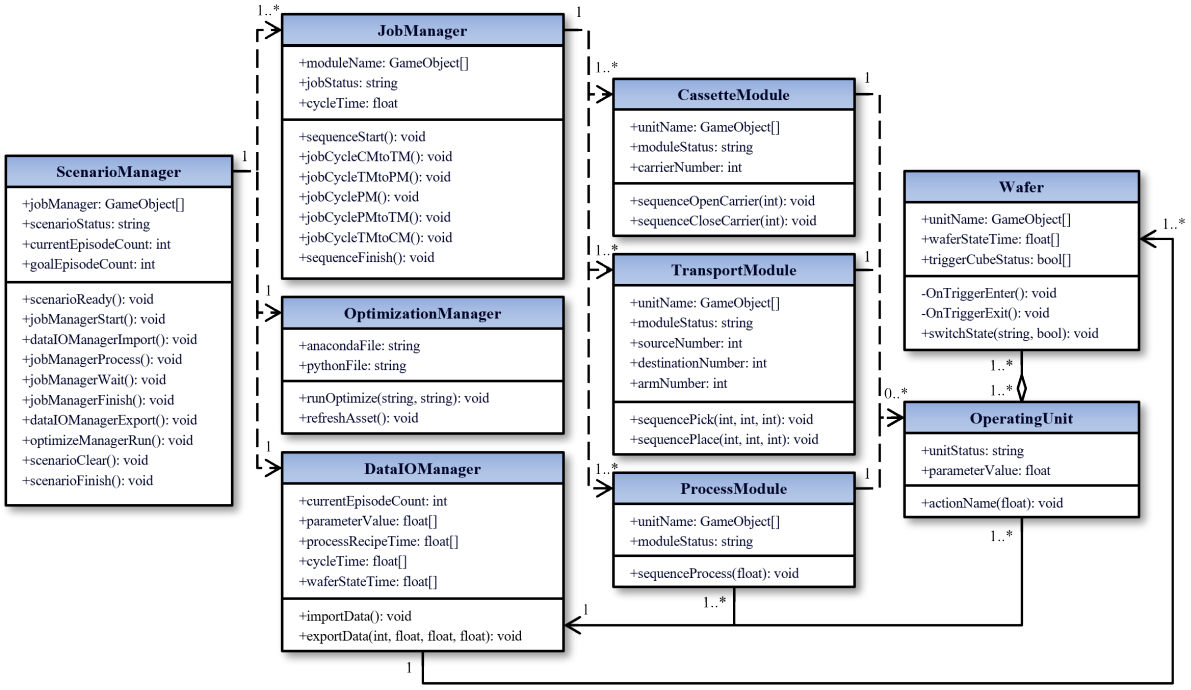

FIGURE 13. Class organization of a simulation engine.

