International Journal of Control and Automation
Vol.7, No.2 (2014), pp.75-84
http://dx.doi.org/10.14257/ijca.2014.7.2.08

A Generic Simulation Framework for Efficient Simulation Analyses for Semiconductor Manufacturing: A Case Study

Dae-Eun Lim[1] and Minseok Seo[2*]

[1] *Division of Business and Commerce, Baekseok University, South Korea*
[2] *Semiconductor Business, Samsung Electronics Co., South Korea*

del@bu.ac.kr, smskorea@gmail.com

Abstract

A generic simulator and data-driven generic simulation are widely used to reduce modeling efforts. The concept of a generic simulator includes the extracting and storage of the common parts of a domain. The common parts in semiconductor manufacturing simulation can be the scheduling logic and the operation rules of automated material handling systems, such as auto-guided vehicles or overhead hoist transfers. The execution of a simulation instance with pre-programmed common parts is triggered by providing the required data. Simulation analysts do not need to create separate simulation models unless special logic or operation rules not included in the common parts are considered.

Emerging requirements for a generic simulator can be summarized as follows. For the best decisions, it is necessary to study every aspect of alternatives thoroughly. However, the volume of data increases significantly as the number of alternatives and variables associated with the data increases. Because previously introduced generic simulators have difficulties in handling large volumes of data, simulation analyses are often limited and not satisfactory. In addition, the statistical functions of commercial simulation packages used in semiconductor line modeling leave something to be desired. That is, they are not easy to use and not particularly powerful.

This paper demonstrates an improved generic simulation framework with enhanced features. It is capable of handling large volumes of data, utilizing abundant statistical functions, and reducing the data preparation time. With the proposed framework, simulation analysts can reduce the simulation modeling time while exploring many alternatives with a powerful statistical analyzer and a rapid visualization tool. These features are expected to contribute to more efficient decision making. This research is a case study of a leading electronics company in South Korea.

Keywords: *Semiconductor Manufacturing, Simulation Analysis, Generic Simulator, Data driven simulator, Automod, R, SQLite*

1. Introduction

The semiconductor industry is known as both a capital- and technology-intensive industry. Semiconductor companies, after overcoming the high entry barriers, make efforts to survive given the fierce global competition. One way they do this is improve their manufacturing system more efficient. Improving their manufacturing systems involves optimizing the scheduling and the real-time dispatching logic, and introducing automated material handling systems such as OHT (Overhead Hoist Transport) and AGV (Automatic Guided Vehicle).

* Corresponding author.

ISSN: 2005-4297 IJCA

In addition, wholly new manufacturing system can be devised. Recently, Song et al. [11] introduced the high-performance 'in-line' type system. Another recent concept of manufacturing is introduced and summarized by Macia-Perez *et al.*, [4].

However, the effort should not be halted at a certain level because internal and external conditions such as the continual developments of new products, changes in customer preferences, and changes in management strategies changing quickly and continuously**.** Improvement schemes for manufacturing systems should go through the steps of evaluation and identification of profits and potential problems before the actual introduction of the system. During this process, one of the mostly used tools for evaluations is a simulation. For various applications of simulation refer to Dachyar [3], Rashid et al. [9], Taktak *et al*., [10]. Simulations are known to be a very effective for evaluating various recommendations for manufacturing systems even if the systems do not yet exist, which becomes very helpful for reducing the level of investment risk. Although simulations have clear advantages for better decisions, one notable disadvantage is the lengthy modeling time. The more the details of manufacturing lines are encapsulated, the more modeling time is required. Creating only one ordinary simulation model of a semiconductor fabrication plant (henceforth fab) comprises about the 50% of project duration.

AutoMod [1] has been widely used in simulating fabs because it copies various physical movements of automated material handling systems (AHMSs) very well. This feature is advantageous in the sense that modern fabs are fully automated with various AMHSs. However, the low code reusability level and the required tedious coding are disadvantages when using AutoMod. For example, when converting a fab floor plan from AutoCAD [2] file to an AutoMod simulation model, the work is equal to that required when redrawing the plan. As a matter of course, much time and tedious work are required for this non-core activity. Among the simulation phases described in Figure 1, the most time-consuming phases are planning and model development (modeling). The time duration for the planning phase cannot easily be reduced, while the modeling phase, which involves many simple repetitive jobs, is relatively easy. For better decisions, various simulation analyses from diverse viewpoints should be considered. Therefore, reducing the modeling time can be very valuable.

