The System Architecture Concept in Axiomatic Design Theory: Hypotheses Generation & Case-study Validation

by

Tae-Sik Lee

B.S., Mechanical Engineering
Seoul National University, 1997

Submitted to the Department of Mechanical Engineering
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1999
[June 1999]

Signature of Author.....
Department of Mechanical Engineering
May 7, 1999

Certified by.....
Nam P. Suh
Ralph E & Eloise F Cross Professor &
Mechanical Engineering Department Head
Thesis Supervisor

Accepted by.....
Ain A. Sonin
Chairman, Department Committee on Graduate Students

The System Architecture Concept in Axiomatic Design Theory: Hypotheses Generation & Case-study Validation

by

Tae-Sik Lee

Submitted to the Department of Mechanical Engineering
on May 7, 1999 in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Mechanical Engineering

ABSTRACT

This thesis presents an investigation of a System Architecture (SA) as defined within Axiomatic Design (AD) theory. A system design is essentially the same as the design of simple products, when considered within the theory of Axiomatic Design. This correspondence enables the Axiomatic Design framework for a design process to be applicable to the design of systems. A list of hypotheses is presented to formalize the issues concerning system design using Axiomatic Design and the concept of System Architecture. The aim is to provide an understanding of the concept of System Architecture in Axiomatic Design for designers to design a system in a rational approach.

The hypotheses address three issues: system representation, system design, and the coordination of system elements. Fundamental concepts within Axiomatic Design support these hypotheses, and an industrial case study illustrates the validity of the hypotheses. The concept of System Architecture and system design using Axiomatic Design discussed in this thesis can contribute to the development of a rational approach to the design of a system.

Thesis Supervisor: Nam P. Suh

Title: Ralph E & Eloise F Cross Professor & Mechanical Engineering Department Head

Acknowledgements

I would like to thank Professor Nam P. Suh for his vision and support throughout this research project. This thesis was possible because of his guidance and patience to give an opportunity to explore ideas. I would also like to thank Dr. Larry H. Oh of Silicon Valley Group, Inc. for his intellectual and moral support. My sincere thanks go to Derrick Tate for his invaluable suggestions and sound criticism.

I would like to thank all the members of the Axiomatic Design Group at MIT for their constant contribution to the current body of knowledge within Axiomatic Design. Deserving special mention are Jason Hintersteiner and Sunghee Do for the numerous inspiring discussions and general assistance throughout my time at MIT.

I am grateful to Silicon Valley Group, Inc. for funding this thesis work and offering an opportunity to validate the theory through an industrial application.

Most of all, I wish to thank my family for their unconditional support and constant encouragement through the years of education. Without their love, the task of completing this thesis would not have been as enjoyable as it has been.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. Introduction

The role of design in product development and manufacturing has been widely appreciated in recent years. In product design research, it has long been said that the design determines 80% of manufacturing cost. In order to meet such industrial needs for competitiveness, extensive work has been done for the last 20 years in the area of product design [1-4]. As our interest moves toward systems design, more complex and larger scale design object, there arises a strong need for a rational methodology that addresses system design issues. The topic of systems design has not been explored enough to provide a formal theoretical framework for systems design, and as a consequence, system design has been practiced mostly based on heuristic or empirical lessons.[5, Ch.4] In Axiomatic Design theory, it is claimed that the Axiomatic Design approach for a simple product design is equally applicable to the design of complex systems. By following the Axiomatic Design processes, a designer/design group will be able to come up with a proper system design and generate a comprehensive roadmap for a system.

In this thesis, hypotheses for systems design, based on Axiomatic Design, are discussed in detail, and a case study is presented to illustrate the application of the hypotheses.

1.1 Motivation

The ultimate goal of engineering design is system design. It not only requires solving specific engineering problems, but also coordinating every component's functionality to yield a final goal of the system. A variety of issues are involved in system design. A designer needs to make sure that each of the well-functioning components are integrated properly, so that they perform together to produce the system's objective. A design manager should also overcome the complexity in order to plan and execute a large number of design tasks. Most tasks related to system design have been done heuristically or empirically, because system design practice has lacked a formal framework. Furthermore, heuristic approaches emphasize qualitative guidelines, which can result in expensive and unpredictable after-built testing processes.

There is a need to create a rational approach for system design. Rational system design should be method-based and include system-level analysis, as opposed to the design practice based on heuristic lessons, solution-based approaches using building codes and handbooks, etc. Systems engineering has been accepted as a general problem-solving paradigm that divides the decision-making process into a series of stages [6]. These stages typically cover the activities in a design process: comprehensive problem identification; formalization of goals; generation of solution concepts; evaluation of concepts and selection of the best solution; and implementation. For example,

Templeman [7] identifies four fundamental questions to be answered during the system design process:

1. What are the goals that must be achieved?
2. How are the goals achieved, limited, and related to one another within each solution alternative?
3. How are solution alternatives assessed in terms of good and bad?
4. How can the best solution be identified?

Traditional systems engineering, however, does not provide explicit concepts and a formal strategy with which one can answer the above questions. In Axiomatic Design theory, the concepts of design domains, zigzagging between the domains, and the two design axioms are clearly presented, and those concepts provides formal and rational answers to the above questions. By offering designers a rational system design methodology, system design practice is improved.

1.2 Objectives

Ulrich and Eppinger [8] propose three needs which a systematic design process can provide for a successful product development. A systematic design process provides 1) an explicit decision making method that is well understood by all team members, 2) a checklist to ensure that important steps are not forgotten, and 3) a natural documentation of the decisions made for future reference. The above needs are equally applicable to system design, since system design is an extension of product development.

The concept of System Architecture in Axiomatic Design theory satisfies these needs for rational and systematic system design. It is an efficient way of dealing with system design. It guides a designer through the system design process, helps with managing system design and the interrelationships of the components, and finally yields a well-organized document as well as the design itself.

The objective of this research is to elaborate the concept of System Architecture within Axiomatic Design theory. A theoretical review on the System Architecture concept is followed by a case study conducted to verify the concept and illustrate the application of the System Architecture.

1.3 Research method

This section discusses the method for researching System Architecture, that was followed in this thesis. This concerns the start and progress of research in Axiomatic Design and System Architecture.

Axiomatic Design begins with two axioms, namely, the Independence Axiom and the Information Axiom. The axioms are stated in abstract language and provide generic guidelines for designers. Suh

developed a number of theorems and corollaries out of the axioms to facilitate their use. However, many people who try to apply Axiomatic Design to their own design projects say that it is difficult to apply Axiomatic Design to an actual problem. Why is it so difficult? Are there some straightforward steps? Answering these questions requires serious research activity. Case-studies have been considered as an effective means to learn Axiomatic Design, and various case-study works have been done by Suh and his colleagues. Still, there is a need for a clear definition/description for each concept used in Axiomatic Design, rules/steps to follow in the decomposition process, guidelines with which a designer can select a proper set of functional requirements, and so forth.

One can elaborate the concepts used in Axiomatic Design by answering every prospective question: e.g., why are functional requirements inherently independent of each other? How do functional requirement and design parameter differ? It is theoretical research to some extent, and requires intensive philosophical efforts. Developing rules/steps in decomposition process has been addressed recently by Tate[9]. The rules are developed through the observation of various design processes, and verified by large-scale case-studies. Writing guidelines for functional requirement selection is not simple If they are stated in purely abstract terms, they are not likely to be readily useful. On the other hand, if they are biased toward specific fields or applications, they will not be applicable to problems in other disciplines. This conflict results from the peculiar nature of design research. Nordlund and Tate categorize the fundamental knowledge areas within design as the following[10].

- Designers,
- The design process,
- The design object,
- The relevant field(s), and
- Resources (such as time and money).

The areas, which are connected by Axiomatic Design theory, are the design process and the design object. Specifically, the concepts relating to the process are domains and zigzagging, and the concepts relating to the design object are independence and information. It is noteworthy that Axiomatic Design itself does *not* cover the field where the individual design object belongs: specialized methodology will correspond to the relevant field. Thus, an individual theory in design discipline does not necessarily need to encompass all of the above five areas.

The issues discussed in this thesis fall into the category of concept definition/description within Axiomatic Design theory. Axiomatic Design theory claims that the practice of System Architecture improves the result of the design. Although the concept of System Architecture has been present in

Axiomatic Design since the early 1990s, little work has been done regarding System Architecture and application of it. The concept of System Architecture needs to be elaborated, and the methodology must be applied to actual system designs to verify the concept. As stated above, the System Architecture concept is defined and described by answering the expected questions Those questions are formalized in terms of hypotheses. The hypotheses are also the statements the author believes to be true based on Axiomatic Design theory, regarding system design. Based on the fundamentals of Axiomatic Design theory, the author can argue that these hypotheses are reasonable. Then, they are tested through the case study.

The first phase of the research is to list a number of hypotheses regarding SA. Logical reasoning will be given for each of the hypotheses. The justified hypotheses, then, will be applied to an actual system design for the verification.

1.4 Thesis overview

This thesis is structured in the following way: the concept of System Architecture is reviewed, issues related . System Architecture – in general, system design – are discussed, and finally a case study is presented. The following list summarizes the content of each chapter.

- Chapter 2 reviews the System Architecture concept, and discusses activities in the Axiomatic Design process from the perspective of system design.
- Chapter 3 presents a list of hypotheses concerning System Architecture. The hypotheses are described based on Axiomatic Design theory and practices. At the end of the chapter, an approach to test and validate the hypotheses is discussed.
- Chapter 4 provides a case study to illustrate system design based on System Architecture concept. A case study on photoresist processing system covers the practice of Axiomatic Design for a system design and verifies the hypotheses with detailed examples.
- Chapter 5 concludes the discussion presented in this thesis.

2 System Architecture

2.1 System Architecture in Axiomatic Design

A System Architecture (SA) is an integrated form of application of Axiomatic Design (AD) in the following ways: 1) it is created SA when the AD approach is applied to a design of a system, the ultimate goal of design, 2) it includes all the aspects of AD process, and 3) it provides structured information about a system based on the fundamental concepts of AD. The basic motivation of SA is that we want to design a system in a *systematic* manner as opposed to the more chaotic approach typically applied in system design.

All of the above statements will be thoroughly covered in this paper. Before starting the main discussion, a couple of keywords must be clarified so that readers understand the terminology used in this thesis. Unfortunately, misunderstanding of terminology is not uncommon in the design research community and frequently hinders researchers from communicating effectively.

2.1.1 What is a 'System' and 'design of a system'

The very first term to be defined is a *'system'*[1]. It is, indeed, very difficult to define precisely, and there are various definitions for a system. Among the various definitions is Suh's definition [11].

> A system may be defined as an assemblage of sub-systems, hardware and software components, and people that is designed to perform a set of tasks to satisfy specified functional requirements and constraints.

Different definitions for *system* are also available from other researchers. One of the broad definitions of a system is as follows: *a system is an integrated set of elements that accomplishes a defined objective* [12]. Another definition is available: *a system is a set of different elements which together produces a result which they could not achieve separately* [13].

Although it is not easy to obtain a unique definition in formal language, it seems that most people intuitively know what a system means. Observing the above three different definitions, one can notice two essential concepts in common throughout those definitions. They are

- A large number of components are integrated to yield a system.
- A system has its ultimate goal(s) which is(are) achieved through the accomplishment of distributed functions.

[1] It is reasonable to confine our interest within the area of design theory.

The above two statements are necessary conditions for the definition of a system. In addition to these conditions, it should be noted that whether an entity is system or not is determined in the context of a discussion. For example, a computer, which consists of main unit, monitor, keyboard, and a mouse, is a system in the context of computing machine design, but it is only a subsystem when we think of it as a part of Internet environment. Despite this relativity, agreeing to these two necessary conditions of the system definition, provide a sufficient basis to proceed with a discussion. We need to clarify the meaning of the terminology used in defining a 'system', e.g. component, subsystem, etc. INCOSE has good definitions regarding system design, and they are shown here to give a common understanding of the terminology used in this thesis. Table 1 shows a list of terminology definitions commonly used in systems engineering field. In Figure 1, the hierarchical nature of a system is illustrated using this terminology.

The result of a system design is a physical realization/entity that performs the ultimate functionality as initially intended. Although it may sound obvious, a system design is, therefore, the process through which a designer obtains the result: defining system's functionality, assigning elements to satisfy lower level functions, detailing them into subsystems, etc. The practice of system design is *systemized* with the help of the Axiomatic Design approach.

Fundamental concepts of Axiomatic Design can be summarized to three categories: design domains, mapping/zigzagging, and design axioms. Based on these concepts, the AD process – or the AD approach – is composed of several steps. The first step of the design process is to state Functional Requirements (FRs). Secondly, these FRs are mapped into the physical domain to yield Design Parameters (DPs). The third step is to examine the relationship between FRs and DPs and determine

Term	Definition
Segment (element)	A major product, service, or facility of the system, e.g., the aircraft element of an air transportation system
Subsystem	Applied to apparatus which performs a cleanly separated function, such as communications, electronics, structures, or controls
Assembly	An integrated set of components and/or subassemblies that comprise a defined part of a subsystem, e.g., the fuel injection assembly of the propulsion subsystem
Subassembly	An integrated set of components and/or parts that comprise a well-defined portion of an assembly
Component	Comprised of multiple parts; a cleanly identified item
Part	The lowest level of separately identifiable items

Table 1 . Definitions on system design terminology [12]

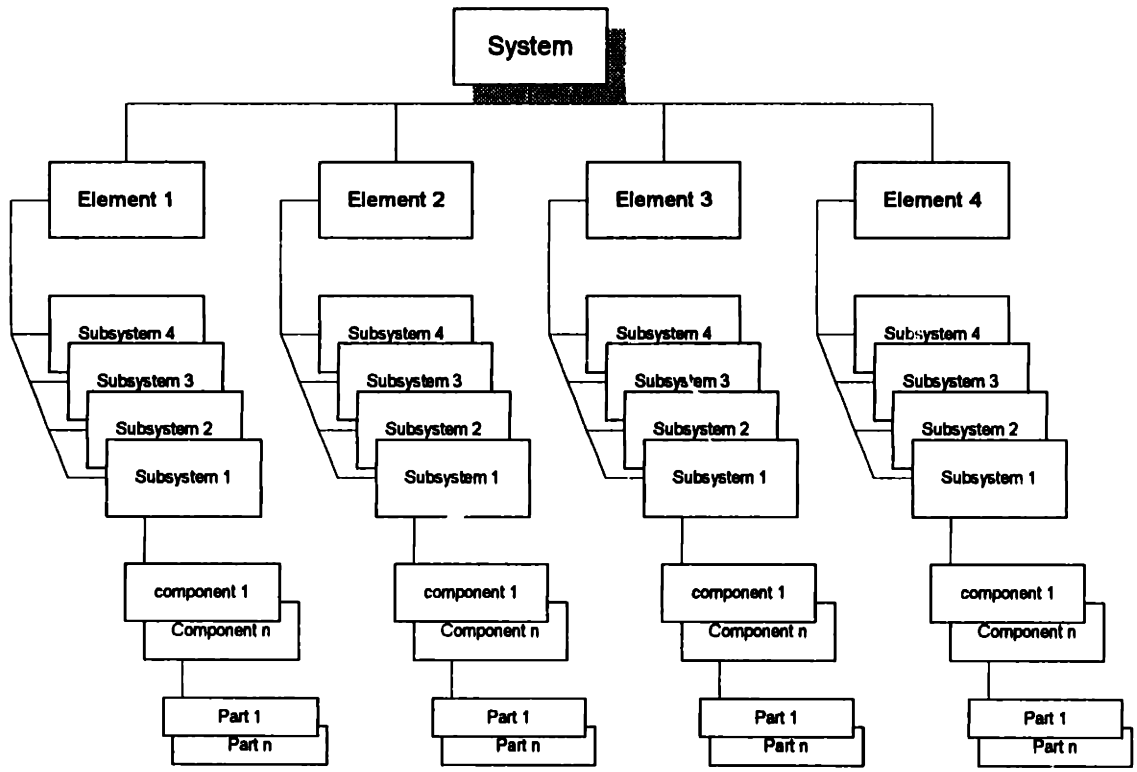


Figure 1 . Hierarchy of System Elements [12]

the design matrix. If the choice of DPs satisfies axiom 1, which is the Independence Axiom, these steps are repeated in the same manner to the next level of design. This is known as *decomposition*.