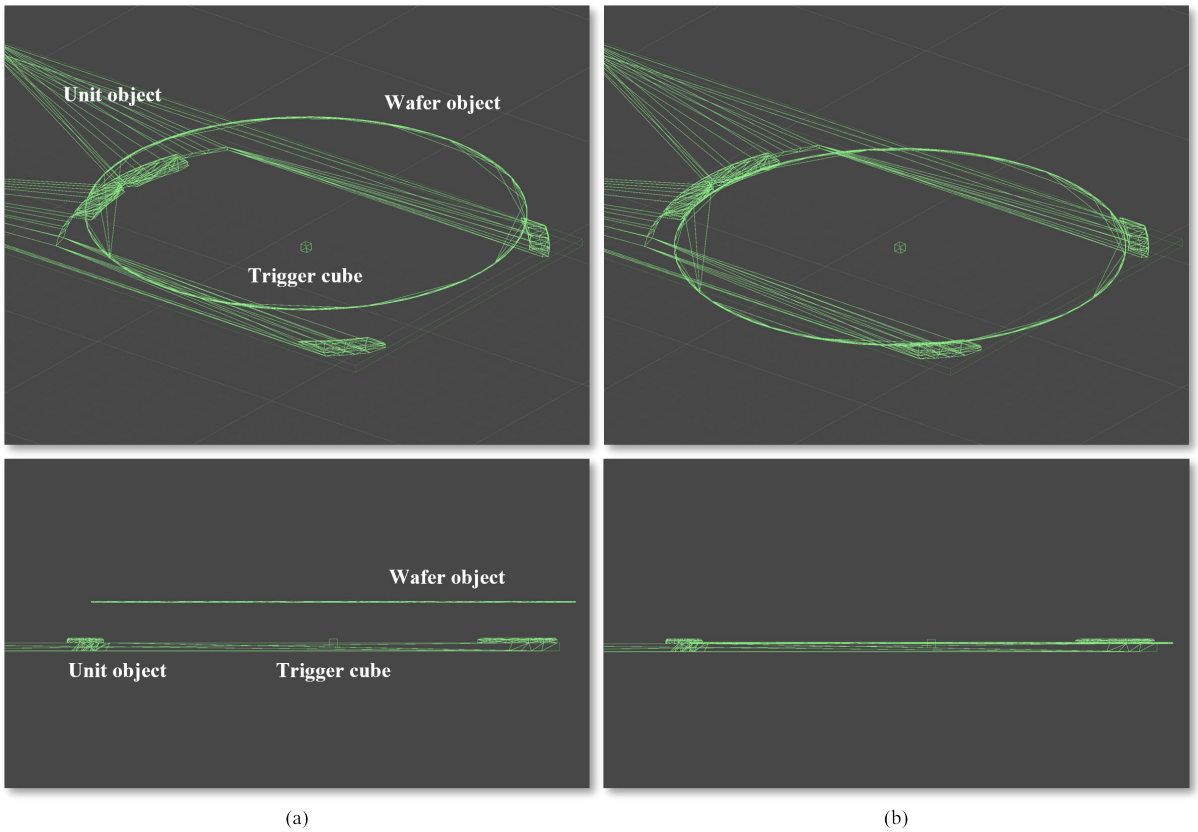

FIGURE 14. Example of setting up state transitions for undetected (a) and detected (b) wafers using a trigger cube.

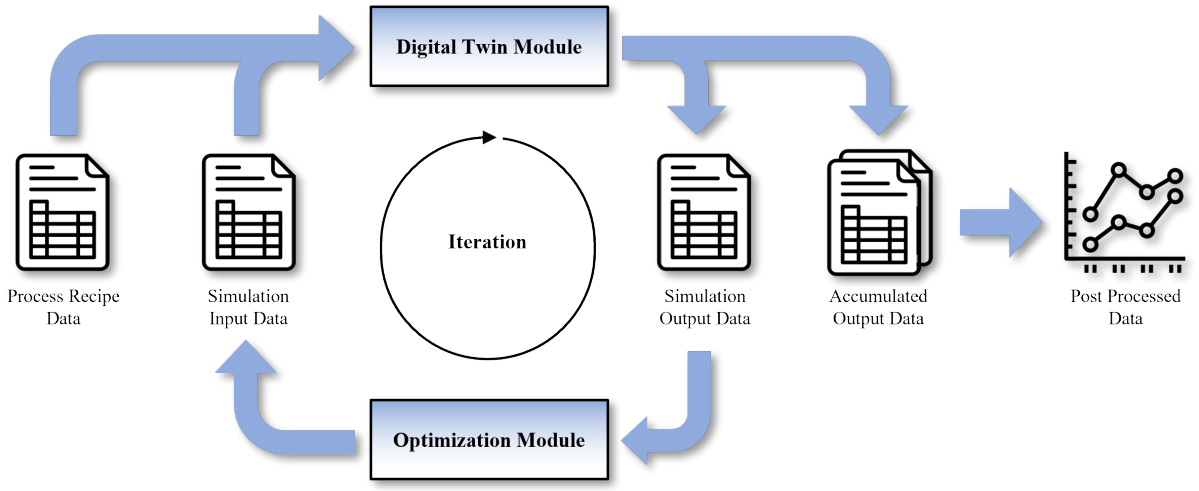

FIGURE 15. Data input/output process in simulation and optimization.