One of the many ways to reduce the modeling time is to use a generic simulator. A generic simulator is considered as a synonym of a data-driven simulator in this research. Some research papers [6, 8] distinguish between the two concepts above but they share the same goal, i.e., reducing the simulation building time. In addition, Wy *et al*., [13] stated that "both approaches can be considered to be similar because every simulation program generator should have a data-driven generic simulation model within its code generator." Kim *et al*., [5] claimed that "a data-driven simulator which is generic to a target domain can simulate different instances of systems in the domain." A generic simulator has the common parts of a domain. For example, in the semiconductor industry, the operation of an AMHS or scheduling or dispatching logic can be common parts that are pre-programmed. Simulation analysts provide data differently from instance to instance. Examples are factory layouts, product mixes, and the number of AGVs. The provided data enables the program immediately, and there is no need for the analysts to do any formal programming [8]. By pre-programming common and complex parts, the simulation modeling time can be significantly reduced. The generic simulator is not the latest concept, but it has been successfully applied to various areas, such as the automobile and semiconductor manufacturing industries [5, 12, 13]. For previous research on generic simulators, the reader can refer to Kim et al. [5] and Wy *et al*., [13].

With generic simulators, a simulation analyst can reduce the model building time, and create models that are more reliable, as the level of code reusability is high [7].

This research proposes an improved framework for generic simulators with enhanced statistical analysis functions and the ability to handle large volumes of data. It is not easy to conduct various statistical analyses using AutoMod without using commercial packages as well. When it becomes necessary to investigate various alternatives, many simulation models are built and run based on the design of experiments. If only variations in the parameters are concerned, AutoMod can deal with generating various models with different parameters using its built-in function known as the snap function. However, when variations in the facility layouts are concerned, AutoMod is not an efficient solution because factory plans in simulation models cannot be automatically generated. If simulation models with variations in the layouts can be generated and executed automatically, the modeling time can be reduced significantly. Furthermore, the scope or study period of the simulation is often limited due to the deficient capability of the software for handling a large volume of data. Microsoft's Excel is commonly used for analyzing output files. As a result, only simple performance measures such as the mean or variance are used. It is difficult to run an in-depth analysis of thousands instances of the utilization of production and transfer equipment over time, for example.

While this research is a case study that was developed in a department of Samsung electronics, actual features such as screen captures are not provided due to Samsung's security policy. This paper is organized as follows. Section 2 introduces several real problems on semiconductor manufacturing lines and emerging requirements to solve these problems by means of a simulation. Also, pertinent previous studies are introduced. In Section 3, every components of the generic simulator framework proposed here is described. Finally, Section 4 summarizes and concludes this research.

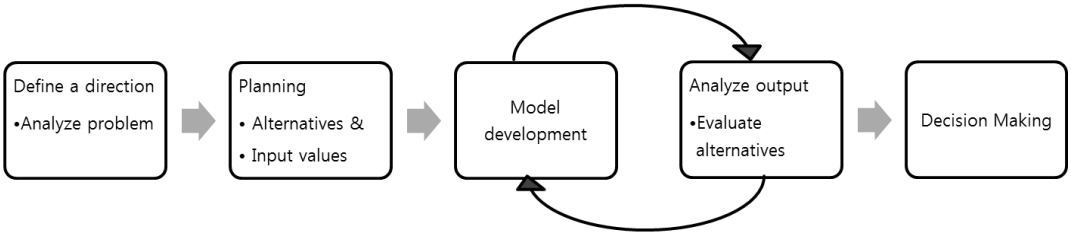

Figure 1. Simplified phases of simulation analysis

2. Emerging Requirements for a Generic Simulator

A wafer is a processing unit in a fab that undergoes hundreds of processing steps. After being fully fabricated, a wafer is electrically tested and undergoes a final process called 'test & package' before its shipment. The broadly classified three processes of fab, electrical testing and test & package have their own issues that prevent efficient manufacturing but they frequently have the following issues in common.

- How can material flows in manufacturing lines be made more efficient when improving facilities layouts?
- What is the optimal batch size between processes?
- What is an optimal number of automated material handling units, such as AGV or OHT? Also, the average utilization or utilization over time is curious.

