# DEVELOPING A DISCRETE-EVENT SIMULATION MODEL FOR SEMICONDUCTOR SUPPLY CHAIN

By:

**LIM SI YING**

(Matrix no.: 142665)

Supervisor:

**Assoc. Prof. Ir. Dr. Chin Jeng Feng**

July 2022

This dissertation is submitted to<br>
Universiti Sains Malaysia<br>
As partial fulfillment of the requirement to graduate with honors degree in

**BACHELOR OF ENGINEERING**

**(MANUFACTURING ENGINEERING WITH MANAGEMENT)**

School of Mechanical Engineering
Engineering Campus
Universiti Sains Malaysia

## DECLARATION

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed ……………………………………………….…………. (LIM SI YING)

Date ……………………………………….

STATEMENT 1

This thesis is the result of my own investigations, except where the otherwise stated. Other sources are acknowledged by giving explicit references.

Bibliography / references are appended.

Signed …………………………………………………………..… (LIM SI YING)

Date ……………………………………….

STATEMENT 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for interlibrary loan, and for the title and summary to be made available outside organizations.

Signed ……………………………………………………..……... (LIM SI YING)

Date ……………………………………….

# ACKNOWLEDGEMENT

I would like to express my gratitude towards everyone who has supported me throughout the final year project timeline. I am grateful for the aspiration, and the guidance, constructive feedback, as well as advice given to tackle the issues related to project. It is impossible to complete the thesis without the contribution and assistance given by the supervisor of my final year project, Associate Professor. Ir. Dr. Chin Jeng Feng. The comprehensive online resources in the form of article, journal, blog, webpage, and video tutorial from YouTube are provided as references throughout the time when I was making progress on the project. These guiding tools helped me to convert the concept into a real model throughout the iterative process. I have learnt a lots of knowledge from this project. I would also like to thank Mr. Yau, the person from semiconductor manufacturing industry, for clarifying my doubts and providing the real-world scenario in the semiconductor manufacturing industry.

Furthermore, I will take this opportunity to thank the course coordinator, Dr. Muhammad Fauzinizam Bin Razali for guiding us throughout the course. A lot of references were given to help the student in writing the thesis through the official e-learning portal. The requirements and important dates are also stated clearly in the messages. I appreciate that my faculty can provide a conducive learning environment for me as well as other students to gain the knowledge and skills before stepping into our career in professional field and contributing to the society.

Last but not least, I would like to thank my friends and family for the support. They gave me motivation and emotional support which are really matter to me. I am glad that they are always be there for me.

## TABLE OF CONTENT

# LIST OF FIGURES AND TABLES

# ABSTRAK

Disebabkan ketidaktentuan permintaan litar bersepadu (IC), adalah penting untuk mempunyai aktiviti perancangan induk bagi rantaian bekalan pembuatan untuk meramalkan permintaan. Keputusan pengeluaran dan perancangan pengeluaran adalah berdasarkan ramalan permintaan. Dengan hasil ramalan yang tepat, faedah boleh didapati dalam pengurangan kos inventori, penambahbaikan pemenuhan pesanan, tahap kepuasan pelanggan yang tinggi dan banyak lagi. Sebaliknya, kesan paling ketara yang disebabkan oleh ramalan permintaan yang buruk ialah pembaziran wang dan masa. Khusus untuk syarikat semikonduktor fabless, tempoh yang lebih singkat antara membuat pesanan jualan oleh pelanggan dan tarikh pesanan yang diminta berbanding dengan masa kitaran pembuatan yang diperlukan untuk proses adalah cabaran paling ketara yang dihadapi oleh industri. Pesanan jualan biasa ditempah pada lapan hingga dua belas minggu lebih awal daripada tarikh pesanan yang diminta. Manakala masa kitaran pembuatan untuk proses semikonduktor hujung ke hujung mengambil masa antara 20 hingga 30 minggu. Oleh itu, dengan membangunkan model simulasi peristiwa diskret, beberapa pembolehubah keputusan penting seperti kuantiti pengeluaran produk pada peringkat yang berbeza, kuantiti keluaran wafer kosong kepada fab wafer, jumlah inventori produk dan wafer kosong di akhir sesuatu tempoh boleh diperiksa dari segi perancangan dan kawalan. Model simulasi yang dibina dalam Python akan mempunyai parameter input seperti permintaan pelanggan untuk produk dan *GDPW (good die per wafer)*. Model ini diprogramkan dengan penggera stok untuk memberi amaran kepada pengguna apabila kuantiti produk mencapai tahap kritikal tertentu, oleh itu model ini dapat membantu syarikat mengawal proses tersebut. Sebaliknya, syarikat boleh menggunakan model tersebut untuk mensimulasikan aktiviti pembuatan apabila merancang. Melalui menjalankan simulasi, syarikat dapat mengetahui tempoh yang diperlukan untuk memenuhi permintaan pelanggan dan tahap inventori pada setiap langkah untuk mengelakkan kos penyimpanan inventori yang tinggi akibat lebihan stok.