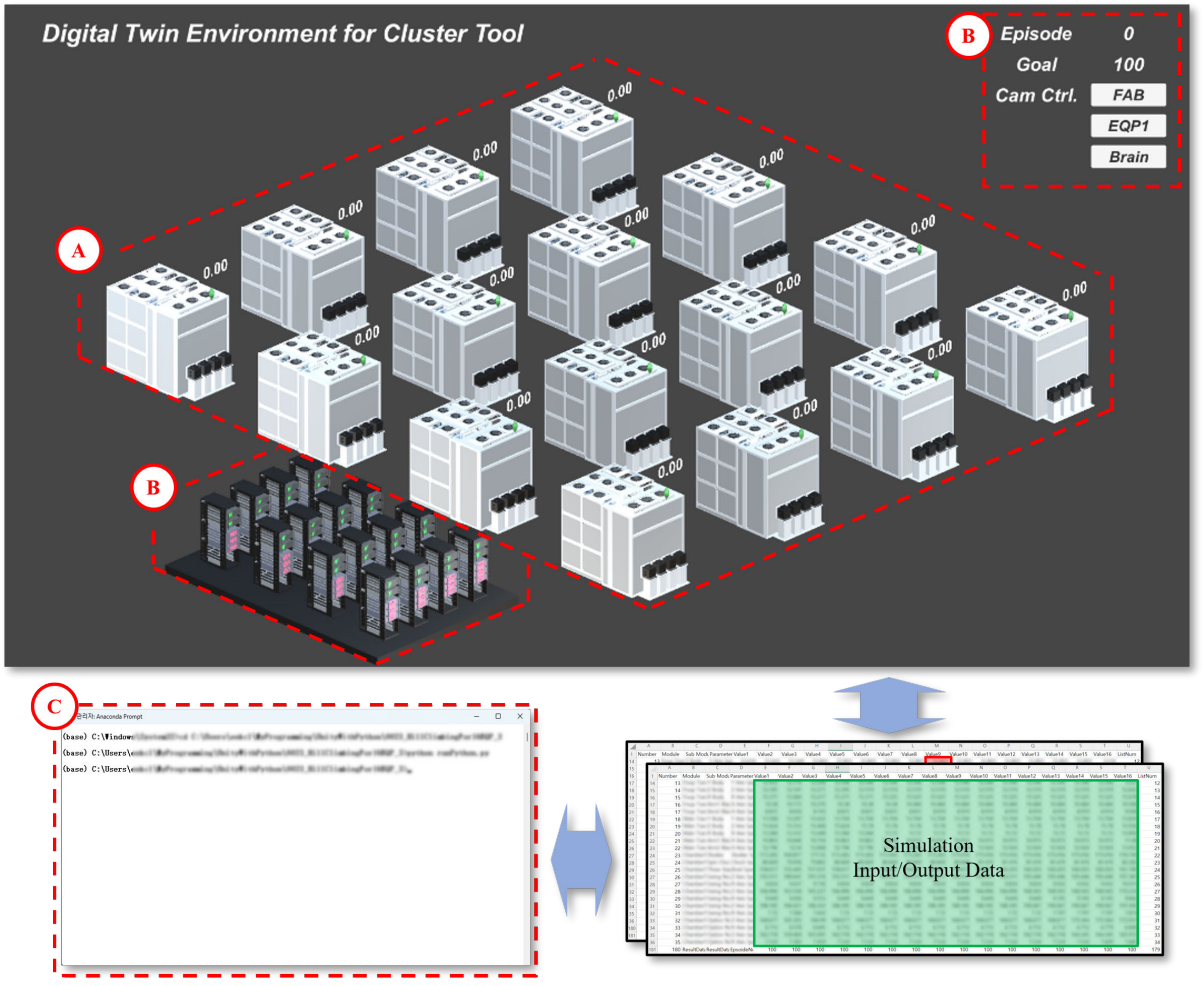

FIGURE 16. Example of a screen from an implemented digital twin application.

consisted of the speed settings of the actuators comprising the transport and process modules of the cluster tool, with a total of 23 items. In the process of generating neighboring solutions using the hill climbing algorithm, values ranging from 98% to 102% of the initial solution were sequentially arranged, as shown in Fig. 17.

TABLE 11. Results of optimizing operational parameters.

Item	As is	To be	Fluctuations
Cycle time of manufacturing for 1 FOUP (s)	750.50	441.75	-41.14%
Average speed of the actuators in the transport module (mm/s)	493.00	1258.84	+155.34%
Average speed of the actuators in the process module (mm/s)	55.25	102.66	+85.79%

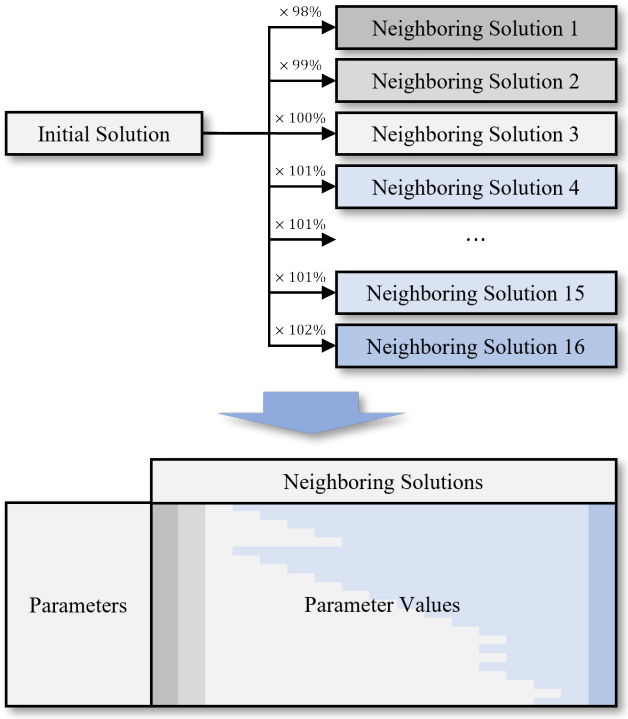

FIGURE 17. Creating neighboring solutions via cascading arrays.