- When changing factory layouts, the batch sizes between processes or the production plans of products, quick and accurate estimations of factory lead time are necessary.

To cope with these issues, many alternatives (simulation models) must be thoroughly analyzed. To do this, following features are required for a simulator.

- The time and effort for the modeling, execution and analyzing of many simulation instances should be minimized.
- A large volume of data should be handled easily.
- Various statistical and visualization functions must be easy to use.

Previous studies are not sufficient to satisfy these requirements. Recent work such as that by Kim [5], Wy [13] and Wang [12] focused on developing suitable generic simulators for various domains. No works considered how to deal with many simulation instances and a large volume of data. Simulation analysts at Samsung electronics, including the last author here, understood the need for new modules to handle these requirements. The newly developed modules are included in a generic simulator at Samsung electronics. This research proposes a generic simulation framework that can meets the needs of analysts and improves the productivity of simulation analyses. The newly included features are listed below.

- To reduce the necessary modeling effort, the execution effort and the subsequent analysis time, the following functions are implemented: converting facilities layout in the AutoCAD file format into the AutoMod data format almost automatically, examining the integrity of all relevant data, and executing many simulation instances automatically.
- To handle a large volume of data, the database SQLite is suggested and the manner of making a connection is demonstrated.
- To support various statistical analyses and visualizations, a connection with the statistical software R is made.

A detailed description to the proposed simulator with the newly developed modules is given in the next section.

3. Components of the Proposed Generic Simulation Framework

This section introduces the functions of every component in the framework and shows how they work. The framework has four main components: the simulation engine, the main program, the database and its connector, and the statistical analyzer. The main program has five sub-components.

3.1. Simulation Engine

AutoMod is completely in charge of executing the simulation models. Usually, after gathering all of the required data, the simulation models are coded and then executed. The main role of our framework is to generate simulation instances in the form of AutoMod codes that can be executed by AutoMod without modification. AutoMod is used as the engine in this framework because it takes a considerable amount of time to develop an entirely new simulator. Moreover, maintenance is difficult.

3.2. Main Program

The main program plays the following essential roles:

- Gathers input data, it is the factory layouts, product mixes, and the output of each product. Checks the integrity of all the gathered data.
- Generates simulation models in the form of the AutoMod code if no error is found during checking integrity.
- Executes all the simulation models automatically.
- Transfers simulation output from the database to the statistical analyzer, and shows statistical analysis tables and graphs generated by R.

3.2.1. Layout editor

AutoCAD is mostly used to make the layout drawing of a factory plan. Although, AutoMod provides a function that can deal with AutoCAD files, it is inconvenient and hard to process big files. To create simulation instances, a factory layout in the AutoCAD format is often drawn again. In addition, drawing a layout with AutoMod takes a long time but an inaccurate layout can be produced. It would be ideal if a layout file in the AutoCAD format could be automatically converted into the AutoMod format. DXF file format is devised for exchanging data between AutoCAD and other programs refer to manual [2] for further consideration. One of the main functions of this layout editor is reading a factory layout file in the DXF file format, and displaying it. Also, editing a layout is possible so that equipment for the production or AMHS, such as conveyors, diverters, lifters, can be added or erased. The edited input data is converted into a simulation model (equivalently, AutoMod code) after other data is exhaustively collected.

The most important function of the layout editor is transforming figures contained in a DXF file into objects that are defined in AutoMod. There are many lines, circles and arcs in a DXF file and they represent equipment for production, material handling, etc. However, after reading this file, the types of figures cannot be automatically identified. They can be equipment, pillars or even walls. Thus, additional manual works must be done to characterize what the figures stand for and to designate the identity of figures. Converting simple figures into equipment one by one is tedious and cannot be fully automated. Here, distinctively unique feature of our framework is introduced. To reduce the converting time, the metadata of AutoCAD such as block and layer is fully used. Previously introduced generic simulators can also process DXF files, but the metadata has not been utilized before. The metadata usages are given below:

- Usage of layer information: To make a factory layout plan many, departments are in cooperation. Contents of a file are diverse and look complicated. However, contents are distinguished by layers. It means each department makes their own layer and these layers are combined in a file. Rail paths of OHTs or AGVs are contained in different layers. In this proposed framework, for example, by indicating a layer as a rail layer, arcs and lines in that layer are simply converted to ‘pathmovers’ of AutoMod.
- Usage of block information: blocks are made to draw the same figure (equipment) repetitively, or to insert a certain layout drawing into the target layout. For example, the number of Sawing machine is hundreds in the test & packaging process. With a Sawing machine block, hundreds of equipment can be drawn in a very short time. In

this generic simulation framework, by indicating a block as a machine of certain process all blocks are converted into the same ‘process’ or ‘queue’ of AutoMod.