From the AD perspective, the design of a system is not fundamentally different from the design of simple mechanical products [11]. A system can be designed by following the AD approach described above, and all of the fundamental concepts of AD are applied to a system design. Therefore, all of the AD aspects are necessarily incorporated in system design activities, and the resultant is a System Architecture, hereafter called SA. If a system has only one or two layers of hierarchy, which is unlikely in any real case, a system design is not that different from the straightforward application of AD. However, as the scale of a system gets larger in terms of the level of the hierarchy and its span, it is not a trivial task to view the design object with a system perspective. SA, once created, can provide a whole view of a system.

One can raise the following question regarding the system design activity: how far shall a system designer reach in designing a system? When can a system designer say the system design is 'done?' Obviously, a system design doesn't have to focus on part level design: for example, a track system – pre/post photolithography processing system – designer may not be interested in selection of

bearings used in a robot carrier.[2] AD names the last module of the FR/DP hierarchy as a *leaf module*. In any kind of design, the design is said to be done when decomposition reaches to leaf module for every branch of the decomposition. Then, how do we know that the decomposition reaches a leaf module? What is a leaf module? Tate argues that if an operand of a sub-FR is different from that of parent-FR, design for that branch is done by assigning appropriate sub-DP [9, pp. 15-18]. In his logic, it is the FR that determines whether a module is a leaf or not. Yet, it is more intuitive to consider the DP as a key factor in determining an answer to this question. If a selected DP is something that already exists and either needs no re-design or needs no further decomposition for the purpose of system interaction investigation, then the module – FR/DP pair – is considered to be a leaf. The concept of a leaf module as an indicator for design completion can be applied to system design in the same way. Whether it be a component, subsystem, or even an element, once a designer specifies all the requirements and constraints enough to consider the design as a *local* design task, the system design can be said to be completed.

2.1.2 *System architecture (SA)*

The concept of *system architecture*, in general, is receiving enormous attention in recent days, because many people recognize the lack of formal theoretical framework for complex systems [11, 14]. The term, *system architecture* can be named and defined in various ways. Ulrich defined '*product architecture*' as the scheme by which the function of a product is allocated to physical components [15]. INCOSE defines '*system architecture*' as the arrangement of elements and subsystems and the allocation of functions to them to meet system requirements. In the AD theory, system architecture is used to mean the structure by which each physical parameter interacts with each of system's functional requirements. It is the aggregation of all of design decisions during the AD process. Therefore, SA does imply not only the allocation of functions to system's (physical) *components* but also the interconnection between those functions and physical *parameters*. Tate also states that SA can be exploited as a tool for decision making in AD[9], which is the prominent value of the AD SA concept. Each of these definitions has its proper form with which the researcher can use the term "system architecture" in his research context. Ulrich's product architecture shares some common concepts with the AD SA. Ulrich's product architecture is defined as follows: 1) the arrangement of *functional elements*; 2) the mapping from *functional elements* to *physical components*; 3) the specification of the *interfaces* among interacting physical components. The differences are 1) Ulrich's product architecture is used in the context of the effect of a product's physical structure onto the manufacturing firm performance; 2) the meaning of the mapping process is relating functional

[2] Detailed discussion on track system architecture is available in Chapter 4.

elements with physical components[3]. The concept of system architecture recognized by INCOSE does not capture the interaction between functional and physical elements. It seems that INCOSE's initiative is to focus more on the managerial aspect of system design. Throughout the thesis, by system architecture I will follow the definition used in AD theory, since it captures the most essential activity, making design decisions, when discussing the design of a system.

Potential applications of the system architecture are discussed by Suh [5], who lists some of the applications which are related to system design and operation. They are 1) diagnosis of system failure, 2) engineering change orders, 3) job assignment and management of a system development team, 4) distributed systems, 5) system design through assembly of modules, and 6) system consisting of hardware and software. In this thesis, these applications will be discussed in detail in terms of hypothesis verification.

2.2 Creation of System Architecture

As stated in the previous section, creating SA is equivalent to the design a system, and an SA is created by following the AD processes. In this section, the AD processes will be discussed, and elements of SA will be described.

2.2.1 *Design processes in Axiomatic Design*

- State FRs and Cs

After investigating the customers' needs, a designer – or group of designers – states FRs explicitly in solution neutral terms. The FRs should be the functions a design object has to perform. – The object could be a system/element/subsystem/component/part, e.g. a designer's task could be a part design or a system design. Thinking in the solution neutral environment guarantees the freedom in the subsequent step of the design, which is devising DPs. Stating FRs is a stage of problem definition, and it cannot be emphasized too strongly. Stating/selecting FRs is a totally subjective activity and there is no dependable criterion by which the set of FRs is determined to be 'good'. With the parent DP given, different designers may come up with different sets of sub FRs, depending on his/her own background knowledge and bias. Is there any kind of logical measure based on which one set of FRs is said to be "better" than the other? So far, the activity has depended almost only on the definition of FR: *a minimum set of independent requirements that 'completely' characterizes the functional needs of the product in the functional domain.* Accordingly, the only way to make the judgement is to examine – by

[3] For example, following Ulrich's definition, a coke can may be categorized into integral architecture, which means one-to-many mapping from functions to components. On the contrary, from the AD perspective, a coke can is still one-to-one mapping from functional domain to physical domain with 'decoupled' interaction. In AD, physical component must not be confused with design parameter.

any means – whether the parent DP is completely realized and consequently the parent FR is satisfied. Recently, Tate discussed the idea of 'completeness' of the selected FRs and developed rules that could be a guide to FR selection.[9] A *complete* set of sub-FRs is defined as a set of sub-FRs that is sufficient for producing the parent-level FR, that also satisfies the parent-level Cs and the parent-level DM, and that describes the functions of parent-level DP.

With this limited conceptual guideline, it is never an easy task to come up with the "right" set of FRs. It requires creativity and logical reasoning, and often involves quite a few iterations to conceive a proper set of FRs. The reason why iteration is needed is because verifying the correctness of FRs in current practice necessarily incorporates bottom-up characterization as opposed to top-down approach, a part of the main philosophy of AD. In spite of the ambiguity, a lot of designers seem to manage to get reasonable sets of FRs in terms of achieving parent level FRs. They tend to have more trouble stating FRs in solution neutral terms and keeping the number of FRs to a minimum. The process of conceiving FRs is illustrated in Figure 2. Since the question "is parent DP (→FR) achieved?" sometimes cannot be answered until the design is decomposed further, there will be another feedback loop departing from somewhere far downstream of the whole design process, which is also regarded as part of the iteration process.

The discussion of the above paragraphs is better understood with an example. The example shown here is adapted from the case study which is presented in section 4.2.

Below is an example of top level functional requirements. It is a list of FRs for the design of photoresist processing equipment.

FR1 = coat wafers with *desired resist film*

FR2 = *develop* exposed film

FR3 = process wafers at *desired rate*

FR4 = *control* the system functions

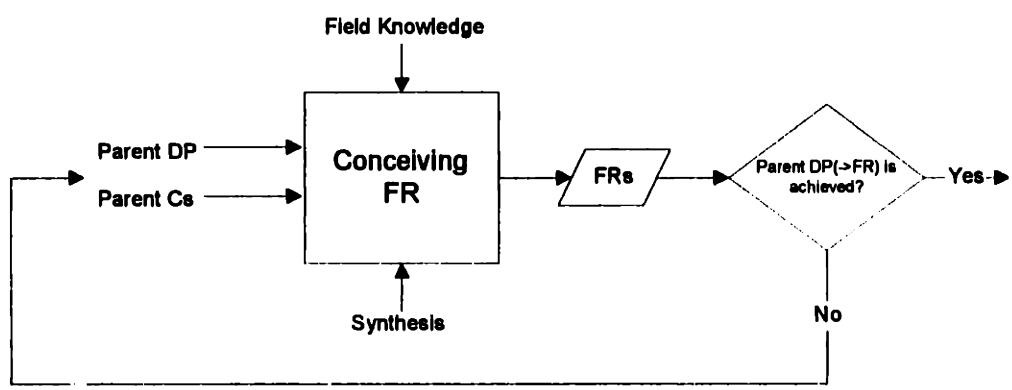


Figure 2. Synthesis of FRs

How were these 4 FRs obtained? At first, a designer looks at the parent FR and parent DP. In this case, the parent FR is "fabricate wafers before/after photolithography process." The parent DP to satisfy the FR is "track system", which is clustered to a photolithography system. Though the concept of "track system" at this level is not detailed at all, a designer has at a least rough picture of what it will look like. Besides the parent FR/DP, there are many system level constraints such as cost, footprint, throughput, safety, just to name a few. Those constraints have potential impacts on the design decisions, although the impacts are not directly expressed in FR statements. Given the parent FR/DP and constraints, a designer thinks about what type of functions the parent DP has to perform in order to achieve the parent FR and satisfy the constraints. Here comes a designer's process-knowledge and logical reasoning based on engineering science. In the example, four types of FRs are identified as a result of this thought process. They are the essential functions required for the track system.

As mentioned above, a different designer may come up with different set of FRs even with the same inputs, i.e. same parent FR/DP, and same set of constraints. Although two sets of sub-FRs are not identical, if the highest level FR, which is "fabricate wafers..." in this case, is eventually achieved, then both sets should carry the essential functional requirements somewhere in the FR hierarchy. By its definition of 'equivalent design', the difference is neglected after all[4].

Now that we have FRs, the next step is to devise DPs that satisfy the FRs – this is known as the mapping process.

- Devise DPs for each of FRs

Each FR must be mapped into the physical domain, which results in a set of DPs. When choosing DPs, we have the freedom to come up with any feasible idea only if they satisfy FRs subject to the related constraints. Thanks to this freedom, we may have several alternative DPs for a particular FR among which we can choose the best one. The mapping process is illustrated in Figure 3.

This mapping process, of course, requires quite an amount of knowledge and effort regarding the design object: e.g. material science, fluid dynamics, electromagnetics, etc. The more a designer has field knowledge, the higher chances he/she can come up with good design because he/she will have a better quality database for DPs. The typical notion for 'engineering design' refers to this mapping

[4] Definition S1: Two designs are defined to be "equivalent" if they perform the same set of the highest level FRs within the bounds established by the same set of constraints, even though the mapping and decomposition process might have yielded designs that have substantially different lower level FRs and all DPs for each of these designs. Equivalent design doesn't mean that two 'equivalent design' will have the same quality and/or cost effectiveness. Equivalency just implies that two – or more – equivalent design all satisfy the highest level FR.

process: extensive research, creative thinking, physical synthesis, etc. A lot of misunderstandings about AD occur here because some people think that AD should be able to make this challenging process easy. It is similar to saying that a computer should solve all the problems if we just give the problems to a computer. What AD really does is to provide sound criteria for judgement on whether a certain choice of DP will make system work or not. The AD process requires a designer to come up with any feasible solution to an FR, and helps him with making the best choice out of his/her own database.

Conceiving a number of candidates for DP is still in the role of human designer. A design theory, including AD, can help the designer by leading him to the right way of thinking. As an example, among the design methodologies that help a designer in that way is TRIZ – The Theory of Inventive Problem Solving. It guides a designer to formulate the existing problem. It also tries to help a designer with his design task by providing detailed rules to various kinds of problems, which are known as 'algorithms' in TRIZ. These efforts are useful when a designer seeks a core understanding of the problem when the original design violates the design axioms, and those are consistent with AD. Although both AD and TRIZ are very useful at the problem definition stage and provide some help when conceiving a particular solution, those activities are still highly dependent on the synthetic ability of human beings.

An example of DPs is shown below. These DPs are mapped from the previous FR example.

DP1 = coating process modules

DP2 = developing process modules

DP3 = system configuration

DP4 = system control command

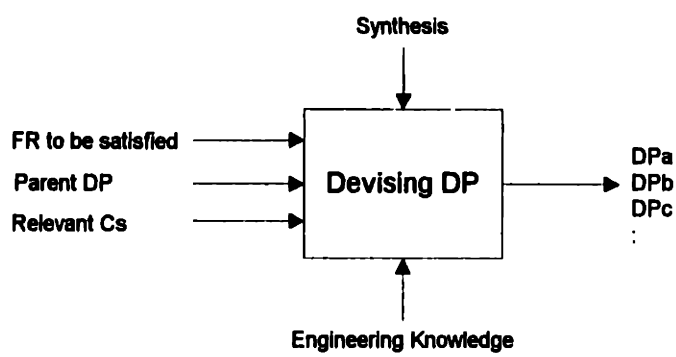


Figure 3. Synthesis of DPs

FR1 is 'coat wafers with desired resist film,' and the parent DP is 'track system.' Relevant constraints at this level of design are cost, throughput, process flexibility, serviceability, etc. Based on those inputs, DP1 is chosen to be 'coating process modules.' By choosing 'coating process modules' as DP1, the designer implies that the system needs a series of processes to perform resist film coating. With the information, a modular structure is adopted as a reasonable way to satisfy FR1 with specified constraints. In the same way, 'developing process modules' are selected as DP2. FR3 is 'process wafers at desired rate,' and the identified DP for the FR3 is 'system configuration.' DP4 is 'system control command' which will be the control software of the track system. The above set of DPs is abstract and has little information with them since they are at such a high level. All that is stated with DPs is that the system will have different modules to perform corresponding functions. It is acceptable because it gives the designer a conceptual direction for further design.

- Determine Design Matrix

Once we have FRs and DPs, the next step is to examine their relationship by determining a design matrix. Design matrix is defined as:

$$\begin{Bmatrix} FR1 \\ FR2 \end{Bmatrix} = [DM] \begin{Bmatrix} DP1 \\ DP2 \end{Bmatrix} \qquad \textbf{(2.1)}$$

DM is 2×2 matrix in the above case, and it shows the relationship between FRs and DPs. That is, it shows whether *DPi* will affect *FRj* or not. Strictly writing,

$$\begin{Bmatrix} \Delta FR1 \\ \Delta FR2 \end{Bmatrix} = \begin{bmatrix} \frac{\partial FR1}{\partial DP1} & \frac{\partial FR1}{\partial DP2} \\ \frac{\partial FR2}{\partial DP1} & \frac{\partial FR2}{\partial DP2} \end{bmatrix} \begin{Bmatrix} \Delta DP1 \\ \Delta DP2 \end{Bmatrix} \qquad \textbf{(2.2)}$$

Based on the definition of DM, a typical question a designer asks when determining each element of design matrix, is:

Does DPj affect FRi?

The question is refined maintaining equivalence,

Does a change in DPj, consistent with fulfilling FRj, affect FRi?

or

Does the choice of DPj affect the choice of DPi?