Fig. 18-(a) shows a graph with the arithmetic mean of the actuator speed settings on the x-axis and the resulting cycle time on the y-axis. The optimal solution for a single episode is marked in red while neighboring solutions are marked in blue. Fig. 18-(b) presents a graph with the number of episodes on the x-axis, and the corresponding cycle time and arithmetic mean of the actuator speed settings on the y1- and y2- axes, respectively. In the initial phase of the optimization scenario, as the number of episodes increases, the average actuator speed settings tend to increase, resulting in a decrease in cycle time. However, after the number of episodes exceeds approximately 70, the average actuator speed settings exhibit unstable fluctuations with a slight upward trend, and the cycle time converges to approximately 440 seconds.

To understand the impact of individual operational parameters on the reduction in cycle time, a graph was plotted with the sequence number of operational parameter items on the x-axis, cycle time on the y-axis, and the ratio of the current value to the initial value of the actuator speed settings on the z-axis, as shown in Fig. 19. This graph shows that the actuator speed settings were evenly scattered until the cycle time converged; however, as the cycle time converged, the speed settings of certain actuators increased sharply.

Table 11 presents the optimization results for the operational parameters obtained through this demonstration. "As is" represents the results corresponding to the initial operational parameters, whereas "To be" represents the results corresponding to the operational parameters with the lowest cycle time among the 100 episodes. The cycle time improved by 41.14% compared to the previous time, with the average speed setting of the actuators composing the transport module increasing by 155.34%, and the average speed setting of the actuators composing the process module increasing by 85.79%.

Studies on the improvement of wafer productivity in cluster tools have been conducted on different equipment models and processes, making direct comparisons challenging. However, from the perspective of wafer productivity improvement, the 41.14% cycle time reduction achieved in this study significantly surpasses the 10.00% improvement achieved by LeBaron et al. through step time optimization [15], the 6.90% improvement achieved by Watanabe through the elimination of misunderstanding logics [16], the 3.07% improvement achieved by Kim et al. through the optimization of scheduling variables [19], and the 13.45% improvement achieved by Sivasubramanian et al. through reticle management using a rollout policy [46]. These results suggest that the methodology proposed in this study may be more effective in improving wafer productivity.

Several significant experimental results were observed in the example of determining the operational parameters. First, the cycle time tended to converge after a certain point as the simulation-optimization iterations were repeated. This result shows that simple changes in the operational parameters are not sufficient to sustainably improve productivity. This suggests that productivity improvement requires a comprehensive consideration of various factors, such as equipment specifications and wafer transfer scheduling, in addition to the operational parameters.

Furthermore, in the simulation-optimization process for the example equipment model, it was found that the operational parameters of the transport module had a greater impact on improving the cycle time after a certain point than those of the process module. Through this analysis, it is expected that operational parameters that have a greater impact on productivity can be identified not only for this model but also in the process of determining operational parameters for other equipment. This can provide users with valuable insights into

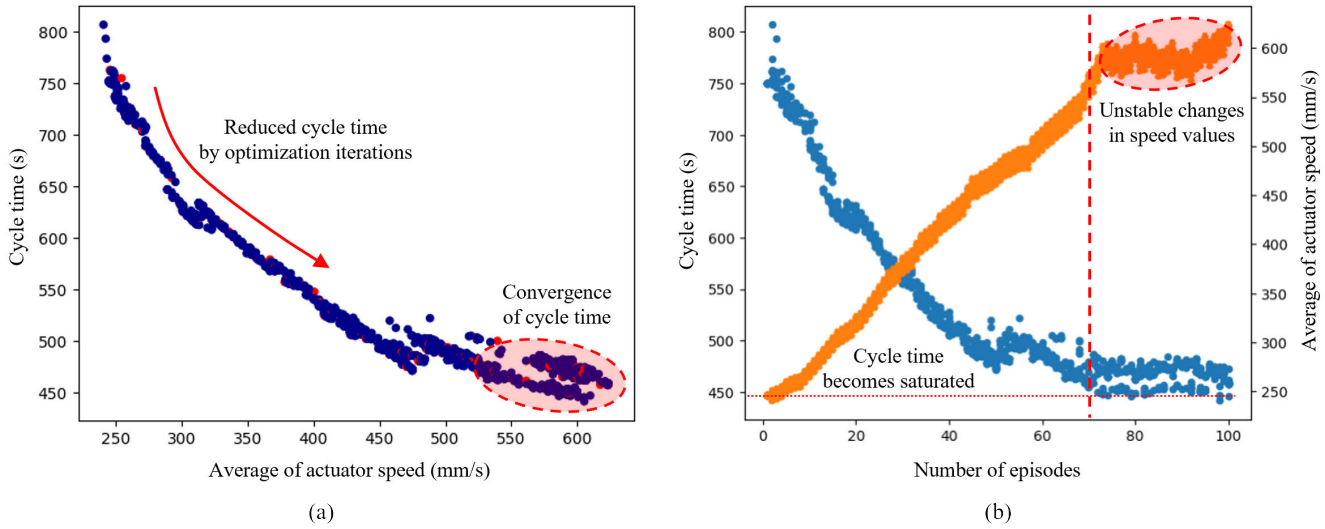

FIGURE 18. Graph of cycle time vs. average of actuator speed (a) and Graph of number of episode vs. cycle time and average of actuator speed (b).