# ABSTRACT

Due to the volatility of demand of integrated circuits (ICs), it is vital to have master planning activity for the manufacturing supply chain to forecast the demand. Production decisions and production planning are based on the demand forecast. With accurate forecasting result, the benefits could be found in the reduction of inventory cost, improvement of order fulfilment, high level of customer satisfaction and many more. In contrast, the most significant impact causes by bad demand forecast are waste of money and time. Specifically for fabless semiconductor company, the shorter duration between placing of sales order by customer and the order requested date compared to the manufacturing cycle time needed for the processes are the most significant challenge faced by the industry. The typical sales order is booked at eight to twelve weeks ahead of the order requested date. Whereas manufacturing cycle time for an end-to-end semiconductor process takes anywhere from 20 to 30 week. Therefore, by developing a discrete-event simulation model, some of the crucial decision variables such as the production quantities of products at different stages, the release quantity of bare wafer to the wafer fab, the amount of inventory of product and bare wafer at the end of a period could be examined in term of planning and control. The simulation model that is constructed in Python will have input parameters such as customers' demand for product and GDPW (good die per wafer). The model is programmed with stock alarm to alert the user when the quantity of products reaches certain critical level, thus this model could help the company to control the process. On the other hand, the company could utilise the model to simulate the manufacturing activities when comes to planning. Through running the simulation, the company could get to know the duration needed to fulfil the customer's demand and the level of inventory at each step to avoid high inventory holding costs due to overstocking.

# CHAPTER 1

# INTRODUCTION

## 1.1 General Introduction

The role of supply chain management in semiconductor industry becomes more important nowadays since the advent of fabless business model. The vertically integrated semiconductor manufacturing companies who used to own manufacturing facilities and develop own process technologies have shifted their focus into horizontal integration investment because the continuous advancement in process technology and assembly packaging requires heavy investment to upgrade manufacturing facilities.

Fabless semiconductor companies are the company that focuses on horizontal integration investment as they outsource manufacturing operations and majority of materials management to suppliers while focusing on master planning. However, the transformation raises the supply chain management issues as the front-end operations are often performed in highly industrialised nations, whereas back-end operations are carried out in countries where labor rates are cheaper. (Mönch et al., 2018)

Besides, there are center of competencies for wafer fab, probe, assembly, and test, which speed up innovations and strive for more economical solution, causing the increase in the complexity of supply chain management. (Ehm et al., 2011)

In this project, a discrete-event simulation model which is constructed by using open-source Python library SimPy will be used as the tool to examine the variables such as attributes, BOM, and flow set in planning and control of selective manufacturing process.

Before conducting the simulation and examine the simulation results, linear programming methodology is used to construct the selective manufacturing system with four-step manufacturing processes. This methodology helps to build a linear programming model based on four steps semiconductor manufacturing process namely wafer fab, wafer sort, assembly, and test. The input and structure required for demonstration of simulation model is obtained from the industry.

## 1.2 Problem Statement

Supply chain management (SCM) and demand forecast for the semiconductor industry is challenging due to its market volatility and long manufacturing lead time. Therefore, a simulation model for SCM in the semiconductor industry should be developed for master planning purpose, followed by examining the components in the model and testing it with variables. Besides, according to the academic literature, there is lack of supply chain design which is built and tested against real data obtained from industry. Hence, developing a simulation model for SCM in semiconductor industry from scratch by using discrete event simulation method could assist the SCM in semiconductor industry in a practical way as most of the existing study for supply chain model emphasise on improving the link between the supply chain design and market characteristic and redesigning the existing model due to misaligned supply chain.