We can answer these criteria questions relatively easily when design reaches leaf level, where mathematical expressions may be available. Answering the question, which is equivalently defining

the relationship between FRs and DPs, is not a design decision at the leaf level, but a fact to be analyzed. On the contrary, at early stages of design, it may not be answered easily in the same way. The reason is that we have a relatively small amount of information about the chosen DP. The intellectual activity a designer performs when determining the design matrix at this level can be considered with two different point of views. First, it is more like a decision of design direction than solid analysis with concrete DP. What a designer is really doing is to conceptualize abstract design and impose a direction on later design. Therefore, the types of questions are changed to:

***Shall** DPj affect FRi?*

***Shall** a change in DPj consistent with fulfilling FRj, affect FRi?*

***Shall** the choice of DPj affect the choice of DPi?*

In the ensuing design, a designer should be always well aware of the design direction imposed by the design of the previous level. Another way of viewing the process is that it involves the further detail of the DP, which is not yet decomposed, to some degree. If the answer to the question is 'yes,' it must have some supporting argument such as 'The aspect xx of DPj will affect FRi.' The aspect may come from the designer's reasoning toward a sort of decomposition of the DP. The latter argument implies that bottom-up thinking is necessary to determine the design matrix. In fact, the next step of design process, consistency checking, shows the bottom-up feedback characteristic in AD processes.

Returning to the example, a design equation is constructed as following:

$$\begin{Bmatrix} FR1 \\ FR2 \\ FR3 \\ FR4 \end{Bmatrix} = \begin{bmatrix} X & O & O & O \\ O & X & O & O \\ X & X & X & O \\ X & X & X & X \end{bmatrix} \begin{Bmatrix} DP1 \\ DP2 \\ DP3 \\ DP4 \end{Bmatrix} \qquad (2.3)$$

FR1/FR2 and DP1/DP2 have an uncoupled relationship. Since DP1 and DP2 are chosen to be modular with respect to each other, it is reasonable to proceed with the subsequent design in an uncoupled way. FR3 is 'process wafers at desired rate.' DM(3,1) and DM(3,2) are marked 'X' indicating that DP1 and DP2 will affect FR3. The modular structure of DP1 and DP2 and various detailed features which will be addressed in further decomposition will affect FR3 as design requirements or constraints so that DP3 has to be designed (decomposed) in a way that it accounts for these effects. DM(1,3) and DM(2,3) are 'O's since we *do not want* DP3 to affect FR1 and FR2. FR1 and FR2 are the most fundamental requirements for a track system, and no effect from other DPs is acceptable. One of the efforts to make DM(1,3) and DM(2,3) to be 'O' is presented in the case study section, which shows the algorithm preventing irregular delay time in transport. All of

DP1, DP2, and DP3 have some effects on FR4 in a sense that FR4 can be detailed only after DP1, DP2, and DP3 are characterized. Thus, DP4, 'system control command' will be designed last.

Once a DM is determined with given DPs, we have to examine the type of design: uncoupled, decoupled, or coupled. If the design is either uncoupled or decoupled, it is an acceptable design, and we can go to next level of design. But if the design is coupled, we have to revisit the choice of DPs and do something – change the DP that causes coupling or pose certain constraints on the DP so that it will not cause coupling. During the design at the high level such as one in the example, the design matrix is determined by our own decisions to some degree, and we explicitly decide not to have a coupled design. Maintaining the validity of high level decisions should be emphasized more than the decisions themselves. The feasibility of the decisions will be fully realized through further design decomposition.

- Decision consistency checking

As mentioned above, maintaining former design decision has much importance. Therefore, there should be a systematic way to force a designer to take it into account. A multilevel design matrix has been used by a few [6, 16], and the author believes it is a good tool to view the decision consistency.

The basic motivation for a multilevel design matrix originated from the investigation of off-diagonal terms in a design matrix. When one decomposes the design, the focus is normally on the diagonal elements in the design matrix. Little information is known about off-diagonal elements. In order to justify the decision on an off-diagonal element in the design matrix, one needs to examine the relationship between sublevel FRs and DPs. It is simply done by subdividing columns and rows into sublevel ones. For example[5]:

Suppose that the first level design equation is

$$\begin{Bmatrix} FR1 \\ FR2 \end{Bmatrix} = [DM0] \begin{Bmatrix} DP1 \\ DP2 \end{Bmatrix} = \begin{bmatrix} X & O \\ X & X \end{bmatrix} \begin{Bmatrix} DP1 \\ DP2 \end{Bmatrix} \qquad (2.4)$$

With the above DM0, we can say that DP1 affect FR2, and DP2 does not have any effect on FR1.

The second level design equations are

$$\begin{Bmatrix} FR11 \\ FR12 \end{Bmatrix} = [DM1] \begin{Bmatrix} DP11 \\ DP12 \end{Bmatrix} = \begin{bmatrix} X & X \\ O & X \end{bmatrix} \begin{Bmatrix} DP11 \\ DP12 \end{Bmatrix} \qquad (2.5)$$

[5] The example given in this section is arbitrary one. A more detailed example for a track system is available in section 4.2.2.

$$\begin{Bmatrix} FR21 \\ FR22 \\ FR23 \\ FR24 \end{Bmatrix} = [DM2] \begin{Bmatrix} DP21 \\ DP22 \\ DP23 \\ DP24 \end{Bmatrix} = \begin{bmatrix} X & O & O & O \\ O & X & O & O \\ X & X & X & O \\ X & O & O & X \end{bmatrix} \begin{Bmatrix} DP21 \\ DP22 \\ DP23 \\ DP24 \end{Bmatrix} \quad (2.6)$$

To examine whether the choice of sublevel DPs is consistent with higher level design decision, we construct multilevel design matrix which includes the first and second level design matrices as shown in Table 2.

FR11	X	X	O*	O*	O	O	DP11
FR12	O	X	O	O	O	O	DP12
FR21	O	O	X	O	O	O	DP21
FR22	O	O	O	X	O	O	DP22
FR23	X	O	X	X	X	O	DP23
FR24	X	O	X	O	O	X	DP24

Table 2. 1st/2nd level DM

The unshaded portion of the design matrix is DM1 and DM2, respectively. The emphasis here is on the shaded area. Filling the elements in the shaded area requires examination of the cross effects such as the effect of DP1x on FR2y. If any one of DP2x has strong effect on FR1y, then it will violate the earlier design decision. One should impose some constraints or specify conditions with which it will not happen. The superscripts in Table 2 indicate there are constraints that DP21 and DP22 must satisfy. If it is inevitable for a sub DP to have effect that leads to the violation of the former design decision, there are two options for a designer: choose new DP or revise the higher level DM. In the latter case, the higher level DM must be either uncoupled or decoupled matrix, otherwise, serious redesign from the 1st level needs to be done.

The steps illustrated above are repeated until a designer gets to the final stage of design. It is a continuous application of AD.

2.2.2 *Elements of System Architecture*

System Architecture is created while we follow AD design processes. Consequently, the elements of SA include all the things we bring forward during AD activities.

Fundamental elements of SA are, needless to say, FRs, DPs and design matrices, and constraints. For the system design, they combine to form a large hierarchy, which is equivalently a set of *design*

decisions. As we saw in section 2.1.2, "design decision" is a keyword in describing SA in AD. In AD process, decisions are made at various stages of the design while affecting each other.

First of all, the selection of FRs is the most fundamental design decision. After a designer studies customer needs (CNs), he/she must make a decision which one should be mapped into functional domain and which ones should not. Since all of CNs are not likely to be mapped into top level FRs, the designer also has to decide an appropriate hierarchical level for mapping CNs. From the second level hierarchy, the upper level DPs give a boundary to a designer's scope in establishing FRs. In the narrowed-down design space, he/she selects *minimum set* of FRs. A minimum set of FRs is a set of FRs that is *necessary and sufficient* to describe the functionality of a parent DP. As mentioned in the previous section, section 2.2.1, defining FRs can be considered a subjective task, and involves design decisions.

Secondly, choosing a DP among alternatives is a design decision. The mapping process itself is not a decision making process: it is concept generation rather than decision making. Only when there is more than one option is the decision making activity involved.

Determining the design matrix is also a design decision. In the previous section, it was argued that at the high level of system design, determining design matrix elements is more like a decision on design direction. The high-level design matrix directly shows the designer's intention for the ensuing design. At lower-levels, the design matrices represent a physical analysis of FR/DP interactions. It determines the type of the design: uncoupled, decoupled or coupled. A coupled design matrix indicates that a designer should change the previous decision - choice of DPs or higher level design matrix.

The above set of design decisions is all incorporated in the SA. In other words, they are important elements of the SA. SA shows a clear picture of the whole set of design decisions using its appropriate forms, which are explained in the next section. Later on, when an alternative design decision is made, the decision will be substituted into the structure of the SA. To summarize, the essence of SA is its hierarchical structure of design decisions.

Documenting System Architecture is important. Documentation structure includes the essential elements – all of the design decisions. Therefore, it consists of FR/DP/design matrix/constraints table, description of FRs/DPs, justification of DM, and visual representation. A standard template to document SA has been developed in Axiomatic Design Group at MIT [17]. In the case study, an example SA documentation is presented.

2.2.3 *Representation of System Architecture*

One of the critical factors in SA representation is how clearly and concisely a system can be represented or visualized. It should give an insight into a system with a quick look. It should also provide essential information in a concise format. There are three ways in AD to visualize SA: 1)Tree diagram, 2)Module-Junction Diagram, and 3)Flow chart.

2.2.3.1 Tree Diagram

A tree diagram just shows the hierarchical structure of a system. It is has two types of tree structures one of which is for FRs and the other for DPs. Figure 4 shows the example of a tree diagram.

This representation is good for showing concisely the system's hierarchical structure of FRs and DPs. However, the tree diagram is nothing but a depiction of the hierarchical structure.

2.2.3.2 Module-Junction Diagram

A Module-Junction diagram is created to represent SA more efficiently. It contains more information than a tree diagram. It indicates the type of each design matrix as well as the hierarchical structure of a system.

A module is, conceptually, equivalent to the row of the design matrix that yields the FR when it is provided with the input of its corresponding DP[11]. By its definition, a module represents the relationship between FR and related DP(s). In mathematical form,

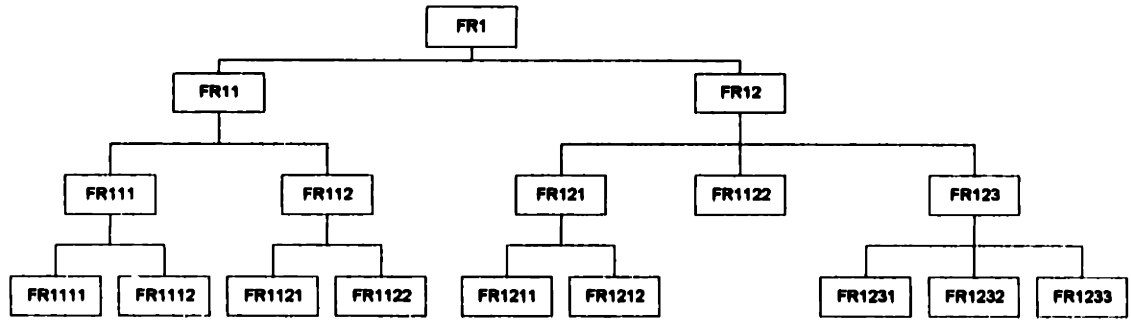
$$Mi = \frac{\partial FRi}{\partial DP1}\frac{DP1}{DPi} + \frac{\partial FRi}{\partial DP2}\frac{DP2}{DPi} + \cdots + \frac{\partial FRi}{\partial DPi} + \cdots + \frac{\partial FRi}{\partial DPn}\frac{DPn}{DPi} \qquad (2.7)$$

$$FRi = Mi \times DPi \qquad (2.8)$$

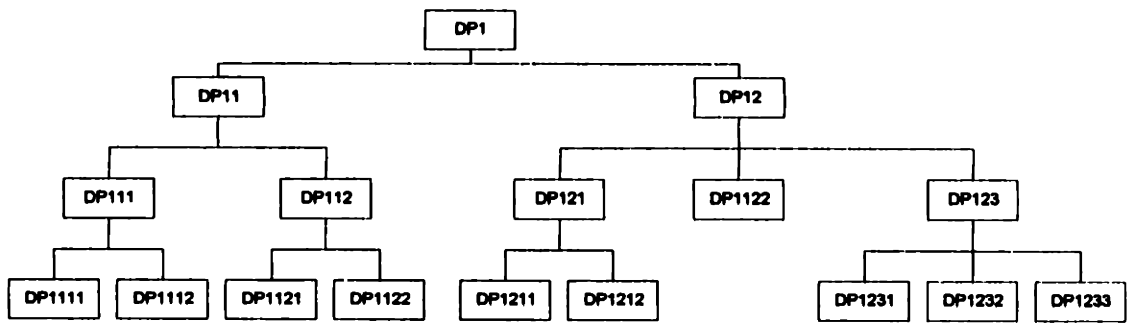
where Mi is the module relating DPi with FRi.

In an uncoupled design matrix, a module is simply the diagonal element in DM. In a coupled case, it includes off-diagonal elements and DPs as well as diagonal elements. This is explained in more detail by Suh [11]. When we mention a module, we imply FR, DP, and their relationship as a whole. Therefore, the Module-Junction diagram is a combined representation of FR tree, DP tree, and DM.

A junction represents the type of each design matrix. A junction is a conceptual space where modules are combined. Each module is processed at a junction according to instructions given by the DM. There are 3 kinds of junctions: summing junctions (S), control junctions (C), and feedback junctions (F). They represent uncoupled, decoupled, and coupled design matrices respectively.



(a)



(b)

Figure 4. Tree diagram for a system (a) FR tree (b) DP tree

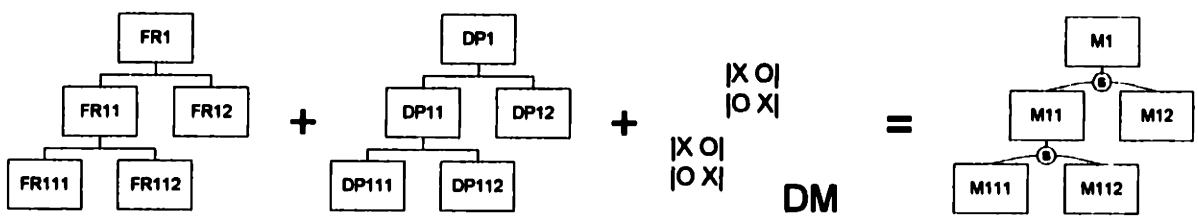


Figure 5. Illustration of Module-Junction diagram

A Module-Junction Diagram is based on the definition of module and junction. Basically, it is similar to a tree diagram. Additional features are

- Combination of FR tree and DP tree diagram
- Information on the type of design matrix

The Module-Junction diagram is good in a sense that it shows both the hierarchical structure and the type of design at each level – whether it is uncoupled, decoupled or coupled. In case of decoupled and coupled designs, knowing the detail of the design matrix, such as the sequence of design (operation), source of coupling, etc., is essential. Although we know the information by examining each DM, a representation scheme that visualizes the information is more advantageous. The next

section discusses another representation scheme, a flow chart, which satisfies the aforementioned need of visualization.

2.2.3.3 Flow chart

The third way to represent SA is with a 'flow chart.' It is concise in its form and has explicit information from each design matrix: it shows the details of design matrices pictorially as well as the types of design matrices. The definition of module enables this simple representation of the SA in a form of the flow chart [11]. The Flow chart uses the symbolic expression of junctions in Figure 6.