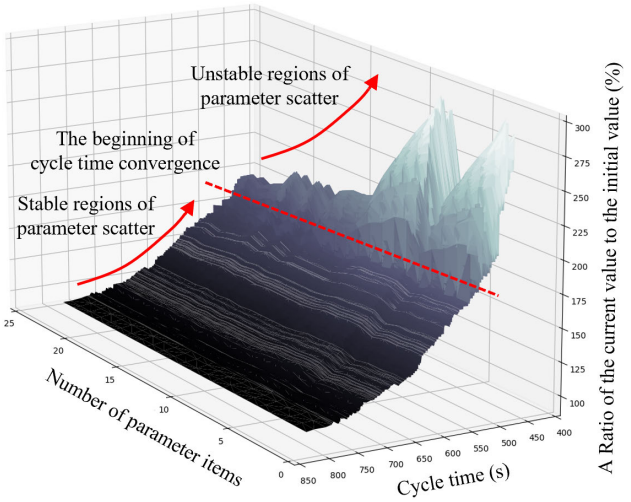

FIGURE 19. Graph of parameters, cycle time, and a ratio of current value to an initial value.

formulating operational strategies for actual equipment and laying the foundation for optimizing production processes. Moreover, it is expected to minimize the wastage of human resources and operational inefficiencies in the improvement of actual equipment.

V. CONCLUSION

Technological advancements brought about by the Fourth Industrial Revolution require large amounts of data processing and analysis, driving the demand for related semiconductor products. In this context, the importance of manufacturing and supply chain management in the semiconductor industry is emphasized, and semiconductor manufacturers focus on efficient production management and technological development. Cluster tools are essential equipment in the semiconductor manufacturing process, and various research and improvement efforts are being made to utilize them efficiently. However, efficient operation and management are still a challenge owing to various physical and system limitations. Therefore, there is an increasing need for a system that predicts and optimizes productivity based on the actual operational parameters of cluster tools.

As an alternative solution to these problems, this study proposed a framework for determining the operational parameters of semiconductor manufacturing cluster tools using digital twins and demonstrated its implementation through a specific example. Through manufacturing simulations and operational parameter optimization based on digital twins, we were able to significantly reduce the cycle time of cluster tools in wafer fabrication.

These research findings extend beyond mere productivity enhancement, considerably influencing the development of operational strategies for actual equipment and optimization of production processes. First, by utilizing digital twin-based simulations to pre-test and optimize real processes, on-site trial and error can be minimized. This approach reduces the risks faced by engineers due to the use of hazardous chemicals and prevents unexpected equipment failures or product quality degradation. Furthermore, by optimizing based on verified MES data rather than relying on the expertise of equipment manufacturers, it contributes to the establishment of operational strategies for actual equipment, using cycle time improvement data derived from changes in operational parameters. Ultimately, this is expected to contribute to cost reduction and improved efficiency in human resource allocation in manufacturing environments.

Furthermore, by visually clarifying the manufacturing simulation process of cluster tools through the application of digital twin technology, it is anticipated that users will be provided with insights into the complex interactions that occur within cluster tools.

This study has some limitations that need to be addressed in future research. First, the example of the base model implemented in the digital twin application was based on publicly available patent data for security reasons; thus, it requires validation through the application of actual equipment to increase the reliability of simulation results. Second, the range of input operational parameters may not encompass all factors affecting productivity in wafer manufacturing. Conducting research that considers comprehensive factors such as the physical limitations of equipment specifications and productivity changes depending on wafer transfer scheduling methods would yield more reliable results. Third, this study is limited to a single process. It is necessary to analyze and generalize the experimental results in various situations, such as when different process recipes are performed for different wafers on the same carrier [60]. Finally, this study did not consider the impact of operational parameter changes on component failures or defects. Excessive use of components may not only lead to component failure due to deterioration but also reduce the reliability of the equipment and eventually affect wafer quality due to physical factors. Considering these factors will lead to further empirical research.

REFERENCES

[1] J. Voas, N. Kshetri, and J. F. DeFranco, "Scarcity and global insecurity: The semiconductor shortage," *IT Prof.*, vol. 23, no. 5, pp. 78–82, Sep. 2021.

[2] W. Mohammad, A. Elomri, and L. Kerbache, "The global semiconductor chip shortage: Causes, implications, and potential remedies," *IFAC-PapersOnLine*, vol. 55, no. 10, pp. 476–483, 2022.

[3] L. Mönch, J. W. Fowler, and S. J. Mason, "Semiconductor manufacturing process description," in *Production Planning and Control for Semiconductor Wafer Fabrication Facilities: Modeling, Analysis, and System*, vol. 52. New York, NY, USA: Springer, 2013, pp. 11–28.