## 1.3 Scope of Work

Nowadays, owing to the advancement in semiconductor technology and process, the number of attributes influencing the accuracy of semiconductor manufacturing processes increase tremendously, therefore the process flows are no longer simple four step stages but a complete selective manufacturing capability. However, the linear programming model used in this project is limited to the four steps selective manufacturing flow. Besides, the model used in this project has limited scope as it focuses on the selective manufacturing system instead of a complete supply chain.

# CHAPTER 2

# RESEARCH BACKGROUND

## 2.1 Literature Review of Simulation

Discrete Event Simulation (DES) concerns the modeling of a system as it evolves over time by a representation in which the state variables change instantaneously at separate points in time (A. M. Law, 2015). DES operates around entities such as people, animals, and information. These entities flow through a design of system stages created by a modeler, interacting with various resources under the created parameters, then collecting an output of tailored statistical information which is used to the decision maker (Luke Opacic & Taraneh Sowlati, 2017). This simulation method is good at providing a detailed analysis of systems involving linear processes and modeling discrete changes in system behavior. A classic use for discrete-event simulation is to evaluate the performance of a small number of proposed changes to systems that are too complex to model analytically. More recently, DES has been used to generate project and production schedules. This is accomplished by removing all uncertainty from the model. The model is then run from the initial state and executes jobs based on the defined dispatching/selection rules. Throughout execution, the model tracks the start and completion of jobs along with the resources being utilized to generate a schedule (Siedlak et al., 2018). However, there are no feedback loop in DES and the simulation result produced by DES may be misleading to a decision maker as in the absence of historical data, data from performance of similar system might be used (Sweetser, 1999). It is also not appropriate to be used when there is movement. Movement has to be represented as a time delay and a switch of position. During the time delay, the state of model will not accurately represent the system. A system that is sensitive to this is not suitable for discrete event simulation (Jundén, 2011). DES is applied within the agriculture sub-sector of livestock management which relates to crop production, resilience, and natural disasters (Gittins et al., 2020). DES is also used to analyse planning based on specific maintenance tasks by work package value, which is the impact of each task, to record the duration of maintenance activities during the aircraft line maintenance period accurately

(Albakkoush et al., 2021). Besides, DES is used in resource allocation methodology in production logistics system to solve the problem of unbalanced and unreasonable resource allocation (Li et al., 2020).

In System Dynamics (SD), system is defined as a collection of elements that continually interact over time to form a unified whole (Sweetser, 1999). Whereas dynamics refers to change over time (Sweetser, n.d.). System dynamics is a type of continuous simulation that is used for designing and improving policies or strategies in business, government, and the military (A. M. Law, 2015). There are key distinctions between DES and System Dynamics methodology (Sweetser, 1999). DES is often used to model particular processes but not entire system; SD is more focused on analysis of system. In addition, DES model is built from process map or flow chart; SD is built from casual loop diagram. SD is good at capturing the behavioral and qualitative relationships within the structure of a system. It is best suited to problems associated with continuous processes where feedback significantly affects the behavior of a system, producing dynamic changes in system behavior (Sweetser, 1999). The weakness of SD is that it cannot easily model the interarrival rates of discrete entities in a system (Sweetser, 1999).

Apart from that, Parallel Discrete-Event Simulation, also known as Distributed Simulation, is an alternative approach to execute a simulation model on a parallel computer (Fujimoto, 1990). Two or more different simulation models operating on separate computers were tied together over a network to produce one overall simulation model. This simulation methodology can reduce the overall time for simulation and execution time as well as allows increase in size of communications networks that can be simulated. One of the examples of adoption of this simulation methodology in real-time decision making is the air-traffic control system, which several hours of air traffic is simulated to decide for reroute traffic. Besides, when simulating a communication network with thousands of nodes or large military model, execution time might be excessive, thus parallel simulation might be considered (A. M. Law, 2015).