3.2.2. Input data editor and integrity checker

Among the simulation analysis phases, one of the most time-consuming jobs is collecting and checking the integrity of data. The integrity of data refers, for example, if a machine named ‘AA’ is included in a processing step file, this machine ‘AA’ must exist in a factory layout file. Simulation models cannot be run properly or output can mislead if data integrity is not guaranteed. Input data, e.g. layouts, production plan or processing steps are saved in separate files and their formats are also various. Accordingly, checking manually the integrity of all the input data is another time-consuming work and there is a strong call for automating checking integrity. Checking integrity is also needed inside a file itself since duplicated machine names or omitted fields can be included.

The basic task of the input data editor is reading each input data file, displaying them, checking integrity, and suggesting rectifying errors. In the proposed generic simulation framework, data integrity is forced to be checked before every creation of a simulation model.

3.2.3. AutoMod model creator

To create an AutoMod model, basically two files should be generated. Other files are created by AutoMod after compiling is over. But files about ‘systems’ used in AutoMod such as pathmover system saved in a separate file. Two basic files are saved in the same directory that named ‘[ModelName].arc’. Their file extensions are ‘amo’ and ‘asy’ (henceforth ‘amo’ file and ‘asy’ file, respectively).

The ‘amo’ file contains the configurations for modeling. For example, the unit of length and time is defined to be ‘feet’ and ‘seconds’, respectively. Also, to control the length of simulation a snap function is defined refer to a line starts with ‘CONTROL’. ‘DRAWPOS’ is about the position of objects. Finally, this simulation model has two systems. One is movement system and the other is process system. They are enrolled at the last line.

The ‘asy’ file is rather lengthy, and contains information about load type, the shape and capacity of queues, the operation rules of resources, initialization function, the logic of processes, etc.

It was difficult to figure out all lines of these files. However, simulation models in the same domain have many similar codes and default setting. Thus, after observing many models, AutoMod analyst can figure out the meaning of each line.

```
VERSION 9.1
BUILD 1412.29
UNITS       Feet Seconds
DRAWPOS     minx -150 maxx 108.272899985313
            miny -52.3181999921799 maxy 53.1363999843597
            minz 0 maxz 4
            try 0 trx 0
CONTROL     snaplen 10 Days counts 1 autorep
debugger on
MOVESYS     name pm
 PROCSYS    name proc
```

Figure 2. An example of ‘amo’ file

3.2.4. Batch executor

To review a number of alternatives, the same number of simulation model is needed. Once a run of a simulation model is done, another instance should be started. The time duration, from the instant when the model ends to the instant another model is started is also time-wasting and running simulation models one by one is a cumbersome task. Since delayed time duration between the end and the start of instances contribute to delay decision making, this intermission must be eliminated by automating the start of simulation instances

Furthermore, a series of tasks to execute a simulation model in AutoMod is difficult to be automated. The tasks involves turning AutoMod on, reading a model file, and running a model by clicking 'Run Model' button in menu bar.

However, with the proposed framework, AutoMod program don't need to be manually turned on. On the completion of a simulation experiment, the main program of the framework sends a command of starting a new model to command prompt. Then, the next model is executed automatically. To be more specific, it is possible that compiling and executing the AutoMod simulation models in console mode. By sending a command to command prompt, execution of simulation models are possible even when AutoMod is currently not running. In other words, simulation instances can be executed with the main program even AutoMod is not turned on. Its format is:

c:>amod [-options] [AutoMod model name]

Table 1. Options used in this research and its description

Option	Description
i	Starts the simulation with graphics displayed. The simulation runs until the end of the user defined run control and then closes.
n	Starts the simulation without graphics. The simulation closes at the end of the user-defined run control.

Table 1 represents options used in this research and the other options can be found in user's guide [1]. For example, a command 'amod -n TestModel' means as follows: run a model named 'TestModel' and it will terminate automatically when its termination criterion is satisfied. In addition, graphics are not displayed during a run. When a run of the model is done, AutoMod also terminates automatically.