[4] P. H. Wald and J. R. Jones, "Semiconductor manufacturing: An introduction to processes and hazards," *Amer. J. Ind. Med.*, vol. 11, no. 2, pp. 203–221, Jan. 1987.

[5] B. Jean-Luc and D. Bruno, "Contamination monitoring and analysis in semiconductor manufacturing," in *Semiconductor Technologies*. Rijeka, Croatia: Intech, 1999, pp. 57–78.

[6] W. M. Moreau, "Optical exposure," in *Semiconductor Lithography: Principles, Practices, and Materials*. New York, NY, USA: Springer, 1988, pp. 355–408.

[7] S.-L. Chung and M. Jeng, "Manufacturing execution system (MES) for semiconductor manufacturing," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, vol. 4, Oct. 2002, pp. 1–5.

[8] L. Hsieh and T.-J. Hsieh, "A throughput management system for semiconductor wafer fabrication facilities: Design, systems and implementation," *Processes*, vol. 6, no. 2, p. 16, Feb. 2018.

[9] S. R. Ab Rahim, G. Haw, W. J. Lee, and O. Diehl, "Application of a simulation model to forecast cycle time based on static model input," in *Proc. Winter Simulation Conf. (WSC)*, Dec. 2022, pp. 3372–3381.

[10] L. M. Wein, "Scheduling semiconductor wafer fabrication," *IEEE Trans. Semicond. Manuf.*, vol. SM-1, no. 3, pp. 115–130, Aug. 1988.

[11] Y.-M. Tu, "Short-term scheduling model of cluster tool in wafer fabrication," *Mathematics*, vol. 9, no. 9, p. 1029, May 2021.

[12] Y. M. Tu, "Throughput estimation model of cluster tool in semiconductor manufacturing," *Key Eng. Mater.*, vol. 814, pp. 196–202, Sep. 2019.

[13] R. Kohn and O. Rose, "Automated generation of analytical process time models for cluster tools in semiconductor manufacturing," in *Proc. Winter Simulation Conf. (WSC)*, Dec. 2011, pp. 1803–1815.

[14] Y.-F. Hung and R. C. Leachman, "Reduced simulation models of wafer fabrication facilities," *Int. J. Prod. Res.*, vol. 37, no. 12, pp. 2685–2701, Aug. 1999.

[15] H. T. LeBaron and M. Pool, "The simulation of cluster tools: A new semiconductor manufacturing technology," in *Proc. Winter Simulation Conf.*, Dec. 1994, pp. 907–912.

[16] H. Watanabe, "Development of wafer transfer simulator based on cellular automata," *IEEE Trans. Semicond. Manuf.*, vol. 28, no. 3, pp. 283–288, Aug. 2015.

[17] S. Hur, H. Lee, Y. J. Park, and S. H. Yang, "A scheduling method with wafer transfer delay in the cluster tool process," *J. Korean Prod. Oper. Manage. Soc.*, vol. 22, no. 4, pp. 417–430, Dec. 2011.

[18] J.-H. Kim, T.-E. Lee, H.-Y. Lee, and D.-B. Park, "Scheduling analysis of time-constrained dual-armed cluster tools," *IEEE Trans. Semicond. Manuf.*, vol. 16, no. 3, pp. 521–534, Aug. 2003.

[19] Y. Kim, G. Lee, and J. Jeong, "ML-based JIT1 optimization for throughput maximization in cluster tool automation," *Appl. Sci.*, vol. 12, no. 15, p. 7519, Jul. 2022.

[20] M. Grieves and J. Vickers, "Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems," in *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*. Cham, Switzerland: Springer, 2017, pp. 85–113.

[21] K. Ding, F. T. S. Chan, X. Zhang, G. Zhou, and F. Zhang, "Defining a digital twin-based cyber-physical production system for autonomous manufacturing in smart shop floors," *Int. J. Prod. Res.*, vol. 57, no. 20, pp. 6315–6334, Oct. 2019.

[22] K.-M. Choi, J.-E. Lee, K.-Y. Cho, K.-S. Kim, and S.-H. Cho, "Clean room structure, air conditioning and contamination control systems in the semiconductor fabrication process," *J. Korean Soc. Occupational Environ. Hygiene*, vol. 25, no. 2, pp. 202–210, Jun. 2015.

[23] M. Brain, R. Gould, U. Kaempf, and B. Wehrung, "Emerging needs for continuous flow FOUP transport," in *Proc. 24th IEEE/CPMT Int. Electron. Manuf. Technol. Symp.*, Oct. 1999, pp. 76–82.

[24] H. M. Lee and T. J. Chung, "The cluster controller," *Electron. Telecommun. Trends*, vol. 13, no. 1, pp. 29–40, Feb. 1998.

[25] M. Dümmler, "Semiconductor manufacturing and cluster tools," in *Modeling and Optimization of Cluster Tools in Semiconductor Manufacturing*. Würzburg, Germany: Institut für Informatik, Lehrstuhl für Verteilte Systeme, 2004, pp. 5–28.