Monte Carlo Simulation is the other type of simulation methodology. It is defined as a scheme employing random numbers, that is random variates, which is used for solving stochastic or deterministic problems, therefore the stochastic discrete-event simulation is also included in this definition (A. M. Law, 2015). Monte Carlo method is a continuous simulation methodology which is the opposite of discrete-event simulation model (Dirk P. Kroese, 2014). Monte Carlo technique is easy and efficient. Its algorithm is parallelizable; hence several models can be run on different computers, and computation time is reduced (Dirk P. Kroese, 2014). The inherent randomness of Monte Carlo also enables better exploration of the search space (Dirk P. Kroese, 2014). This simulation methodology is used to modelling real-life system such as a complex road network, the transport of neutrons, and the evolution of the stock market (Dirk P. Kroese, 2014).

Spreadsheet simulation refers to the use of a spreadsheets as a platform for representing simulation models and performing the simulation experiment (Seila, 2005). Both discrete-event simulation and Monte Carlo simulation methodologies can be done in spreadsheets such as Excel if the problem of interest is not too complex (A. M. Law, 2015). Spreadsheet simulation is able to generate random values from some basic probability distributions such as normal, uniform, binomial and poisson distribution, the summary statistic such as mean and variance, as well as graphical plots like histogram. The limitations of spreadsheet simulation model result in only the simple data structures are available and complex algorithm are difficult to implement. Besides, spreadsheet simulation may have longer execution times than simulations built in a discrete-event simulation package and only limited data storage is available (Seila, 2005). This simulation is used for performing risk analyses in application areas such as finance, manufacturing, project management, oil, and gas discovery (A. M. Law, 2015).

## 2.2 Simulation Methodology

When developing a simulation methodology, the building process can be first divided into four phases as stated by (Banks, John S., Barry L. Nelson, & David M. Nicol, 2010). Problem formulation will come as the first step at the beginning of the phase, followed by second steps, setting of objectives and overall design. Second phase is related to model building and data collection; therefore, the third step is model conceptualization, and fourth step is data collection, followed by fifth step, model translation, verification, and the step seven, which is validation. Then, the third phase of model building process concerns the running of the model. It involves steps 8, experimental design, step 9 which is production run and analysis, as well as step 10, more run. This phase focuses on the plan to experiment the simulation model thoroughly.

According to (Robert G. Sargent, 2010), decision makers obtain information from the results of simulation models to solve the problems. Therefore, individuals affected by decisions based on the simulation models are concerned with the accuracy of the results. (Law and Kelton, 2000) stated that simulations can be subject to start-up variations. Hence, a system is expected to reach a consistent state, or also known as “steady-state” in order to operate under normal conditions. The model should be run for a warm-up period to achieve steady state before collecting the statistics from the system. (Winston, 2004)

An important consideration in developing simulation methodologies specifically for the supply chain management in fabless semiconductor company is to have large set of big data which can be analysed using the model developed to determine the accuracy. Big data includes information related to internal product like test time, yield, binning, and flow as well as electronic-data-interchange of customer forecast and orders, and WIP from suppliers.

Since the discrete-event simulation is a statistical experiment, the output variables may contain random error. Therefore, a proper statistical analysis should be carried out by running the simulation more than one time. According to (Whitt, 1989), when dealing with unknown system, like the dynamic planning matrix for an end-to-end manufacturing process that taken place in fabless semiconductor companies, it is best to run the simulation for several iterations and perform statistical analysis to obtain the confidence intervals for the statistics in question. Finally, the fourth phase, implementation involves step 11, documentation, and reporting, as well as step 12, implementation. The most crucial step is step 7, validation because an invalid model will lead to erroneous result and wrong decision.