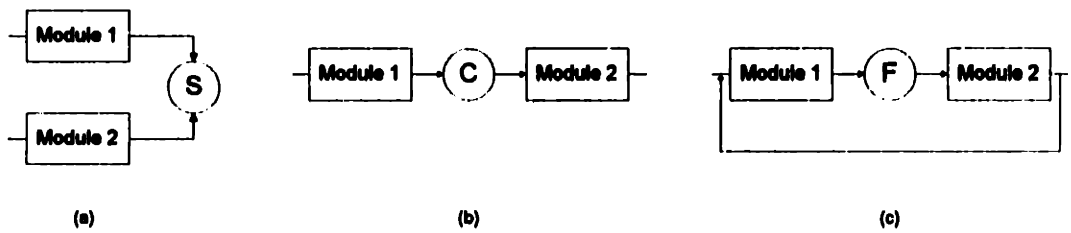


Figure 6. Junction Properties of the flow chart: (a) Summing Junction (Uncoupled), (b) Control Junction (Decoupled), (c) Feedback Junction (Coupled) [18]

Figure 7 is an example of a flow chart, based on the one 1st level design matrix and four 2nd level design matrices. M_i represents each module, and the flow chart clearly shows the structure of each design matrix. For example, we see from Figure 7 that the design matrix for FR1x/DP1x is diagonal matrix which means an uncoupled design. On the other hand, we see that the design matrix for FR4x/DP4x is a decoupled design, which represents a full-triangular matrix. Supposing that each design matrix is created properly, we can see the effects among components immediately with a flow chart, and we also get the right sequence to design the system.

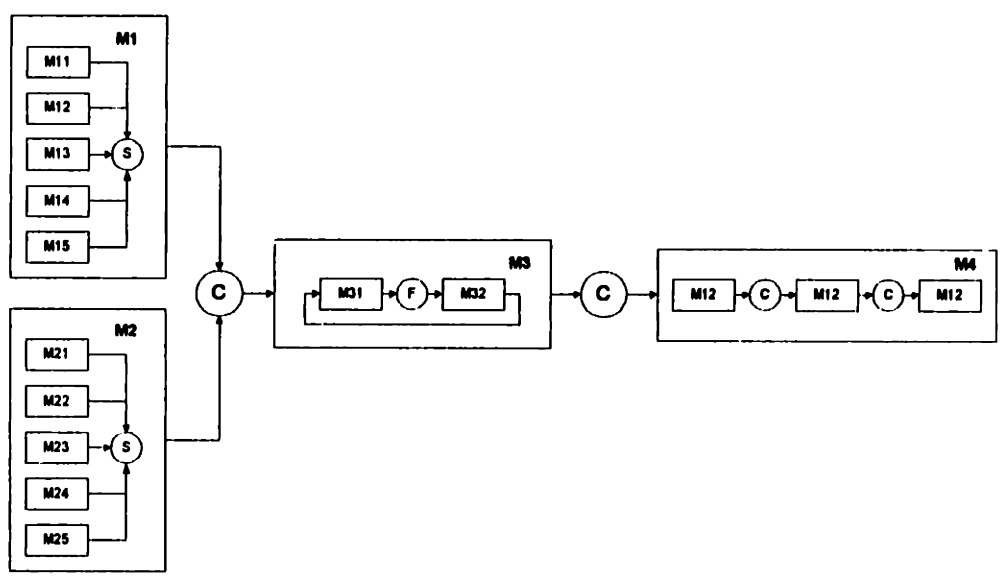


Figure 7. Flow chart (1st and 2nd level)

2.3 Summary

Chapter 2 discussed the topic of system design from the Axiomatic Design perspective. The author characterizes a *system* with two aspects: an integration of a large number of components and distributed functions to achieve the goals as a whole. Those characteristics imply that the fundamentals of system design are the same as the fundamentals of the design of simple mechanical products within the Axiomatic Design framework. *System architecture* is defined as the structure by which each physical parameter interacts with each of a system's functional requirements. The design processes in Axiomatic Design theory was reviewed to describe how to create a System Architecture - equivalently, how to design a system following the design processes in Axiomatic Design. The last two sub-sections, 2.2.2 and 2.2.3, explained the elements of a system architecture in terms of design decisions and representation schemes for the system architecture.

Chapter 3, based on the concepts presented in this chapter, discusses the system design and system architecture in more detail.

3 Hypotheses on Axiomatic Design and System Architecture

A list of hypotheses concerning SA is presented in this section. In section 4.1, the hypotheses are stated and described. Section 4.2 justifies the hypotheses based on AD theory and practices. Section 4.3 discusses how to test and validate the hypotheses with a practical application.

3.1 List of hypotheses

In this section, a list of hypotheses is presented and described. The hypotheses listed here have a hierarchy. There are three high level hypotheses, the second and the third of which are directly adapted from Suh [11, Chapter 4.2].

- ***A system (design of a system) is represented in SA***

System Architecture incorporates the essential elements, which show the structure of a system created.

- ***A system is designed using a top-down approach based on the fundamental concepts of AD***

A designer starts system design by conceiving top level FRs/DPs and proceeds with the design through detailing them level by level. By doing so, he/she can complete the system design.

- ***The complex relationships between various components of a system are coordinated and managed using the SA***

Most systems consist of a large number of various sub-components. In order to achieve the ultimate goal of a system, each component should be designed in a certain order and their interaction should be controlled properly. The SA contains necessary and sufficient information for this task.

The first hypothesis relates SA and system representation, the second is about SA and system design, and the third concerns SA and system management/control. The last two hypotheses are elaborated with lower levels hypotheses listed below:

- **A system is designed using a top-down approach based on the fundamental concepts of AD**
  - *The AD approach forces the designer to think in the right direction*

    By following the AD approach, a designer is kept on the right track, which should lead to a better design solution in less time.

This hypothesis can be refined further:

- *Mapping & zigzagging is the right way to design and is always possible*

Mapping & zigzagging, or equivalently decomposition, is required in order to do design properly, and it is possible regardless of the hierarchical level within the system.

- *A designer can proceed with design given only a small amount of information[6] at a high level*

It is often the case with system design at high levels that DPs can not be defined precisely. Nevertheless, high level DPs can be decomposed into sub level FRs/DPs successfully.

- *Design knowledge is captured in a structured way*

Design knowledge includes a thorough understanding of the design objectives (FRs), physical methods (DPs) to achieve those objectives, relationships between them, constraints and instruction for physical implementation. These are captured and organized in the SA clearly.

- *Concept of Concurrent engineering is necessarily incorporated in AD system design*

Concurrent engineering has two essential characteristics: 1) it is a *concurrent* process, and 2) it is carried out by a multifunctional product development team [19][7]. Following AD system design implies these types of activities.

- *Design lead time is decreased*

Applying AD to complex system design has the benefit of shortened design lead time. This is illustrated in Figure 8.

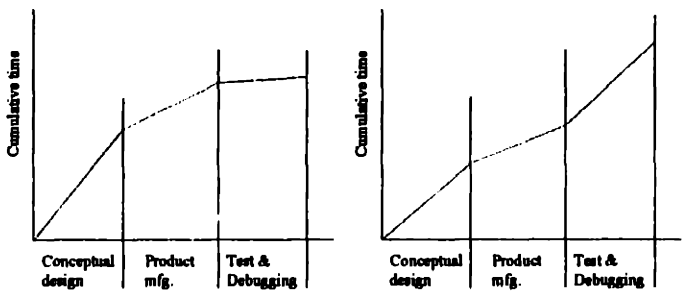


Figure 8 Design lead time (a) AD process (b) typical design process

- **The complex relationships between various components of a system are coordinated and managed using the SA**

[6] The term 'information' is used as general word. It is not related to the word 'information,' as defined by the Information Axiom.

- *SA predicts the effects of the Engineering Change Orders*

The meaning of ECO in a system is how many sub-components must be changed as a result of a change of certain component – or requirement. SA shows the effect of DP changes in the system and of FR changes.

- *SA provides instructions for controlling the system*

With the SA, one can easily obtain instructions regarding the sequence of system control.

- *SA is useful as a diagnostic tool for a system*

SA can serve the task of diagnosis and be the basis of diagnostic tool development by indicating potential sources of the failure.

A full list of the hypotheses mentioned above is shown in Figure 9.

- A system (design of a system) is represented in SA
- A system is designed using a top-down approach based on the fundamental concepts of AD
  - The AD approach forces the designer to think in the right direction
    - Mapping & zigzagging is the right way to design and is always possible
    - A designer can proceed with design given only a small amount of information at a high level
  - Design knowledge is captured in structured way
  - Concept of Concurrent engineering is necessarily incorporated in AD system design
  - Design lead time is decreased
- The complex relationships between various components of a system are coordinated and managed using SA
  - SA predicts the effects of the Engineering Change Orders
  - SA provides instructions for controlling the system
  - SA is useful as a diagnostic tool for a system

Figure 9 List of Hypotheses

3.2 Argument on validity

In this section, the hypotheses presented are discussed based on AD theory and the nature of the SA.

[7] Clausing refers this concept to 'basic concurrent engineering' to differentiate it from 'world-class concurrent engineering'.

- A system (design of a system) is represented in SA

Suh describes the design process as four distinct activities: problem definition, creative process, analytical process and ultimate check[1]. During an *original system design process* that emphasizes problem definition and design solution synthesis, system representation corresponds to an aggregation of design decisions. Therefore, a system representation should include explicit delineation of functions and selected design concepts (physical domain) of the system. At the same time, it should clearly depict interrelationships between those components. A designer starts system design by stating fundamental FR(s) and then devises physical entities which will satisfy the FRs. Then, the design matrix is determined. These processes are repeated at the next level, which is known as decomposition. Along with the design processes, system representation is done in a form known as the System Architecture. SA covers all of these areas in its simple format as mentioned in section 2.2.2 and 2.2.3. Providing that a designer ends up with final design of a system through AD approach, the system built and SA should be identical. *For improvement of an existing design*, the analytic aspect dominates the design process. A designer is supposed to observe existing physical entities first, which are in the physical domain, and figure out which FRs are subject to each of the DPs. The focus is to organize FRs/DPs/design matrices to match the existing system. Out of an arbitrary[8] assemblage of physical entities, the designer has to construct a hierarchy of FRs and DPs. The resulting SA shows the structure of the existing system. Different aspects from the SA of an original design are that all the DPs are conjectured from existing physical entities and that we should not assume those are the right set of FRs since our basic motivation for analysis is to improve the existing design.

- A system is designed using a top-down approach based on the fundamental concepts of AD

In designing a system, it is obvious that a reasonable designer will do his design in a hierarchical manner even though he is not aware of AD theory. According to Suh, that is because of the hierarchical nature of design objects. Suh indicates that the hierarchical nature of design objects justifies the top-down approach [1, p.4]. The design object of a system also has the hierarchical nature, and Figure 1 shows the physical hierarchy of a system. The top-down approach is one of the fundamental concepts in AD theory. Decisions are made in the order of importance by decomposing the problem into a hierarchy. Through the top-down process, upper-level design decisions propagate

[8] The term 'arbitrarily' is used in the sense as follows:

Physical entities are integrated in ordered manner to achieve the overall goal, which is the top level functional requirement. However, in most cases of existing design, functional requirements at every level of a system are not clearly defined.

to lower-level design. It is exhibited by the facts: 1) previous choice of DP drives sub level FRs (zigzagging) and 2) the upper-level design matrix can lead to constraints that impact the choice of sub-level DPs. The top-down approach results in a hierarchical FR/DP structure, which is equivalent to a system's hierarchical structure. Therefore, considering the hierarchical nature of a system's functionality, the top-down approach is, in principle, the most rational way of designing a system.

It should be noted, however, that some of the activities during the top-down work flow need to be done in a bottom-up fashion. These are mentioned in section 2.2.1: verifying the correctness of FRs and determining design matrices. INCOSE describes this process as the combination of a top-down & bottom-up approach followed by compromise: "*These (requirements and specifications of a system) are developed in a top-down/bottom-up fashion, with systems engineers defining what they want in a top-down fashion and subsystem engineers describing what they can provide in a bottom-up fashion*". [12, p3] AD views the process as design limitations or constraints realized afterward, since AD wants to maximize a designer's freedom during the conceptual design stage.

- *The AD approach forces the designer to think in the right direction*

Following the AD approach means that a designer will proceed with a design through repeating a series of characteristic activities: 1) identify FRs in a solution-neutral environment, 2) devise DPs, 3) determine design matrices and make sure that the design axioms are satisfied, 4) check design consistency with respect to higher-level design decisions, and 5) repeat steps (1)-(4) at the next level. The 'right' direction in design thinking should lead a designer to *better* design solutions in *less* design time. The AD process will guide a designer toward this goal – better solution in less time. First of all, AD enables a designer to define the term, '*better*.' It is one of AD's fundamental contributions to the area of design research. By elucidating the criteria of good design explicitly, AD directs a designer to the better design solution among the alternatives. Secondly, one of the claims in AD is that time/resource-consuming iteration process after the concept generation stage is reduced as a result of following the AD process: a system satisfying the two axioms is guaranteed to be working properly. The essential activities that contribute to those benefits include stating FRs, classification of uncoupled/decoupled/coupled designs, measuring information contents, and the decomposition process.

- *Mapping & zigzagging is the right way to design and always possible.*

Mapping and zigzagging are combined to complete the decomposition process, and decomposition is one of the crucial activities in design because of the hierarchical nature of the design object. Mapping and zigzagging are carried out between different domains and throughout all levels of the hierarchy.

Between the functional domain and physical domain, mapping and zigzagging are required: choose FRs, map them into DPs, select sub FRs, etc. It is the process of moving between 'what' and 'how,' which is the core of the design activity. There seems to be no doubt about it. Zigzagging, however, is not feasible between the customer domain and the functional domain because of the nature of customer domain. A designer does map between the two domains but does not zigzag between them. Customer Needs (CNs) are given from the customers, and a designer doesn't have freedom to define CNs. Sub-level CNs – if there is a hierarchical structure in CNs – are not driven by upper-level FRs. Defining sub-DPs based on parent FRs is the essence of zigzagging. Because of the lack of the characteristic, zigzagging is not performed between the two domains.

The hierarchical level of design does not affect applicability of mapping and zigzagging. Although it is possible to map and zigzag between domains regardless of the hierarchical level, there are some situations, especially at high-levels in a hierarchy, when it is difficult to devise solid (i.e. concrete or physically meaningful) DPs. In the example of automobile hood lock and release mechanism design [5, Chapter 2, p.58],

FR1 = Hold the pin in the locked position

FR2 = Release the pin from the lock position to an open position

After the mapping process, DPs are,

DP1 = Mechanical locking mechanism

DP2 = Release mechanism

DP1 and DP2 carry no detailed information about the mechanisms. Then, they are decomposed to,

FR11 = Locate the pin at the locked position

FR12 = Lock the pin

Again, the above two FRs are mapped into physical domain.

DP11 = A cam plate that provides dead stop position

DP12 = Rotating cam plate with a slot for the pin and a cam profile to engage a spring loaded ratchet mechanism (to keep the ratchet spring loaded against the cam surface)

Looking at the example, at the second level, we clearly see the mapping from FR11 to DP11 and from FR12 to DP12. On the other hand, one may say that the first mapping process,

from FR1 to DP1, is redundant and that one could decompose FR1 into FR11 and FR12. But, the process of going from FR1 to DP1 is not redundant because it implies its sub-FRs within itself: choosing DP1 is more or less equivalent to defining sub-FRs. If, for example, DP1 were 'magnetic locking mechanism', the subsequent design would yield different sub-FRs than FR11 and FR12. Therefore, mapping and, consequently zigzagging, have significance and are possible even at the high-levels of design hierarchy.

- *A designer can proceed with design given only a small amount of information at the high level*

In the original design process, a designer is likely to encounter situations in which he/she does not have enough knowledge to envision the rest of the design. However, a small amount of information does not necessarily mean an insufficient amount. In other words, at a high-level of a design hierarchy, the amount of essential information - knowledge - may be relatively smaller than that needed at the lower-level. The design activities related to concerned FR/DP are 1) determination of the design matrix at that level and 2) next level decomposition. The sequence of performing those activities in AD process is 1)-2). Activity 1) can be done without fully detailed information about the concerned DPs. In this activity, a designer imposes design direction for further decomposition in the absence of information on the details of the DPs. To impose design direction without constraints by lower-level DPs is another form of solution-neutral environment. Activity 2) requires enough information to figure out the next level functional requirements. By repeating these processes, the amount of information – design knowledge – will be increased to the amount needed to implement the final design.