The parameters can be changed during simulation runs with the snap function of AutoMod. However, this function is about numbers, so changes in the factory layouts are cannot be dealt with. There can be a number of simulation models depending on layout alternatives. The batch executor can reduce manual efforts for executing those many alternatives one by one.

3.2.5. Output Viewer

This module displays simulation output and implemented using Adobe FLEX. Also, graphic (output) files of R are good enough to be displayed. Simulation output can be checked in AutoMod and processed using Excel. However, it is hard to deal with large data in a short time period. For example, to determine the capacity of equipment for

storing WIP (work in process) such as stokers, STBs (side track buffer) in fabs simulation is used. The capacity is determined usually based on an average utilization since calculation is simple but utilization fluctuates over time. For similar reasons, time based utilization graphs of equipment for production are also needed to be observed. Using simple average values may mislead. A number of all types of equipment may up to thousands so that storing and processing time based utilization using Excel won't be easy. Thus, an additional module, connected to a database, for displaying data is suggested.

The main task of this module is based on analysts' request, displaying query results from a database. This module is devised to handling large data and

4. SQLite Database and DB Connector

Suppose while simulating a number of alternatives (simulation models), the utilizations of equipment is stored every predefined time interval. Then file size can be so large that files can be difficult to handle. The database is needed to save and process a large volume of data easily.

This framework recommends adopting a SQLite database, free and open software. The SQLite is light-weighted and it can be embedded our framework very easily. Thus the SQLite is used in this framework. In addition, installation and setting is not that difficult. Because simulation analysts travel all over the world, they thought the lighter the better. Finally, SQL can be used without extra study.

AutoMod basically provides a database function open and navigate the data sources, execute database commands, retrieve data. Using these functions, event logs are stored in a database during simulation experiments. The main program delivers output to output viewer or statistical analyzer. By coupling the SQLite database, in-depth and rapid analyses of thousands instances are expected.

5. Statistical Analyzer

The role of statistical study in simulations is considerable. Before conducting simulation analyses, simulation instances are designed based upon the design of experiments. After obtaining results from simulation experiments, descriptive statistics are computed, and some indexes are statistically tested. AutoMod is fairly good at doing complex simulation experiments but is not that good at statistical analyses and visualizations. For example, suppose that analysts want to find the effect of batch size between processes on the lead time. Regression analysis can be used but that function is not included in AutoMod and Excel is less suitable for handling large data files. Additional software about statistical analysis is needed and R is adopted in this framework.

The framework interfaces with R using JRI (JAVA/R interface) [14]. Typical statistical analysis can be done only in the framework. Suppose that a simulation analyst request regression analysis and Figure 3 is an illustrative example of regression analysis using JRI. Many powerful function of R can be used in the framework after developing codes for interpreting results.


```
import org.rosuda.JRI.REXP;
import org.rosuda.JRI.Rengine;

public class JRItest
{
    public static void main(String[] args)
    {
        Rengine rengine = new Rengine(null, false, null);
        REXP coeffi, adjRSquared;

        rengine.eval("setwd(\"D:/WorkSpace_JAVA/JRI/src/\")");
        rengine.eval("example = read.csv(\"example.csv\");");
        rengine.eval("result = lm (A ~ B, data = example)");
        /*Store coefficients*/
        coeffi = rengine.eval("result$co");
        /*Store Adjusted R Square*/
        adjRSquared = rengine.eval("summary(result)$adj.r.squared");
        /*Print coefficients*/
        System.out.println("Intercept: "+coeffi.asDoubleArray()[0]+"\n"
                           +"Slope: " +coeffi.asDoubleArray()[1]);
        /*Print Adjusted R Square*/
        System.out.println("Adjusted R Square: "+adjRSquared.asDouble());

        rengine.end();
    }
}
```

Figure 3. An illustrative example of regression analysis

5. Conclusion

Semiconductor manufacturing is so lengthy and complicated that modeling for simulations can be an overwhelming task. To reduce modeling efforts, previously created and used simulation models are often reused after modification. Even so, repetitive and tedious works are required for a new but modified simulation model and errors can also be inherited. A generic simulation model has been used to reduce the modeling time and to make simulation model more reliable. This research proposes an improved generic simulation framework. The framework is capable of handling large volumes of data, and various statistical tests. Also, another time reducing technique is suggested such as using metadata of AutoCAD file.