## 2.3 SimPy: The Simulation Library

SimPy is a computing modelling and simulation software (M&S) module written in pure Python (Dmitry Zinoviev, 2018). It is a simulation framework for a process-oriented discrete event simulation and most probably the oldest and best-known DES package for Python (Lang et al., 2021). As an open-source software, SimPy is released under the Massachusetts Institute of Technology license, which not only allows active communication of users through the dedicated mailing list but is also permissive as it gives flexibility to license the work. The documentation also provides understanding of SimPy concepts and example of simple simulation models, which is adequate as mentioned by several authors (Dagkakis et al., 2016) (Dagkakis & Heavey, 2016). SimPy was introduced for more than a decade, SimPy3 version was released and introduced many changes in syntax, hence rendering models written in SimPy2 not compatible with the new release (Dagkakis et al., 2016). SimPy3 introduced a completely new and easier-to-use API, still relying on Python's generators as they proved to work very well. SimPy is described as a package providing a simulation clock, an event and process system, and capacity-constrained resources (Jefferies & Göhlich, 2020).

SimPy toolbox acts as a modelling tool of asynchronous tasks by using primitive operations such as events, resources, and processes. These primitives will be used for modelling the distinct components that encompass the forest-based supply chain, such as machines, clients, and vehicle routing. The implementation of SimPy as the simulation tool for modelling the forest-based supply chain can be seen in the article (Pinho et al., 2021). SimPy library was used to imitate the behaviour of supply chain object which includes chipper, truck, wood pile, client, and depot. These supply chain objects are specifically created for the Discrete Event Simulation of forest fuels supply chain and each of the object has its own class data type. (Pinho et al., 2021) Besides, the SimPy was used as programming environment for capacity planning purposes (Hristoski & Mitrevski, 2013).

SimPy tool has no graphical user interface (GUI), hence it is more directed to the developers. The behaviour of active components like vehicles, customers, and messages, is modeled with processes, that is why SimPy will be categorized as one of the process-oriented DES programming paradigms. All the processes live in environment; hence they interact with environment and each other via events. Processes are described by simple Python generator. When a process yields event, the process suspended. SimPy resumes the same process when the event is triggered, and multiple processes can wait for the same event (Dagkakis et al., 2016).

This process-based DES programming language is successfully used to model different e-commerce paradigm which consists of three processes, named as source, customer, and request (Hristoski & Mitrevski, 2013). Furthermore, in a queuing system, the application of process-based DES paradigm would have two threads, one for simulating queuing system and the other one for simulating operation of the server.

## 2.4 General Description of Semiconductor Supply Chain

The volatile demand cycle in semiconductor industry is the main challenges for the companies that owned manufacturing facilities and developed own process technologies for manufacturing of their integrated circuits. Therefore, companies begin focusing on horizontal integration investment to increase the number of goods and product portfolio, at the same time, emphasizing on the master planning. Figure 2.1 shows the planning matrix of semiconductor supply chain (Meyr et al., 2015). Figure 2.1 indicates some operations that require the companies to make manufacturing strategic decision. The operations, which include capacity planning, production planning, materials requirement planning, inventory control and scheduling, are utilised for effective resource planning.