- *Design knowledge is captured in a structured way*

Design knowledge includes the design goals, physical methods to achieve those goals, relationships between them, and instructions for physical implementation.

Design objectives are explicitly stated in terms of FRs and their target value with tolerances. An FR is, by its definition, the requirement to be satisfied, and equivalently 'what' a designer wants to achieve. Starting from abstract FRs at high levels of design, it is detailed with continuing decomposition. The detailed FRs are itemized design objectives.

Physical methods to achieve these FRs are expressed in terms of DPs in the physical domain. Along with the detailing process for FRs, DPs are also specified in detail. A designer should try to come up with alternative DPs to yield a database of DPs for one FR. He/she will choose one DP as the best solution among them, and the rest of the database, which are alternative DPs, will facilitate any possible change of DPs later on.

The design matrix at each level shows the relationship between each of the FRs and DPs. The dependencies between FRs and DPs are clarified in the form of the matrix. In the case of engineering design, each element in the design matrix has to be expressed, eventually, with mathematical formulas so that it gives exact information between the DP value and FR value. The interrelation between different branches of a design hierarchy, e.g. DP112 and FR221, is also clarified by evaluating the multi-level design matrix.

The above three categories of design knowledge are captured and organized directly in the SA documentation. They are, as discussed in section 2.2.2, the basic elements of the SA. Instructions for physical implementation, however, are not explicitly available for the SA. The instructions must include configuration of the system, layout, etc. It requires more study to see whether that type of information can be captured in clear way in the SA.

- *Concept of Concurrent engineering is necessarily incorporated in AD system design*

In terms of AD, concurrent engineering can be interpreted in two ways: 1) concurrent work in the functional domain, the physical domain, and the process domain, and 2) extensive and multidisciplinary thought given to every aspect of the system. The first aspect is discussed by Suh [5, pp.37-38]. He presents the requirements for concurrent engineering in terms of two design matrices, one for FR-DP and the other for DP-PV. Since FR is eventually mapped into PV, the resultant design matrix [A][B][9] should not be a coupled design matrix.

The AD process for a system design also involves the second aspect of concurrent engineering. Following the AD process properly necessarily involves the multifunctional workforce. The first step of the system design in AD is to state top-level FRs, and the top-level FRs certainly incorporate the issues about the whole area of a system. At least for the first several levels of decomposition, engineers from different areas of development should get together and agree on the design decisions at those levels. After reaching an agreement, they will see more clearly what kind of effect one's own design has on others' designs. Leaders of each team will continually check their team's design to prevent conflicts with higher-level design decisions in a formal fashion. One critical factor in concurrent engineering is the amount of information that should be shared among different design groups. It is not efficient to share the whole set of specialized information with other design teams, which will not affect their designs at all. The SA plays the role of filtering the information in terms of the design matrices in the hierarchy. This enables concurrent engineering with the minimal, essential information to be shared.

[9] [A] represents DM for FR-DP, and [B] for DP-PV.

- *Design lead time is decreased*

The idea of doing AD is 'think in the right direction so that we make the product work well at the first test-run.' It can be rephrased as the following: prevent mistakes and expose trouble at an early design stage. As a consequence, it emphasizes the reduction – hopefully, elimination – of the follow-up effort to make the product work after it is manufactured.

The emphasis on the reduction of the rework leads to correct conceptual design, minimized iteration, understanding of cause and effect for the system components, etc. In order to achieve them, a designer should spend time stating FRs explicitly, organizing his/her thought process in AD format, deciding the design matrix at every level, and giving extensive thought to the interrelationships throughout the entire system. These series of activities require more time for the conceptual design when doing AD than the typical design approach. Although it may finish the conceptual design earlier than AD, the conventional design process has little opportunity or means to evaluate conceptual design. Therefore, a number of iterations are required even after the conceptual design is done. Moreover, the iteration does not guarantee a quick approach to a good solution. Current industrial design practice shows that much time is spent in debugging the final product, and it is one of the biggest sources of failure in on-time delivery. In that sense, in spite of the longer time for conceptual design, the final product release will be earlier, as illustrated in Figure 8.

Successful industrial applications of AD support this hypothesis. Nordlund et al. [20] reported a successful product development time reduction with an aid of the AD process. Another illustrative work was done by Do[21]. Do applied AD to the design of software, and he claimed that the design and development time was significantly reduced.

- The complex relationships between various components of a system are coordinated and managed using the SA.

Most systems consist of a large number of various elements, subsystems, assemblies, subassemblies, components, and parts arranged in a hierarchical manner. These constituents are integrated and perform their own particular functions to yield the ultimate goal of the system, which is the top-level FRs. In order to achieve the top level FRs eventually, each component should be designed in a certain order and their interactions should be coordinated properly.

Coordination of a system has a peculiar aspect in the context of an AD discussion. AD theory discusses the issue from the *design* perspective. The design aspect of coordination includes understanding of the functional/physical structure of a design object and predicting behavior on certain changes of components or conditions. Among the essential activities of that coordination are

an elucidation of complex interactions between functions and physical components, prediction of the impact of Engineering Change Orders (ECO), and system diagnosis. SA has necessary and sufficient information for the task of design control. It is already noted in 2.2.2 that the SA has design matrix as its one of the core elements. By the nature of the design matrix, SA shows the relationship – dependency, effects of change, etc– between system components. Therefore, one can use the SA to predict the effects of changes in system components and to analyze causes and effects of trouble.

- *SA predicts the effects of the Engineering Change Orders*

System Architecture is constructed based on every design matrix relating corresponding FRs and DPs. Remember the definition of the design matrix given in 2.2.1 equation (2.1) and (2.2). There, the design matrix shows the effect of any change of a DP. All of the design matrices located at the different level/branches within the hierarchy are linked together and constitute one of the essential pieces of information captured in the SA: SA shows the potential effect and/or the possible propagation of any change of DP in the system.

Nordlund discusses an impact of ECO, and describes clearly how one can use the information captured in the design matrices to predict ECO impacts. He derives two theorems [22, pp.51-53]. One is that an ECO propagates only downward in a hierarchy, and the other is that more than one DP must be changed as a part of the ECO in the case of a decoupled or coupled design. However, it turns out that the first theorem is only valid under the following conditions: the upper-level design matrix of the node where DP change occurs is an uncoupled design matrix, and the DP change does not change the upper-level design matrix. The first condition is evident if we consider the Flow chart representation. Discussing the second condition requires the definition of the term DP 'change.' It is noted by Tate that there can be several types of DP-changes[23]. He mentions 1) change of DP itself, 2) change of DP details/parameters, and 3) change of DP values.

For changes of type 2) and 3), the nature of the DP does not change. The design matrix at the level where DP change occurs remains valid, and the higher-level design matrix is also valid. SA shows exactly what should be the consequent change of the system. In case of type 1), since it is a complete change of design, the current design matrix has no information on the new DP and needs to be reevaluated. The higher-level design matrix also needs to be reevaluated for the same reason. If the higher-level design matrix changes from an uncoupled design matrix to a decoupled one, another DP at the higher-level must be changed. If the matrix changes from a decoupled matrix to a coupled one, the impact of the ECO will be catastrophic. This is upward ECO propagation. Let us look at an example.

$$\begin{Bmatrix} FR1 \\ FR2 \end{Bmatrix} = |DM0| \begin{Bmatrix} DP1 \\ DP2 \end{Bmatrix} = \begin{bmatrix} X & O \\ O & X \end{bmatrix} \begin{Bmatrix} DP1 \\ DP2 \end{Bmatrix} \quad \textbf{(4.1)}$$

$$\begin{Bmatrix} FR11 \\ FR12 \end{Bmatrix} = |DM1a| \begin{Bmatrix} DP11a \\ DP12 \end{Bmatrix} = \begin{bmatrix} X & O \\ X & X \end{bmatrix} \begin{Bmatrix} DP11a \\ DP12 \end{Bmatrix} \quad \textbf{(4.2)}$$

Suppose that DP11a changes to totally new DP, DP11b. Then, DM1a needs to be reevaluated with DP11a substituted by DP11b. It may result in new design matrix, DM1b.

$$\begin{Bmatrix} FR11 \\ FR12 \end{Bmatrix} = |DM1b| \begin{Bmatrix} DP11b \\ DP12 \end{Bmatrix} = \begin{bmatrix} X & O \\ O & X \end{bmatrix} \begin{Bmatrix} DP11b \\ DP12 \end{Bmatrix} \quad \textbf{(4.3)}$$

For this case, there is no other DP change needed. The design can be said to be improved, at least, at this level. There is, however, a possibility that DM0 has changed as a result of DP11 change. Constructing two-level design matrix hypothetically,

		DP1		DP2
		DP11	DP12	
FR1	FR11	X	O	O
	FR12	X → O	X	O
FR2		O → X	O	X

Table 3. Multi-level design matrix

Table 3 shows that, as a result of the change in DP11, the upper-level design matrix has changed to a decoupled design matrix. Now that DM0 is a decoupled design matrix, we potentially need to change DP2. Once the change in DP2 is identified, an ECO due to this change is again investigated in the same fashion.

- *SA provides instructions for controlling the system*

Before discussing this hypothesis, the term 'control the system' has to be clarified. In the context of control theory, the term 'control' denotes the manipulation of a system to yield the desired output to given input even in the presence of disturbances. The meaning of a system in this context is, of course, different from what used in this thesis. When considered from the Axiomatic Design perspective, control means an overall scheme through which we can reach the

desired response (FRs). The implication of this is then narrowed down to using the knowledge of design decisions to assign or change the inputs properly to obtain the ultimate goal of a system. Consider some cases to illustrate these concepts.

Case1)

Single level - One FR/One DP

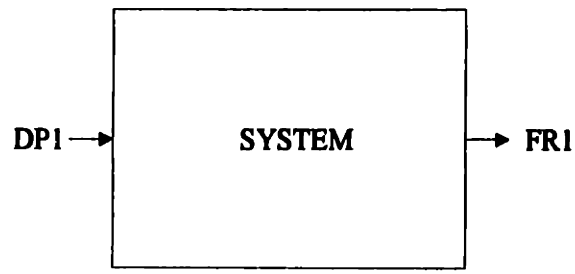


Figure 10. Single level, One FR/One DP

In this case, which is not likely to be a real case, there is no control issue from the view of Axiomatic Design. It illustrates the difference of the concept of between conventional control and AD system control.

Case2)

Single level – multi FRs/DPs

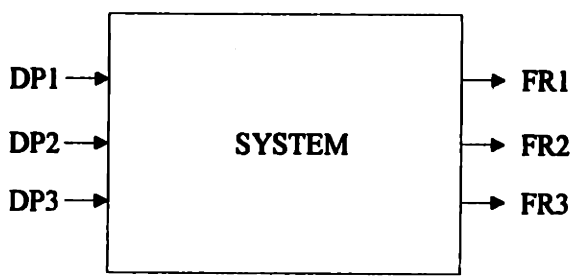


Figure 11. Single level, multi FRs/DPs

Control issue, here, is that in what manner we can achieve all 3 FRs. It is assumed that we know the relationship between each FR and DP: FR1 = f(DP1), FR2 = f(DP1,DP2), etc. Therefore, the only concern is that we have to follow the sequence indicated in the DM at this level, and that is the control of this system.

Case3)

Multilevel

In Figure 12, there are two top-level FRs in the system. They are decomposed down to the fifth level, and the lowest level DPs – or leaves – will be integrated to yield ultimate goals of the system, FR1 and FR2. When the system is built with the large number of low level DPs,

it is hard to predict the effect of one of the DPs on the entire system just with intuition. It is not simple problem to assign or change any DP to maintain the satisfaction of FR1 and FR2. There needs to be a clever and simple way to show the structure of the system with which we can control the system design.

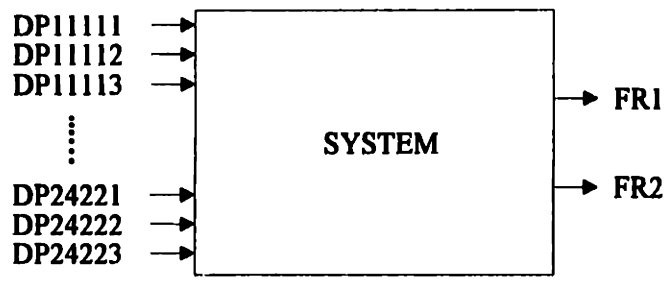


Figure 12. Multilevel

As you may have guessed from the above illustration, to control system in AD is more like to operate system: in which order each function should be performed, what type of subsequent changes are required due to a change of certain parameter, etc. The overall SA provides the operating instruction, especially in form of Flow chart.

- *SA is useful as a diagnostic tool for a system*

As already mentioned several times, the SA consists of FRs, DPs, and design matrices as its essential elements. As Nordlund explains [22, pp.49-51], the hierarchy of FRs enables an engineer to translate a problem, when the system malfunctions, into a failure of specific FR(s). If the unsatisfied FR is a non-leaf FR, then it is necessary to investigate which of the sub-FRs of the unsatisfied FR is not met. This process is repeated until the unsatisfied leaf FR is found. The process is illustrated in Figure 13.

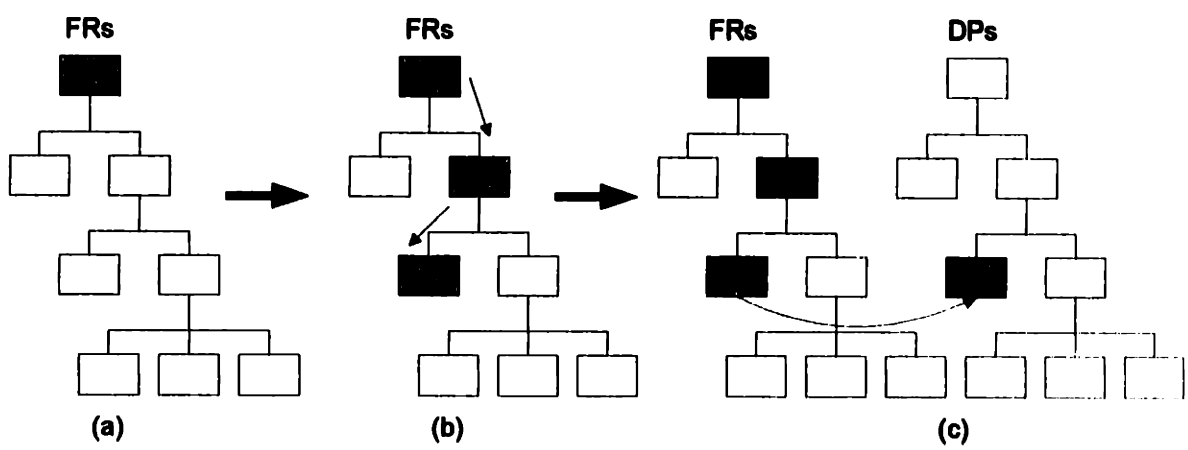


Figure 13. Identifying the cause of a problem: (a) Identify apparently unsatisfied FR (b) Trace the failure route by figuring out the unsatisfied sub-FR (c) Investigate its corresponding DP [22]

Once the unsatisfied leaf FR is identified, the design matrix at the level is investigated. If the design matrix is an uncoupled design matrix, then Figure 13(c) is correct. If the design matrix is a decoupled one and the leaf FR is affected by other DPs, all of the effects need to be examined.

Although the second process, tracing down the failure route, is sometimes difficult, the correct SA provides a clear strategy to find the problem source.