[26] W. M. Zuberek, "Cluster tools with chamber revisiting-modeling and analysis using timed Petri nets," *IEEE Trans. Semicond. Manuf.*, vol. 17, no. 3, pp. 333–344, Aug. 2004.

[27] T.-E. Lee, H.-Y. Lee, and S.-J. Lee, "Scheduling a wet station for wafer cleaning with multiple job flows and multiple wafer-handling robots," *Int. J. Prod. Res.*, vol. 45, no. 3, pp. 487–507, Feb. 2007.

[28] H. Y. Lee and T. E. Lee, "Scheduling and determination of feasible process times for CVD cluster tools with a dual end effector," in *Proc. Korean Oper. Res. Manage. Sci. Soc. Conf. Paper*, vol. 1, Apr. 2000, pp. 107–110.

[29] B. K. Sun and K. R. Han, "Study on measurement of wafer processing throughput and sequence simulation of SWP (single wafer process) cleaning equipment," *Inst. Electron. Eng. Korea Comput. Inform.*, vol. 42, no. 5, pp. 31–40, Sep. 2005.

[30] C. Hong and T.-E. Lee, "Modeling, simulation and supervisory control of semiconductor manufacturing cluster tools with an equipment front-end module," in *Proc. IEEE 16th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2020, pp. 703–709.

[31] D. G. Chung, "Equipment engineering," in *MES Technology for Smart Manufacturing*. Gyeonggi-do, South Korea: Hanul Academy, 2022, pp. 232–264.

[32] C.-F. Chien, C.-Y. Hsu, and P.-N. Chen, "Semiconductor fault detection and classification for yield enhancement and manufacturing intelligence," *Flexible Services Manuf. J.*, vol. 25, no. 3, pp. 367–388, Sep. 2013.

[33] S. Yasuda, T. Tanaka, M. Kitabata, and Y. Jisaki, "Chamber and recipe-independent FDC indicator in high-mix semiconductor manufacturing," *IEEE Trans. Semicond. Manuf.*, vol. 34, no. 3, pp. 301–306, Aug. 2021.

[34] T. Tsuda, S. Inoue, A. Kayahara, S.-I. Imai, T. Tanaka, N. Sato, and S. Yasuda, "Advanced semiconductor manufacturing using big data," *IEEE Trans. Semicond. Manuf.*, vol. 28, no. 3, pp. 229–235, Aug. 2015.

[35] G.-B. Kim, "Design and implementation of a hybrid equipment data acquisition system (HEDAS) for equipment engineering System(EES) framework," *J. Korea Soc. Comput. Inf.*, vol. 17, no. 2, pp. 167–176, Feb. 2012.

[36] N. Kim, H. Choi, J. Chun, and J. Jeong, "Introduction of equipment level FDC system for semiconductor wet-cleaning equipment optimization and real-time fault detection," in *Proc. 33rd Annu. Semi Adv. Semiconductor Manuf. Conf. (ASMC)*, May 2022, pp. 1–4.