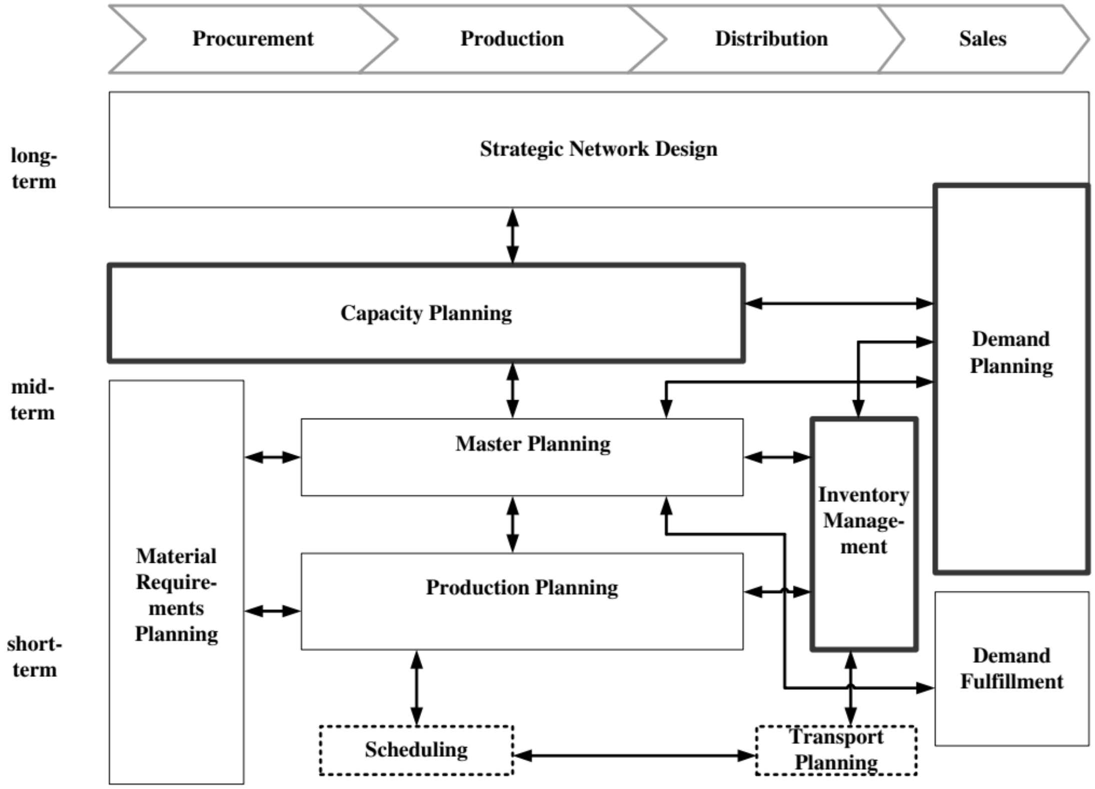

Figure 2.1: Supply chain planning matrix for semiconductor supply chain

Demand forecast is the fundamental of production decisions and production planning (Institute of Electrical and Electronics Engineers., 2008). Accurate demand forecast may reduce cost of inventory holdings, improve order fulfilment, and promote customer satisfaction. Precise demand forecast is important to avoid either an underestimated demand forecast or overestimated demand forecasting value.

On the other hand, Master Planning, or Master Production Scheduling (Mönch et al., 2018),  determines the appropriate inventory required for specific products considering external and internal demand to meet order fulfilment in a specific timeframe with production capacity in consideration. In master planning system itself, the structure of products to be produced like BOM information, routings and process parameters like lead time, yields are predefined or setup. Only then the Master Planning engine can work to formulate the targeted outputs for fulfillment. As shown in figure 1, the Master

Planning function takes input from Demand Planning and Inventory Management functions to formulate an output in Demand Fulfilment.

The automotive sector in 2021 faced loss of $110 billion in sales as factories are sitting idling waiting for semiconductor component (Salter, 2021). Hence, LT is critical attribute in determining inventory availability during selective manufacturing computation. Product yield is another key attribute defining inventory availability due to its uncertainty in fabrication and manufacturing processes.

# CHAPTER 3

# RESEARCH METHODOLOGY

## 3.1 Introduction

Considering there is finite resources and time in manufacturing, optimization is the best option in resolving the complex semiconductor supply chain to find the most productive way in selective manufacturing. Tens of thousands of variants with different demand must be manufactured in defined time and resources. This research attempts to use linear programming methodology to find a standard algorithm applicable for selective manufacturing based on the four steps supply chain process namely wafer fab, wafer sort, assembly, and test. Let’s use selective manufacturing system shown in Figure 1 to outline the linear programming model (Missbauer & Haeussler, 2021).