3.3 Test/validation strategy

In section 4.2, the presented hypotheses were discussed and corroborated based on AD theory. In general, these hypotheses can be tested or validated through 3 ways: 1) conducting analyses from the historical perspective, 2) performing case studies to provide supporting or counter examples, and 3) conducting design experiments [24, Ch.6].

The first approach is to make observations of previous system designs, and compare the results of the work with the expected output predicted from the hypotheses. This approach requires extensive studies for large numbers of examples, some of which have been done based on AD theory, and others of which have been done differently. The second approach can be done in two ways: 1) analyze the system designed without AD system design methodology and prove it could have been done better/worse based on AD, and 2) design a new system practicing AD from the beginning, and show better/worse performance over competing approaches. The third approach is to assign the same design task to two different design groups, only one of which is familiar with AD methodology. Careful attention must be given to the result analysis since various factors will affect the performance of the design groups.

In this thesis, the second approach is adopted to study the validity of the hypotheses. The first approach is not appropriate since, as of today, there are not many systems design work done based on AD methodology. The third approach is not feasible either because the research environment around this work has been oriented toward an industrial sponsor's product development. Design experiments are hardly feasible for a large system design in a highly competitive marketplace environment. The second approach – the first type of the approach, 'analyzing the existing system' – is in line with the project that the author has been involved in. By conducting the AD analysis of an existing product and elucidating the pros and cons of the system, better solutions for particular problems can be proposed.

Now that the reason of choosing the specific validation approach is clarified, the representativeness is to be justified. A case study, in general, should satisfy the following conditions:

- It should represent the issues a researcher is concerned with.

- It should have generic aspects so that the success of a certain hypothesis can be applicable to other cases.

The case study presented in this paper, photoresist processing machine, satisfies the above conditions. It is a highly complex[10] system, which involves the design issues presented in the hypotheses while designing. It has technically challenging problems, which must be solved in a systematic manner. Also, most of the ideas incorporated in the system's SA development are generic enough to be applicable to other system designs.

In section 4.2, the creation of a SA for a photoresist processing machine is presented to prove the validity of some of the hypotheses concerning system design. They are,

- A system (design of a system) is represented in SA.
- A system is designed using top-down approach based on the fundamental concept of AD.
- Mapping & zigzagging is the right way to design and is always possible.
- A designer can proceed with small amount of information at a high level of system design.
- Design knowledge is captured in a structured way.

Some other hypotheses are studied with more specific examples in section 4.3.

- SA is useful as a diagnostic tool for a system.
- A system is designed using top-down approach based on the fundamental concept of AD. (Small scale example)

[10] Complexity can be measured in terms of hierarchical height and span of FR hierarchy.

4 Validation of the hypotheses with example

The example chosen by the author is a photoresist processing system. It is called a 'track system.' Photoresist processing systems are among the fab's[11] most complex pieces of equipment. As the move toward smaller geometry and larger wafers progresses, track systems become more flexible and intelligent. It is a good case study for SA research because it involves a lot of design issues in terms of system design.

4.1 Photoresist processing

Photoresist processing is a part of integrated circuit device manufacturing: more specifically it is a part of microlithographic processes. The lithographic process involves transferring a circuit pattern into a polymer film, which is called photoresist, and subsequently replicating that pattern in an underlying thin conductor or dielectric film. The lithographic process is shown schematically in Figure 7.

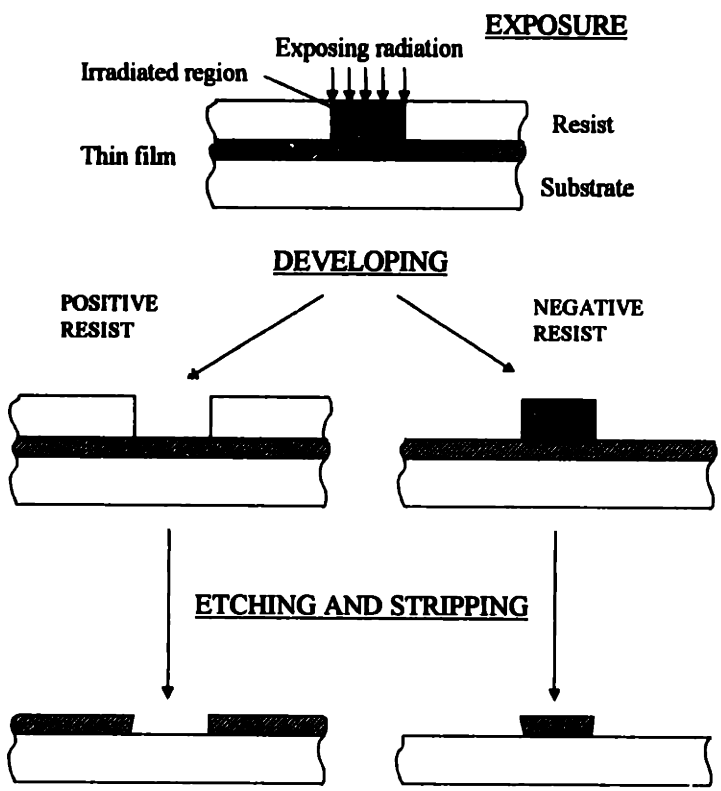


Figure 14. Diagram showing the lithographic process[25]

[11] 'fab' refers to fabrication facilities where computer chips are manufactured.

Photoresist processing covers photoresist-related steps: i.e. resist coating on wafer surface and resist film developing. The resist coating process consists of sub-process steps, likewise so does the developing process. Coating process starts with wafer surface preparation. To ensure sufficient adhesion of a resist material, the surface must be chemically compatible with the specific resist. Therefore, the wafer surface is modified with an adhesion promotor: e.g. a clean silicon surface is modified with hexamethylenedisiloxane (HMDS) to obtain sufficient adhesion for novolac-based photoresist. When the surface is ready for coating, the resist material is applied by an appropriate method. Spin-coating has long been accepted as the best coating method for obtaining a uniform, adherent, defect-free film over the entire substrate. After the wafer surface has been coated with resist material, it is exposed to patterened radiation to create a latent image in the resist layer. Photolithography uses ultraviolet radiation to transfer the pattern from the mask to the photoresist. After the polymeric resist film has been exposed, chemical reactions occur within photoresist film. It is necessary to control the environment of the exposed resist film to enable the desired reactions. A thermal treatment is used for the postexposure treatments. Once the latent image has been formed in the polymer film, it is developed to produce the final 3D images. There are two development processes, which are liquid development (wet develop) and dry devleopment (plasma). Finally, the developed resist film is thermally treated to harden the film, and the wafer is sent to next process stage: etching and stripping. The process is illustrated in Figure 8.

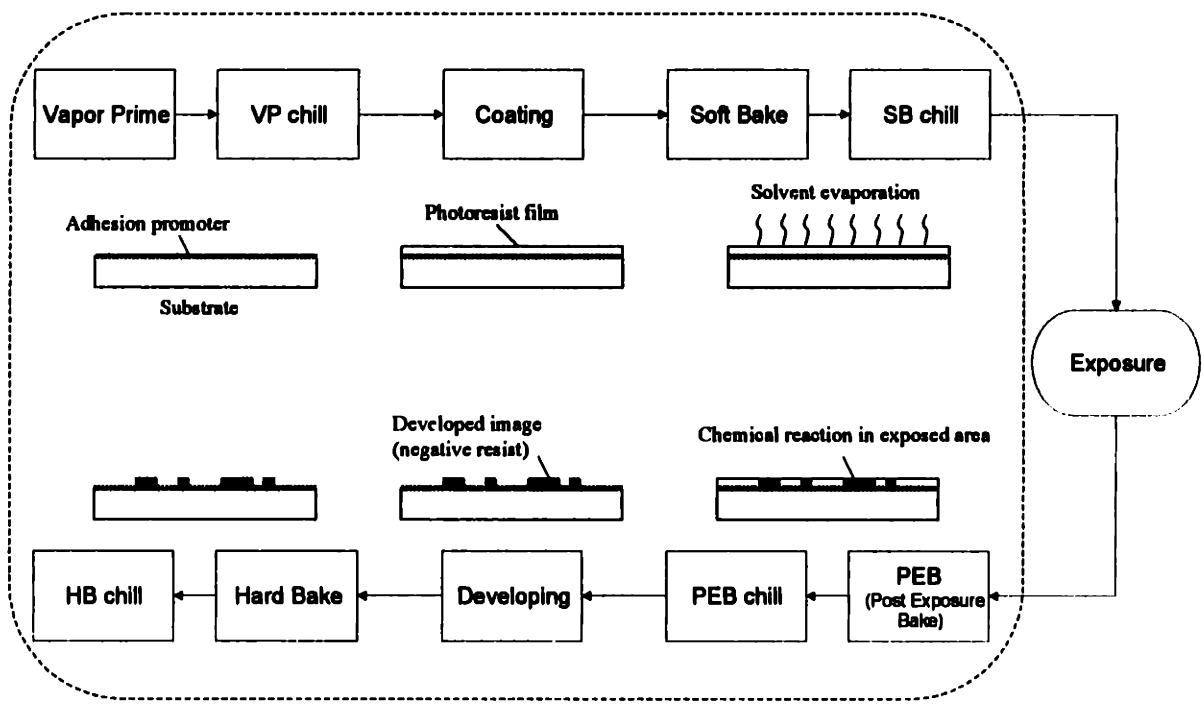


Figure 15. Photoresist processes in Track system. Area enclosed by dotted line indicates processes done by track system

The ultimate goal of the coating process is to expose surface of wafers coated with uniform, defect-free resist film. Achieving the goal becomes more challenging as feature sizes become smaller and smaller. A large number of factors contribute to the quality of resist film, and some of their effects are not yet clearly understood. Through the developing process, final 3D images are obtained. The developing process is also composed of several sub-steps, and has many parameters to be controlled to produce good quality patterns.

More detail is discussed in next section.

4.2 Decomposition for track system

This section details the first few levels of decomposition for the track system. It starts with background knowledge of track system in section 4.2.1. In section 4.2.2, actual decomposition is presented in the format of a SA document. A detailed discussion on track system design is given in this section. In the last section, the SA for the track machine is represented graphically using the three diagrams discussed in section 2.2.3.

4.2.1 Background

In the previous section, 5.1, typical photoresist processes were briefly reviewed and presented schematically in Figure 15. From Figure 15, we clearly see various functions required for photoresist processes. Roughly speaking, those processes are

Preparation for coating	}	
Coating	}	➔ Coating process
Post-coating treatment	}	
Post exposure treatment	}	
Developing	}	➔ Developing process
Finish	}	

Preparation for coating involves promoting adhesion and control of the wafer surface temperature. To promote wafer surface adhesion for photoresist material, the wafer surface should be free not only from contamination but also from moisture. Also, it must be chemically compatible with the resist. To satisfy those requirements, current technology provides a series of processes. The processes are listed below with some of the issues related to the processes:

(a) Dehydration bake (Vacuum bake)

- Moisture from the atmosphere can be rapidly absorbed by substrate surfaces, and such hydrated surfaces have been shown to reduce adhesion.
- Therefore, the time delay between the dehydration bake and coating is important.

(b) Prime (HMDS vapor priming)

- The dehydrated wafer is primed with a pre-resist coating of a material designed to improve adhesion even further.
- HMDS (hexamethylenedisiloxane) is the most widely used priming substance, surface-linking adhesion promoter.

(c) VP[12] Chill

- *Bringing the temperature of a wafer down to* a set point is a critical process because it determines the temperature distribution of a wafer just before coating process
- *Tight temperature* control is required : (18-24)±0.2 C

After a wafer surface is treated properly, the photoresist material is coated onto wafer surface. Coating the substrate with a radiation-sensitive polymer – which is called 'photoresist' – is the first process step in the semiconductor manufacturing sequence involving the resist. Spin coating has long been used for resist coating process. Spin coating is accomplished by flooding the substrate with a resist solution and rapidly rotating it at a constant speed at the order of several thousands rpm. It is summarized below:

(a) Spin Coat

- This process produces a uniform, adherent, defect-free polymeric film of desired thickness over the entire wafer surface.
- The process consists of depositing a resist solution, accelerating to a desired speed, spinning at a constant speed until the wafer surface becomes nearly dry, removing the edge bead (optional), and decelerating to stop.
- The evaporation of the volatile solvents affects the final film thickness and uniformity.
- The required uniformity within wafer is 20Å. Thickness repeatability wafer to wafer needs to be within 25Å

The post-coating treatment includes both baking and chilling processes. After spin-coating and air-drying, the polymer film contains up to 15% residual solvent and may contain built-in stresses due to the shear forces encountered during the spinning process. Both of these adversely affect subsequent processing. A baking process is introduced to deal with these problems. The baked, and subsequently hot, wafer will be cooled during the chilling process.

(a) Soft Bake

[12] (a) and (b) is combined in one step and is called the 'Vapor Prime' process.

- During this process, the solvent is driven off from the spun-on resist. It also improves the adhesion of the resist and anneals the residual stress.
- Temperature and time are the important parameters.

(b) SB Chill

- This process brings temperature of the wafer down to a set point

Through the processes described above, a wafer is readied for exposure. It is removed from the track system and sent to an exposure tool. A conventional exposure tool in current lithography process is called as 'stepper,' which represents step and repeat camera. In fact, a stepper is linked to a track system to form a clustered lithocell. A stepper is beyond this thesis' scope.

After a wafer is exposed to certain type of light source, it is brought back to track system for subsequent processes. There are two types of light sources which the track system must be compatible with: i-line and DUV (Deep UltraViolet). During the i-line exposure step, a latent image is produced in the resist film. DUV exposure creates photoacid that will, later on, act as a catalyst to produce a latent image. That makes some difference in the meaning of post exposure treatment.

(a) Post Exposure Bake (PEB)

- In case of i-line light source, PEB acts as a thermal process after an exposure step whose primary functions are to eliminate standing waves, increase resolution and extend process latitude.
- For DUV (Deep Ultra Violet) process, it is the thermal post-treatment process in which photoacid created by the exposure process acts as a catalyst and initiates a chain reaction that modifies the resist chemistry to allow latent image generation.
- Important parameters are temperature and process time (both are related to chemical reaction) and time between exposure and PEB (line width variations).

(b) PEB chill

- This process brings the temperature of the wafer down to a set point.
- Repeatable transfer time from PEB is critical.

Once the latent image has been formed in the resist film, it must be developed to produce the final 3D relief image. There are three liquid development processes employed: Spray, puddle, and immersion. Among these, spray development generally provide better pattern uniformity and process reproducibility, largely as a result of better time and temperature control and the continuous use of fresh developer. Another development process, namely, dry development (plasma) can also be considered.

(a) Develop

- The latent image is developed to produce the final 3D image of the mask pattern on the resist film.
- The important parameters are exposure time, temperature, photoresist thickness, developer concentration, developer temperature, and developer agitation method.

The last step prior to etching is post-development bake, which is conventionally called Hard Bake. Its purpose is to remove residual developing solvents and promote adhesion by annealing the film. The film is eventually expected to have better etch resistance.

(a) Hard Bake

- This process increases the etch resistance of the developed resist film.
- During this process the wafer is dehydrated to improve resist adhesion, and residual solvent is evaporated to harden the film.

(b) HB chill

- This process brings the temperature of the wafer down to a setpoint.
- This is a relatively non-critical process.

4.2.2 *Decomposition*

In this section, the actual decomposition is presented. The discussion below is organized following the SA document format. It has FR/DP tables, descriptions of FR/DP, Design Matrices, justification of design matrices, and constraints resulting from design decision.