[37] Z. Zhang, G. Yeming, and G. Zailin, "Clearing function-based simulation optimization for release planning under digital twin wafer fabs," *IFAC-PapersOnLine*, vol. 55, no. 10, pp. 2539–2544, 2022.
[38] M. Grieves, "Digital twin: Manufacturing excellence through virtual factory replication," *White Paper*, vol. 1, pp. 1–7, Mar. 2014.
[39] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital twin in industry: State-of-the-art," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2405–2415, Apr. 2019.
[40] Y. H. Son, G.-Y. Kim, H. C. Kim, C. Jun, and S. D. Noh, "Past, present, and future research of digital twin for smart manufacturing," *J. Comput. Design Eng.*, vol. 9, no. 1, pp. 1–23, Dec. 2021.
[41] Y. Lu, C. Liu, K. I.-K. Wang, H. Huang, and X. Xu, "Digital twin-driven smart manufacturing: Connotation, reference model, applications and research issues," *Robot. Comput.-Integr. Manuf.*, vol. 61, Feb. 2020, Art. no. 101837.
[42] L. Wright and S. Davidson, "How to tell the difference between a model and a digital twin," *Adv. Model. Simul. Eng. Sci.*, vol. 7, no. 1, pp. 1–13, Dec. 2020.
[43] A. Rasheed, O. San, and T. Kvamsdal, "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, vol. 8, pp. 21980–22012, 2020.
[44] F. Haddod and A. Dingli, "Intelligent digital twin system in the semiconductors manufacturing industry," in *Artificial Intelligence in Industry 4.0: A Collection of Innovative Research Case-Studies That are Reworking the Way We Look at Industry 4.0 Thanks to Artificial Intelligence*. Cham, Switzerland: Springer, 2021, pp. 99–113.
[45] P. C. Deenen, R. A. M. Adriaensen, and J. W. Fowler, "Building a digital twin of the photolithography area of a real-world wafer FAB to validate improved production control," in *Proc. Winter Simulation Conf. (WSC)*, Dec. 2022, pp. 3394–3405.
[46] C. K. Sivasubramanian, R. Dodge, A. Ramani, D. Bayba, M. Janakiram, E. Butcher, J. Gonzales, and G. Pedrielli, "DTFab: A digital twin based approach for optimal reticle management in semiconductor photolithography," *J. Syst. Sci. Syst. Eng.*, vol. 32, no. 3, pp. 320–351, Jun. 2023.
[47] N. Bratovanov, "Implementation of pick-and-place algorithms for the purposes of wafer handling robotics simulations," in *Proc. 3rd Int. Congr. Human-Comput. Interact., Optim. Robotic Appl. (HORA)*, Jun. 2021, pp. 1–5.
[48] A. F. Nicolescu, G. Enciu, A. Ivan, G. C. Avram, and D. A. Marinescu, "Wafer manipulating robots-design, programming and simulation," in *Sensors, Signals, Visualization, Imaging, Simulation and Materials*. Houston, TX, USA: WSEAS, 2009, pp. 139–144.
[49] S. A. Laghari, S. Manickam, and S. Karuppayah, "A review on SECS/GEM: A machine-to-machine (M2M) communication protocol for Industry 4.0," *Int. J. Electr. Electron. Eng. Telecommun.*, vol. 10, no. 2, pp. 105–114, 2021.
[50] S.-I. Imai, N. Sato, M. Kitabata, and S. Yasuda, "Fab-wide equipment monitoring and FDC system," in *Proc. IEEE Int. Symp. Semiconductor Manuf.*, Sep. 2006, pp. 114–117.
[51] P. Wegner, "Concepts and paradigms of object-oriented programming," *ACM SIGPLAN OOPS Messenger*, vol. 1, no. 1, pp. 7–87, Aug. 1990.
[52] W. Kozaczynski and G. Booch, "Component-based software engineering," *IEEE Softw.*, vol. 15, no. 5, p. 34, Sep. 1998.
[53] I. Crnkovic, "Component-based software engineering-new challenges in software development," *Softw. Focus*, vol. 2, no. 4, pp. 127–133, Dec. 2001.
[54] F. Wagner, R. Schmuki, T. Wagner, and P. Wolstenholme, "Introduction, definitions, and notation," in *Modeling Software With Finite State Machines: A Practical Approach*. Boca Raton, FL, USA: Auerbach, 2006, pp. 63–76.
[55] P. Linz, "Finite automata," in *An Introduction to Formal Languages and Automata*, 6th ed., Burlington, MA, USA: Jones & Bartlett Learning, 2016, pp. 81–132.
[56] D. H. Baek and C. H. Han, "Application of data mining for improving and predicting yield in wafer fabrication system," *J. Korea Intell. Inform. Syst. Soc.*, vol. 9, no. 1, pp. 157–177, Apr. 2003.
[57] B. Selman and C. P. Gomes, "Hill-climbing search," *Encyclopedia Cogn. Sci.*, vol. 81, p. 82, Jan. 2006.
[58] T.-H. Pan, D. S. Wong, and S.-S. Jang, "Chamber matching of semiconductor manufacturing process using statistical analysis," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 4, pp. 571–576, Jul. 2012.
[59] K. H. Baek, K. P. Han, G. Choi, H.-K. Kang, E. S. Jung, K. Song, C. Han, and T. F. Edgar, "Multiple input multiple output controller design to match chamber performance in plasma etching for semiconductor manufacturing," *J. Vac. Sci. Technol. B, Nanotechnol. Microelectron., Mater., Process., Meas., Phenomena*, vol. 31, no. 6, Nov. 2013, Art. no. 062201.
[60] D.-G. Ha, J.-M. Koo, D.-D. Park, and C.-H. Han, "APC technique and fault detection and classification system in semiconductor manufacturing process," *J. Inst. Control, Robot. Syst.*, vol. 21, no. 9, pp. 875–880, Sep. 2015.

JOONICK HWANG received the B.S. degree in electrical engineering from Soongsil University, South Korea. He is currently pursuing the M.S. degree with Sungkyunkwan University, South Korea. His professional background includes significant industry experience as an Equipment Engineer with Samsung Electronics Company Ltd. His current research interests include digital twins, cyber-physical systems, smart manufacturing, and data analytics.

SANG DO NOH received the Ph.D. degree in mechanical design and production engineering from Seoul National University, South Korea. He is currently a Professor with the Department of Industrial Engineering, Sungkyunkwan University, South Korea. His major research interests include CAD/CAM/PLM, modeling and simulation of manufacturing systems, smart manufacturing, smart factories, cyber-physical systems, and digital twins.