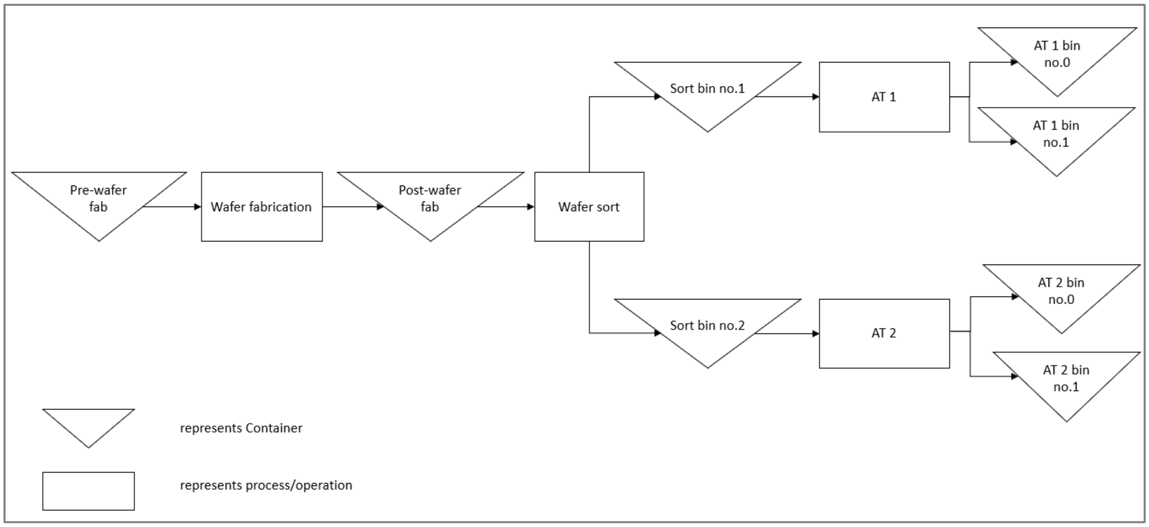

Figure 3.1: Selective Manufacturing System Model.

The definition of model entities which represent the material flow in selective manufacturing system:

- **Wafer**: A thin slice of semiconductor used for the fabrication of IC.
- **Die:** A small block of semiconductor material that has been cut out of the wafer.

- **Pre-wafer fab**: Input buffer of *Wafer fabrication* entity.
- **Wafer fabrication**: Process to produce complete electrical or photonic circuits on semiconductor wafers.
- **Post-wafer fab**: Output buffer of *Wafer fabrication* entity and input buffer of *Wafer sort* entity.
- **Wafer sort**: Also referred to as wafer probe or wafer test. It is an electrical testing process performed on silicon die while the die is in wafer form.
- **Sort bins**: Output buffers of *Wafer sort* entity. They are also the input buffer of *AT 1* and *AT 2* entities respectively.
- **AT 1 and 2**: AT stands for assembly/test, which belongs to the backend operations in semiconductor manufacturing. These processes involve packaging of die in an encapsulated supporting case to prevent physical damage or corrosion, followed by testing process to determine if device work as specified design in package form.
- **AT 1 and 2 bins**: Output buffers of *AT 1* and *2* entities. The completed functioning end device will be shipped out from these output buffers.

The modelling of assembly/test processes are combined in this simulation model and is named as *AT 1 and 2* as the steps consist of two assembly/test machines. This is because most of the projects that created a simulation model for supply chain in semiconductor industry did not discuss fine-grained models. To have a more realistic simulation result, yields of wafer sort and assembly/test processes are not 100% due to the reject rate. The model is built according to the flowchart as shown in Figure 1. Firstly, the customer's demands for products trigger the wafer fabrication process and the bare wafers in the *pre_waferfab* bin were retrieved by the process to produce batches of wafers. The wafers are put into the *post_waferfab* bin when the lead time for wafer fabrication has passed. Next, the wafers are retrieved by wafer sorting process to generate numerous dies by consuming lead time for wafer sorting. The dies are binned into two different categories in which the flawless dies are separated from the dies with lower performance and functionality. The two bins are denoted as *sort_bin no.1* and *sort_bin no.2*. The rejected dies are removed based on reject rate before being put into these bins. Subsequently, the sorted dies in each bin are redrawn into the assembly/test machines, *AT 1* and *AT 2* respectively. After the lead time of assembly and test has passed, the dies from each of the bin are placed into two different categories of *AT bins* to separate the finished goods by the performance and functionality. The rejected finished goods are removed based on reject rate and the final products could be found in four different types in different bins, denoted as AT 1 bin no.0, AT 1 bin no.1, AT 2 bin no.0, AT 2 bin no.1. To sum up, the model produces four different finished goods from one wafer type. The name of entities used in the program and its respective attributes are shown in Table 3.1.