	Parent FR	*Parent DP*	
Process	Perform physical process #1	Process module #1	T
Process	Perform physical process #2	(a) Process module #2 (1st alternative) (b) Process module #2 (2nd alternative)	In
Process	Perform process #3	Process module #3	De
Transport	Perform process #4 (transport)	Transport module	Dr
Control	Schedule and coordinate all local process functions	Command and control algorithm (CCA)	T
Support	Integrate subassemblies	Support framework	U

Table 4. Generic FR / DP table

Table 4 shows the generic template for listing FRs and DPs. The top row indicates the index at this level, where '#' refers to the index in leftmost column, and "φ" indicates the full index of the parent FR/DP. The FR column has been divided into two columns, for "name" and "description". Here, the "name" is a one or two word summary of the FR. Alternate DPs can be listed as shown in DP2 cell in the table. The last column in the table is used for verification codes, so that the designer can specify a code to indicate the verification procedure to be used to ensure that the DP satisfies its corresponding FR. Verification may be done by testing (T), inspection (I), demonstration (De), drawings (Dr), or proven unchanged technology (U). In the case study presented in this thesis, the verification codes column is not used.

The documentation is organized using the above template table followed by detailed explanations. The documentation shown here covers three levels to illustrate how the author proceeded with decomposition.

4.2.2.1 1st Level

Process	Coat wafers with desired resist film	Coating process modules	
Process	Develop the exposed film	Developing process modules	
Transport /Support	Process wafers at the desired rate	System configuration	
Control	Control the system functions	Command and control algorithm (CCA)	

Table 5. 1st Level FR/DP

Description

This is the highest level FR/DP decomposition, and thus it defines the main functions of the track machine.

- **FR1**: As a part of the photolithography system, the track machine *coats the wafer surface* with photoresist before sending the wafer to a stepper[13]. The specifications for FR1 are listed below.

  Coating thickness = 0.8 - 1.2μm

  Coating uniformity

  within wafer = 20 Å (3σ)

  wafer-to-wafer = 25Å

  cassette-to-cassette(24Hr.) = 25Å

- **DP1**: This design parameter is primarily responsible for performing the coating processes. It indicates that the overall coating process, FR1, is achieved through the use of separate sub-process modules.

- **FR2**: As a part of photolithography system, track machine *develops images* which is replicated onto the resist film during the exposure process. Specification for FR2 is listed below.

  Critical Dimension (CD)

  within wafer = 0.015μm

  wafer-to-wafer = 0.015μm

  cassette-to-cassette(24Hr) = 0.015μm

- **DP2**: This design parameter is primarily responsible for performing the developing processes. Like DP1, it indicates that overall developing process is achieved through the use of separate sub-process modules.

- **FR3**: It is important for a track system to process wafers at the desired rate, which is called *throughput*. FR1 and FR2 must be performed while satisfying the system throughput rate. The throughput of the clustered stepper is the throughput requirement for a track system. A stepper is among the most expensive tools in FAB system, and its maximum utilization must be guaranteed. The track system should not be a throughput limiter of the track-stepper cluster for that reason.

[13] Stepper means 'step-and-repeat camera.'

Various process flows typically required by customers are:

- Standard(Simple) Coating/Developing flow
- TARC(Top Anti Reflective Coating)-included flow
- BARC(Bottom Anti Reflective Coating)-included flow
- Stand-alone Coating
- Stand-alone Developing

Throughput rate

- The range of throughput requirement varies significantly depending on process type and time. It varies typically from 50WPH to 100WPH.

- **DP3**: The system configuration is primarily responsible for meeting the process flow & throughput requirements. Given DP1 and DP2, the system must be configured such that the required flow can be processed at desired throughput rate. DP3 necessarily incorporates the physical layout and the transport issue, which are discussed in next level decomposition.

- **FR4**: Since the track system is a complex, *automated* machine, it has to control the functions within the system to make it operate smoothly.
- **DP4**: A command and Control Algorithm (CCA) coordinates the process tasks at this level. This includes all the system level activities such as administration, communication, etc. The CCA at this level is categorized to type I CCA (simply CCA I) by Hintersteiner and Tate[26].

Design matrix

$$\begin{bmatrix} FR1 \\ FR2 \\ FR3 \\ FR4 \end{bmatrix} = \begin{bmatrix} X & O & O & O \\ O & X & O & O \\ X & X & X & O \\ X & X & X & X \end{bmatrix} \begin{bmatrix} DP1 \\ DP2 \\ DP3 \\ DP4 \end{bmatrix} \qquad \textbf{(5.1)}$$

The DPs chosen at this level are abstract and conceptual, providing limited information. Some of the information available here from DPs is;

- The system will have modular structure.

- DP3, system configuration, covers system capacity and transport requirements.

- DP4, CCA, will have all the information for system operation.

With the above information, the design matrix needs to be determined. Determining the high-level design matrix has a large impact on the further decompositions because they must be consistent with the earlier design decisions.

(1, 1), (2, 2), (3, 3), (4, 4): The diagonal elements in design matrix are self-evidently 'X' because each DP has been chosen in such a way that they affect the corresponding FRs[14].

(1, 2), (2, 1): These elements are very clearly determined. DP1 has no effect on FR2 and vice versa, for the track system is based on modular structure, and these two sets of modules are completely separated functionally and physically.

The remaining elements in the design matrix indicate the direction for the later design process. The decisions regarding those elements need careful discussion.

(1, 3), (2, 3): These elements are evaluated by asking, "Does the choice of DP3 affect FR1 (FR2)?" The answer is "it should not." Since the two FRs, FR1 and FR2, are directly responsible for the on-wafer result, there should be no adverse effect onto those functions from any other sources within the system.

(3, 1), (3, 2): These elements are evaluated by asking, "Does the choice of DP1 (DP2) affect FR3?" As indicated in the FR/DP description section, FR3 includes process flow requirements and throughput requirement. By the nature of FR3, there is strong relationship between DP3 and DP1/DP2. For example, the accessibility to each module will affect DP3, consequently sub FR3.x. These off-diagonal non-zero elements can be viewed as a design decision that the interface between system configuration, DP3, and process modules, DP1/DP2, belongs to DP3 side. It is evident that it well be better if we can make theses elements 'O's. The conditions will be flexibility and standard interface.

(4, 1) – (4, 3): Command and Control Algorithm (CCA), which is responsible for system control, must be designed with the information regarding the objects to be controlled. The objects for this case are DP1, DP2, and DP3.

(1, 4) – (3, 4): The change of DP4 should not affect any one of FR1, FR2, and FR3. The implication of these zeros is that FR1, FR2, and FR3 must be immune to any change of system control DP.

[14] Hereafter, on-diagonal elements of every design matrix are assumed to be 'X,' and no discussion on the elements will be given.

The design matrix determined is a semi lower-triangular matrix, which corresponds to a decoupled design. It implies that for further design of DP3 and DP4, information should be transferred from other DP design. For example, design of DP3 requires some of the information from DP1 and DP2.

Constraints[15]

Once the design matrix is determined, the constraints need to be clarified. These constraints are the guideline to make the subsequent decomposition consistent with high-level design decision.

At this high-level, the constraints are stated straightforwardly, and they does not contain details.

- Decomposition of DP1 must not affect FR2
- Decomposition of DP2 must not affect FR1
- Decomposition of DP3 must not affect FR1, FR2
- Decomposition of DP4 should not affect FR1, FR2, and FR3.

More explicit statements will be available in later decompositions.

4.2.2.2 2nd Level

	Coat wafer with desired resist film	Coating process modules	
Process	Improve resist adhesion of a wafer surface	Vapor Prime module	
Process	Bring wafer temperature down to coat - process temperature	Chiller module	
Process	Coat wafer with resist	Spin coater	
Process	Cure the resist layer	Soft Bake module	
Process	Bring wafer temperature down to expose - process temperature	Chiller module	

Table 6. 2nd Level FR/DP - Coating process modules branch

Description

The inputs for this decomposition activity are as follows:

[15] The term 'constraint' used here means the condition to be satisfied in further decomposition. There are other constraints such as cost, time, personnel, etc.

The parent FR is FR1, 'Coat wafers with desired resist film', and the parent DP is DP1, 'coating process modules'. Constraints are all of the constraints imposed on the system and ones identified in the previous level design.

In order to coat a resist film onto a wafer surface, it needs a pre-coating thermal treatment and a post-coating thermal treatment in addition to the coating process itself.

Those FRs and corresponding DPs are stated more precisely.

- **FR1.1**: To promote the wafer surface adhesion to the photoresist material, the wafer surface should be free from any contamination and moisture. A clean silicon surface still does not provide good bonding sites for novolac-based photoresists. It must be chemically compatible with the resist. A couple of sub-steps are required to achieve sufficient adhesion, which will be discussed in the next level decomposition.

  Temperature range : 100 – 140°C

  Temperature control : ± 0.5°C

- **DP1.1**: Vapor Prime is conventionally used to achieve the wafer surface adhesion. Vapor Prime module is basically a heating module, which is a kind of small oven, and it first gets rid of the moisture on wafer surface. The module also performs surface modification task using chemical adhesion promoters. The most common adhesion promoter for silicon photoresist is Hexamethylenedisiloxane (HMDS), which is a volatile liquid at room temperature. Vapor coating – vapor priming – method is used for reproducible results.

- **FR1.2**: In order to be processed in subsequent steps, a heated wafer needs to cool to the preset temperature. It is one of the critical FRs because it determines the temperature distribution of a wafer before the coating process.

  Temperature range : 18 - 24°C

  Temperature control : ± 0.2°C

- **DP1.2**: Chiller module is primarily responsible for cooling the wafer temperature down. Tight temperature control is required to achieve FR1.2 satisfactorily. Design of chiller module – decomposition of DP1.2 – has to reflect the tight temperature control.

- **FR1.3**: Liquid photoresist material has to be deposited and coated onto wafer surface. Successful achievement of this FR is primarily responsible for the coating quality. It must produce uniform and repeatable resist film conforming to the specifications shown in FR1 description. As a sub-requirement, it should be able to carry various photoresists coating capability.

- **DP1.3**: Spin-coating has long been accepted as the best coating method for satisfying the FR. Spin coater module is responsible for the process of flooding the substrate with resist solution and drying the surface by spinning.

- **FR1.4**: After the spin-coating and drying, the polymer film contains up to 15% residual solvent and may contain built-in stresses due to the shear forces encountered during the spinning process. The wafer surface has to be thermally treated to get the final photoresist film on it. Since the dominant mechanism in this thermal process is solvent evaporation, whose rate is extremely temperature dependent, tight temperature control is required.

  Temperature range : 85 - 100°C

  Temperature control : ± 0.2°C

- **DP1.4**: Bake module – so called Soft Bake module – is primarily responsible for removing residual solvent and annealing any stress in the film. Design of Soft Bake module should reflect the tight temperature control.

- **FR1.5**: Resist thickness continues to change until the temperature of the wafer is chilled. The wafer temperature must be brought down to a setpoint.

  Temperature range : 18 - 24°C

  Temperature control : ± 0.2°C

- **DP1.5**: Chiller module is primarily responsible for cooling the wafer temperature down. FR1.5 is same as FR1.2, and DP1.5 will, practically, have the same feature as DP1.5.

Design matrix

$$\begin{bmatrix} FR1.1 \\ FR1.2 \\ FR1.3 \\ FR1.4 \\ FR1.5 \end{bmatrix} = \begin{bmatrix} X & O & O & O & O \\ O & X & O & O & O \\ O & O & X & O & O \\ O & O & O & X & O \\ O & O & O & O & X \end{bmatrix} \begin{bmatrix} DP1.1 \\ DP1.2 \\ DP1.3 \\ DP1.4 \\ DP1.5 \end{bmatrix} \quad \textbf{(5.2)}$$

As discussed in Level 1, we must determine the design matrix for the given information only at this level.

- DP1.1, DP1.2, DP1.4, and DP1.5 are the modules of thermal processes, heating and cooling.

- Each DP will be separated physically.

- There is no direct interaction among DPs.

Based on the above information, the evaluating questions can be answered.

(2, 1) – (5, 1), (1, 2) – (1, 5), (3, 2) – (5, 2), (2, 3) – (2, 5), (4, 3), (5, 3), (3, 4), (3, 5), (5, 4), (4, 5): All of these off-diagonal terms are zero. The design of DP1.1 – DP1.5 is modular, and no interface is required between them.

The resulting design matrix is a diagonal design matrix, which represents the best design at this level. This design decision implies that each DP must satisfy its corresponding FR exactly. Otherwise, the design matrix will be lower triangular matrix because of the process sequence: wafer is processed in DP1.1 first, then sent to DP1.2 and processed in DP1.3, etc. Any failure of one of these FRs will propagate along the process.

Constraints

In order to ensure the design matrix is diagonal, there are some constraints for off-diagonal 'O' elements.

(2, 1) – (5, 1): DP1.2 – DP1.5 have to be protected from any thermal effects caused by DP1.1

(1, 4) – (3, 4), (5, 4): DP1.1 – DP1.3, and DP1.5 have to be protected from any thermal effects caused by DP1.4

(1, 3) – (2, 3), (4, 3) – (5, 3): Neither mechanical vibration nor particle generation from DP1.2 must be transferred to DP1.1, DP1.2, DP1.3, and DP1.4.

Since DP1.1, DP1.2, DP1.3, DP1.4 and DP1.5 are physically separate modules, there are no other constraint across them.

	Develop exposed film	Developing process modules	
Process	activate photoacid created by the exposure process	Post Exposure Bake (PEB) module	
Process	Bring wafer temperature to coat temperature	Chiller module	
Process	Develop the exposed film	Spin developer	
Process	Dehydrate the wafer to harden the film	Hard Bake module	
Process	Bring wafer temperature to a set point	Chiller module	

Table 7. 2nd Level FR/DP - Developing process modules branch

Description

In order to produce final 3D image, we need a series of develop processes.

- **FR2.1**[16]: For DUV process, exposure process creates a photoacid which acts as a catalyst to mediate a cascade of reactions. The photoacid is used to generate latent image, and heating process is required for the reaction. With the photoacid as a catalyst, a chain reaction is initiated such that it modifies the resist chemistry to allow latent image generation.

  Temperature range : 90 - 110°C

  Temperature control : ± 0.2°C

- **DP2.1**: Post Exposure Bake (PEB) module is responsible for the FR2.1, latent image generation. Since chemical reactions after the photoacid generation are all thermally activated, temperature/temperature uniformity control is important. It is also important to control the extent of the chemical reactions by controlling process time.

- **FR2.2**: The baked wafer needs to be chilled until its temperature reaches a set point. Until the chill process starts, the chain reaction for DUV resist continues. FR2.2 is basically same as FR1.5.

[16] For I-Line system, FR2.1 mainly is the requirement for eliminating standing waves in the exposed area. Because of this difference, distinct constraints will be imposed on it. Discussion above is dedicated to DUV process.

- **DP2.2**: PEB chill module is primarily responsible for bringing the wafer temperature down. DP2.2 will have the same feature as DP1.5.

- **FR2.3**: It states the need for producing the final 3D image of the replicated mask pattern from the latent image. It is known to be the most complex of the processing steps, and has the greatest influence on pattern quality. Typical criteria for a developing FR are;

  Uniform CD control: stated in FR2 specification

  Minimum Defect count : 0.02 defects/cm^2, ≥0.2 μm size

  Minimum contamination and minimum reduction in film thickness

- **DP2.3**: Spray type development process module is responsible for achieving FR2.3. This module basically sprays fresh developer, and dries the substrate of all residual developers and rinses after the completion of the developing cycle.

- **FR2.4**: FR2.4 states the requirement to increase the etch resistance of the developed resist film. A wafer needs to be dehydrated to improve resist adhesion as well as residual solvent is evaporated to harden the film. Fundamental considerations for FR2.4 are the same as those discussed in FR1.4, Soft Bake FR, except that it has less limitation, and thus does not require strict control.