The framework is expected to improve the productivity of simulation analyses. In more detail, with the framework, the preparing time for input data and errors are reduced significantly, and handling large volumes of data get much easier. Automatically created simulation models based on the design of experiments are run without interruption. Finally, studying every aspect of alternatives thoroughly become possible with powerful statistical package and database software.

References

[1] Applied materials, “Applied AutoMod 12.2.1 Beginning AutoMod Tutorial”, **(2008)**.
[2] Autodesk, “AutoCAD 2012”, DXF Reference, **(2011)**.
[3] M. Dachyar, “Simulation and Optimization of Services at Port in Indonesia”, International Journal of Advanced Science and Technology, vol. 44, **(2012)**, pp. 25-32.
[4] F. Macia-Perez, J. V. Berna-Martinez, D. Marcos-Jorquera, I. Lorenzo-Fonseca and A. Ferrandiz-Colmeiro, “A New Paradigm: Cloud Agile Manufacturing”, International Journal of Advanced Science and Technology”, vol. 45, **(2012)**, pp. 47-53.

[5] B. Kim, S. Jeong, J. Shin, J. Koo, J. Chae and S. Lee, "A Layout- and Data-Driven Generic Simulation Model for Semiconductor Fabs", IEEE Transactions on Semiconductor Manufacturing, vol. 22, no. 2, **(2009)**, pp. 225-230.
[6] S. C. Mathewson, "The Application of Program Generator Software and its Extensions to Discrete Simulation Modeling", IIE Transactions, vol. 16, no. 1, **(1984)**, pp. 3-18.
[7] G. T. Mackulak, F. P. Lawrence and T. Colvin, "Effective simulation reuse: A Case Study for AMHS Modeling", Proceedings of the 30th Conference on Winter Simulation Conference, **(1998)**, pp. 979-984.
[8] M. Pidd, "Guidelines for the Design of Data Driven Generic Simulators for Specific Domains", Simulation, vol. 59, no. 4, **(1992)**, pp. 237-243.
[9] M. Z. A. Rashid, M. S. M. Aras, M. A. Kassim, Z. Ibrahim and A. Jamali, "Dynamic Mathematical Modeling and Simulation Study of Small Scale Autonomous Hovercraft", International Journal of Advanced Science and Technology, vol. 46, **(2012)**, pp. 95-114.
[10] S. Taktak, W. Hachicha and F. Masmoudi, "A Computer-assisted Performance Analysis and Optimization (CPAO) of Manufacturing Systems based on ARENA® Software", International Journal of Advanced Science and Technology, vol. 39, **(2012)**, pp. 93-106.
[11] E. Song, B.K. Choi and B. Park, "Event Graph Modeling of a Homogeneous Job Shop with Bi-inline Cells", Simulation Modelling Practice and Theory, vol. 20, no. 1, **(2012)**, pp. 1-11.
[12] J. Wang, Q. Chang, G. Xiao, N. Wang and S. Li, "Data Driven Production Modeling and Simulation of Complex Automobile General Assembly Plant", Computers in Industry, vol. 62, no. 7, **(2011)**, pp. 765-775.
[13] J. Wy, S. Jeong, B. Kim, J. Park, J. Shin, H. Yoon and S. Lee, "A Data-Driven Simulation Model for Logistics-embedded Assembly Manufacturing Lines", Computers & Industrial Engineering, vol. 60, no. 1, **(2011)**, pp. 138-147.
[14] "JRI - Java/R Interface" , at http://www.rforge.net/JRI/index.html.

Authors

Dae-Eun Lim

He received the B.S. degree from Korea University, Seoul, Korea, in 2004, and the M.S. and Ph.D. degrees from KAIST, Daejeon, in 2006 and 2009, respectively, all in industrial engineering. He was a senior engineer at Samsung Electronics, Suwon from 2009 to 2012. He is currently a faculty member of the division of business and commerce at the Baekseok University, Cheonan, Korea.

Minseok Seo

He received the B.S. degree in civil and environmental engineering from Korea University, Seoul, Korea in 2001. He received the Ph. D. degeree in industrial engineering from Pennsylvania State University, U.S.A in 2009. Dr. Seo is a senior researcher at Samsung Electronics, Hwasung, Korea.