Table 3.1: Description of model entities related to four steps manufacturing process by activity generator functions, including name, type, function, and all related parameters

| Activity Generator Functions | | | |
|---|---|---|---|
| Entity | Type | Function | Related Parameters (attributes) |
| Waferfab_machine_gen | | | |
| waferfab_machine | Process | Getting WIP from pre_waferfab bin then putting wafer into post_wafer bin | waferfab_batchsize: The released quantity of bare wafer to the wafer fabrication process<br>waferfab_lead_time: The mean lead time of wafer fabrication<br>waferfab_lead_time_std: The standard deviation of lead time of wafer fabrication |
| pre_waferafab | Container | Models inventory on-hand at pre_waferfab container | pre_waferfab_capacity: The size of the pre_waferfab container<br>pre_waferfab_ini_capacity: The initial amount of bare wafers |
| post_waferfab | Container | Models inventory on-hand at post_waferfab | post_waferfab_capacity: The size of the post_waferfab container<br>post_waferfab_ini_capacity: The |

|  |  |  |  |
|---|---|---|---|
|  |  | container | initial amount of wafers |
| sort_machine_gen |  |  |  |
| sort_machine | Process | Getting WIP from post_waferfab bin then putting dies into sort bins based on percentage of bin. | sort_lead_time: The mean lead time of wafer sort<br>sort_lead_time_std: The standard deviation of lead time of wafer sort<br>sort_wafer_batchsize: The number of dies in one batch when being retrieved from waferfab_bin.<br>sort_reject_rate: The mean reject rate of wafer sort.<br>sort_reject_rate_std: The standard deviation of reject rate of wafer sort.<br>num_die_per_wafer: Number of good dies that can be assembled per wafer. |
| sort_bin | Container | Store dies from wafer sorting | num_sort_bin: The number of sort bin<br>sort_bin_capacity: The size of sort bin<br>sort_bin_ini_capacity: The initial amount of die in sort bin.<br>sort_bin_percent: The percentage of allocated dies |

| | | | |
|---|---|---|---|
| AT_machine_gen | | | |
| AT_machine process | Process | Getting WIP from sort bins then putting wafer into post_wafer bin | AT_lead_time: The mean lead time of assembly/test<br>AT_lead_time_std: The standard deviation of lead time of assembly/test<br>AT_die_batchsize: The number of dies in one batch when being retrieved from sort bin.<br>AT_reject_rate: The mean reject rate of assembly/test.<br>AT_reject_rate_std: The standard deviation of reject rate of assembly/test<br>num_AT_machine: The number of assembly/test machine. |
| AT_bin | Container | Store dies from assembly and test (AT) | num_AT_bin: The number of assembly/test bin.<br>AT_bin_capacity: The size of AT_bin_container<br>AT_bin_ini_capacity: The initial amount of die in AT bin |
| ship_out_gen | | | |
| ship_out | Process | Getting finished goods from AT bins then ship out | ship_out_lead_time: The duration to arrange for shipping of finished goods. |

|  |  |  |  |
|---|---|---|---|
|  |  | according to customer demand value |  |
| orderreceive | Store | Storing order from customer and supporting requests for the Store resource. |  |
| orderfulfil | Store | Storing completed customer’s order and supporting request to ship out the order. |  |
| wafer_control | Process | Track the stock level of wafer | lead_time_wafer_supply: The duration to receive wafer from supplier after calling. |

Table 3.1 shows the entities that build up the simulation model whereas Table 3.2 indicates the decision variables which act as the outputs of running the simulation. The variables define the flows and WIP level in system over time. The material balance equations define the network structure. The common structure is found in Equation 3.1. The simulation was run for 1000 hours and the results could be found in the next chapter.

Table 3.2: Decision variables

| Decision Variables | Description |
|---|---|
| $P^{S}_{(fx)t}$ | Production quantity of product, $f_x$ *(x = 1,2,3,4)* in period *t* in the wafer sort. |
| $P^{AT}_{(fx)t}$ | Production quantity of product $f_x$ *(x = 1,2,3,4)* in period *t* in the assembly and test facility. |
| $P^{W}_{kt}$ | Production quantity of wafer type *k* in period *t* in the wafer fab. |
| $R^{W}_{kt}$ | Release quantity of wafer type k in the period t to the wafer fab. |