  Temperature range : 90 - 110°C

- **DP2.4**: Hard Bake module is primarily responsible for satisfying FR2.4, and it is essentially the same module as DP1.4, Soft Bake module.

- **FR2.5**: The wafer temperature must be brought down to a set point. This is a non-critical process requirement as compared to other chilling FRs.

- **DP2.5**: Chiller module is primarily responsible for cooling the wafer temperature down. DP2.5 is essentially the same as DP1.5.

Design matrix

$$\begin{bmatrix} FR2.1 \\ FR2.2 \\ FR2.3 \\ FR2.4 \\ FR2.5 \end{bmatrix} = \begin{bmatrix} X & O & O & O & O \\ O & X & O & O & O \\ O & O & X & O & O \\ O & O & O & X & O \\ O & O & O & O & X \end{bmatrix} \begin{bmatrix} DP2.1 \\ DP2.2 \\ DP2.3 \\ DP2.4 \\ DP2.5 \end{bmatrix} \quad (5.3)$$

The information given in FR/DP statements is:

- DP2.1, DP2.2, DP2.4, and DP2.5 are the modules of thermal processes, heating and cooling.

- Each DP will be separated physically.

- There is no direct interaction among DPs.

Based on the above information, the evaluating questions can be answered.

(2, 1) – (5, 1), (1, 2) – (1, 5), (3, 2) – (5, 2), (2, 3) – (2, 5), (4, 3), (5, 3), (3, 4), (3, 5), (5, 4), (4, 5): All of these off-diagonal terms are zero. The design of DP2.1 – DP2.5 is modular, and no interface is required between them.

The resulting design matrix is a diagonal design matrix, which represents the best design at this level. Due to the similar nature of modular design, same argument used in decomposition of coating process modules is applied at this level.

Constraints

The three constraints for coating process modules will be applied to the developing process modules equivalently.

(2, 1) – (5, 1): DP2.2 – DP2.5 have to be protected from any thermal effects caused by DP2.1

(1, 4) – (3, 4), (5, 4): DP2.1 – DP2.3, and DP2.5 have to be protected from any thermal effects caused by DP2.4

(1, 3) – (2, 3), (4, 3) – (5, 3): Neither mechanical vibration nor particle generation from DP2.2 must be transferred to DP2.1, DP2.2, DP2.3, and DP2.4.

Since DP2.1, DP2.2, DP2.3, DP2.4 and DP2.5 are physically separate modules, there are no other constraint across them.

DUV process will impose some constraints on DP3, system configuration: the wafer has to be moved from the stepper side to PEB module with no delay. This constraint will be discussed in consistency checking section.

	Process wafers at desired rate	System configuration	
Support	Support the system physically	System layout	
Process (Transport)	Move wafer when process is finished	Transport robot system	

Table 8. 2nd Level FR/DP - system configuration branch

Description

In order to satisfy the process flows and throughput requirements, the system has to be designed for two aspects: capacity and transport. Those two aspects are captured at this level FRs.

- **FR3.1**: As the decomposition for FR1/DP1 and FR2/DP2 shows, the track system is composed of number of process modules. Sufficient capacity to meet the flow and throughput requirements must be ensured, and the modules need to be physically supported (allocated within a frame).
- **DP3.1**: System layout includes capacity planning and module allocation. In the next level decomposition, those two issues are discussed in detail

- **FR3.2**: Due to the fact that a wafer has to go through a series of different process modules, wafer transport is one of the essential FRs. Basically, FR3.2 states that every wafer must be transported when its specified process is.
- **DP3.2**: Transport FR is achieved by transport robot system. As the throughput requirement becomes higher, robot(s) is(are), currently, the most effective way of transport. Number (or kinds) of robots is determined with more information which is obtained from further decomposition of FR1.#/DP1.#, FR2.#/DP2.# and FR3.1/DP3.1.

Design Matrix

$$\begin{bmatrix} FR3.1 \\ FR3.2 \end{bmatrix} = \begin{bmatrix} X & \mathbf{X} \\ X & X \end{bmatrix} \begin{bmatrix} DP3.1 \\ DP3.2 \end{bmatrix} \qquad \textbf{(5.4)}$$

(2, 1): The design of DP3.2, or sub DP3.2, must be, definitely, in accordance with the choice of DP3.1, system layout. For example, the layout drives the requirements for robot system such as moving range of certain joint in robot, low bound of robot speed, etc. 'X' of (2, 1) indicates the transfer of that type of information.

(1, 2): Preferably, DP3.2, Transport robot system, should not affect FR3.1. However, in the system analyzed in this case study, it turns out that DP3.2 has effect on FR3.1, and thus neither DP3.1 nor DP3.2 can be chosen – designed – without iteration. This coupling is clarified in next level decomposition presented in section 4.2.2.4. In fact, the manufacturer of this track system had to spend a lot of time to find out the optimum operating point for this coupled design, and ended up with unsatisfactory solution for both internal and external customers. More discussion regarding this coupled design and proposed solution is presented in reference [27].

Constraints

There is no specific constraint within the design matrix of this level. However, more important constraints for both DPs are imposed by other branches such as FR2x/DP2x. They are discussed at the end of 2nd level decomposition.

CCA	Administer system	System administrator	
CCA	Command/control system operation	Operation controller	
CCA	Build internal communication	Communication manager	

Table 9. 2nd Level FR/DP - CCA branch

Description

As mentioned in FR4 description, the parent FR, 'Operate the system functions', is derived from the fact that the track machine is a complex automated machine. Parent DP, 'system control command', will have access to all the information required to control the system such as process recipe, status of modules, error, etc. Based on the parent FR/DP, 3 sub-FRs/DPs are identified. At this level, they are defined in general terms simply to clarify main features of system CCA in structured way.

- FR4.1: It states the requirement of administrating the system. Detailed FRs are presented in next level decomposition.

- DP4.1: System administrator is responsible for the various tasks regarding administration, which are clarified as sub-FR4.1. A group of software modules constitute DP4.1.

- FR4.2: It states the requirement of generating actual command to control system operation.
- DP4.2: Operational controller is primarily responsible for generating control commands. A group of software modules constitute DP4.2.

- FR4.3: All the information, essentially electric signal, needs to be transferred accurately. There is a strong requirement for robust communication.
- DP4.3: Communication manager is responsible for building robust internal communication. It comprises communication protocol, data transfer management, etc.

FR41 concerns the process recipe to operate the system. The recipe is defined by the user and contains such information as process flow, process time, setting temperature, spin speed, priority[17], type of chemicals, etc. The system must be designed to have flexibility enough to support this variation. (➔ this is represented by the 4th column 'X's in the 1st level design matrix.)

In order to control the system, we need to know the states of wafers and modules in real time. FR42 states this requirement and DP42 will be constructed to perform this function through further decomposition.

FR43 is the key concept of DP4, 'system control command' for system. Based on the recipe, process flow knowledge, and design information which will be wholly captured in this document will be resources of DP43, 'command module.' It will do all the tasks for system operation; sequencing operations, determining operating parameters, etc.

Design matrix

$$\begin{bmatrix} FR4.1 \\ FR4.2 \\ FR4.3 \end{bmatrix} = \begin{bmatrix} X & O & O \\ X & X & O \\ X & X & X \end{bmatrix} \begin{bmatrix} DP4.1 \\ DP4.2 \\ DP4.3 \end{bmatrix} \qquad \textbf{(5.5)}$$

Each of the DPs is more logical than physical, therefore design matrix will be determined based mainly on the information flow.

(2, 1): DP4.1, system administrator, has basic information necessary for the operation, and distributes it to other portion of CCA. DP4.1 also has the authority of high priority interrupt.

(3, 1) – (3, 2): The information to be transferred from DP4.1 to DP4.2 and vice versa affects FR4.3, and consequently design of DP4.3.

Design matrix determined is decoupled design matrix. Further decomposition reveals specific contents of the off-diagonal terms, and those off-diagonal terms are formulated in the form of software modules or data transfer.

Constraints

As mentioned above, the decomposition for FR4/DP4 mainly focuses on information flow. There are no physical constraints as in the other decompositions. The only constraint derived from this design matrix is to follow the information flow sequence in design.

4.2.2.3 Check Design Decision consistency (Level 1 & Level 2)

Since the two levels of FRs/DPs and every design matrix are determined, the consistency of the design decisions is to be checked. For example, the first level design matrix has zero element at (2, 1). Does any one of DP1.# have strong effect on FR2.#? If that is the case, either proper constraints need be imposed or the first level design matrix needs to be reevaluated. This activity is discussed in section 2.2.1.

Table 10 shows the consistency-checking matrix for the 1st and 2nd level design. The particular example given below is related to the first two levels, where the concrete information is not available yet. Therefore, an effort is made to impose the proper constraints in order to maintain the higher-level design decisions. The higher-level design matrix is a 4×4 semi-lower triangular matrix. The shaded cells indicate 'X' elements in the first level design matrix. The elements with superscript indicate that certain constraint is specified to ensure the design decision, and the constraint is described.

Condition

1 Thermal effect from the modules of bake processes to other modules in the system must be prevented - the system must either use thermal shields or have appropriate layout.

2 Spin module must not affect each other in terms of vibration, particle generation, etc.

---

[17] Priority is needed because of the transportation problem, i.e. if we let one robot do just one transfer task, then we don't need priority.

		DP1 #					DP2 #					DP3 #		DP4 #		
		1	2	3	4	5	1	2	3	4	5	1	2	1	2	3
FR1 #	1	X	0	0	0	0	0^1	0	0	0^1	0	0^8	0	0	0	0
	2	0	X	0	0	0	0^1	0	0	0^1	0	0^8	0	0	0	0
	3	0	0	X	0	0	0^1	0	0^2	0^1	0	0^8	0^4	0	0	0
	4	0	0	0	X	0	0^1	0	0	0^1	0	0^8	0^5	0	0	0
	5	0	0	0	0	X	0^1	0	0	0^1	0	0^8	0	0	0	0
FR2 #	1	0^1	0	0	0^1	0	X	0	0	0	0	0^8	0^6	0	0	0
	2	0^1	0	0	0^1	0	0	X	0	0	0	0^8	0	0	0	0
	3	0^1	0	0^2	0^1	0	0	0	X	0	0	0^8	0^7	0	0	0
	4	0^1	0	0	0^1	0	0	0	0	X	0	0^8	0	0	0	0
	5	0^1	0	0	0^1	0	0	0	0	0	X	0^8	0	0	0	0
FR3 #	1	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0
	2	X^3	X^3	X^3	X^3	X^3	X^3	X^3	X^3	X^3	X^3	X	X	0	0	0
FR4 #	1	X	X	X	X	X	X	X	X	X	X	0	X	X	0	0
	2	X	X	X	X	X	X	X	X	X	X	0	X	X	X	0
	3	X	X	X	X	X	X	X	X	X	X	0	0	X	X	X

Table 10. Consistency checking matrix for 1st&2nd level

3 Standard mechanism for wafer hand-off at the different process modules will minimize the effect of these elements.

4 Having no delay in wafer transport from a coating module to a Soft Bake module is preferred. If the delay is inevitable, it should be repeatable for consistent on-wafer result.

5 Having no delay in wafer transport from a Soft Bake module to a chiller module is strongly preferred. The delay, if any, should be minimized and repeatable.

6 Having no delay in wafer transport from a PEB module to a chiller module is required. It is one of the most important requirements because it affects the critical chemical reaction within the photoresist film.

7 If unavoidable, repeatable transport delay after developing process is desirable.

8 The design of system layout must be done in such a way that the cross effects among DP1.# and DP2.# are minimized.

The constraints listed above are derived from examining the design matrix of Table 10. Those constraints are considered in subsequent decomposition process for each branch of hierarchy.

4.2.2.4 3rd Level

The 3rd level decomposition is done for each of 15 branches, and involves fair amount of details. In this section, a few of the branches are selected for illustration purpose. As an example of process module design, decomposition of FR1.3/DP1.3, spin coater, is shown. Decomposition of FR3.1/DP3.1, system layout, is presented on behalf of system configuration design. Finally, FR4.2/DP4.2, operation controller, is decomposed to demonstrate system software design.

	Coat wafer with photoresist	Spin coater	
Process	Send/receive wafer from transport device	Wafer I/O mechanism, which changes	
Process	Deposit resist onto wafer surface	Resist delivery/deposit system	
Process	Spin wafer with desired velocity profile	Spin system	
Process	Clean wafer edge with solvent material	Solvent delivery/deposit system for top side & bottom side	
Process	Dispose of wasted material	Material disposal system	
Process	Control local environment	Environmental factor controller	
Control	Control coating process functions	Local process controller (CCA II)	

Table 11. 3rd Level FR/DP - Spin coater branch

Description

Parent FR is 'Coat wafer with photorsist', and parent DP is 'Spin coater.' Based on the parent FR/DP, 7 sub-FRs/DPs are identified. The FRs captures key functions that a spin coater has to perform, and consequently, DPs identify the essential features of spin coater.

- **FR1.3.1**: Spin coating process starts with accepting a wafer from outside, which is in fact certain type of transport robot of DP3.2.#, and ends with sending the wafer out. It should provide the robot with as easy access as possible.

- **DP1.3.1**: Wafer I/O mechanism is responsible for receiving/sending a wafer from/to transporter. What it does is, basically, move a wafer between process position and In/Out position. Alternatively, it could be a mechanism that makes the spin coater to be accessible by the robot without changing the wafer position. In our design, the former concept is selected and designed in detail with further decomposition.

- **FR1.3.2**: Once a wafer is in process position, resist solution is to be flooded over the substrate. The initial resist application must be accomplished with precise position control in order to obtain a liquid film of uniform thickness. The resist material should be under tight temperature control before/during application because resist temperature has huge effect on uniformity and a significant effect on the mean thickness.

- **DP1.3.2**: Resist delivery/deposit system is primarily responsible for FR1.3.2. It is in charge of the delivery of the resist solution from reservoir to deposit device and of resist application over the substrate. In order to satisfy the FR, further decomposition should focus on precise position control, and resist solution temperature control.

- **FR1.3.3**: With the resist solution applied to the substrate, a wafer has to spin to flood the solution over the entire surface uniformly and subsequently to dry it as a constant rate to yield a uniform, solid polymeric film. The initial coating can be accomplished either by (1) performing FR1.3.2 first prior to spinning (static dispense) or (2) performing FR1.3.2 and FR1.3.3 simultaneously (dynamic dispense). For both cases, spinning speed profile should be precisely determined. It is known that spinning speed has strong dominant effect on film thickness and significant effect on the uniformity of the coated film.

  Spinning speed range : 0 – 7,500 rpm

  Spinning speed control : ±1 rpm in the range of 10 - 5,000 rpm

- **DP1.3.3**: Spin system must satisfy high accuracy speed requirement. It should be noted that from the constraints stated at the second level decomposition, minimum vibration is required for the desigr of DP1.3.3.

- **FR1.3.4**[18]: During the spinning process, excess resist is deposited on the edges and backside of the wafer, which can become a source of particle contamination in subsequent process steps. The excess resist needs to be removed for better quality image.

- **DP1.3.4**: Bead removal technique is required to achieve FR1.3.4, and this is done by applying solvent to the edges in precisely controlled manners. DP1.3.4 is basically similar to DP1.3.2: it delivers solvent instead of resist solution, and deposit solvent onto unwanted polymer.

- **FR1.3.5**: This is 'non-value added' FR, but still indispensable. The spin coater needs to handle the splashed material such as resist solution and solvent.

- **DP1.3.5**: Material disposal system is responsible for collecting the splashed material and disposing it out of the system. Design of DP1.3.5 – sub-DP of DP1.3.5 – must guarantee the minimum splashback of the splashed material. Other consideration should be given, which are discussed in constraints section.

- **FR1.3.6**: Since the resist coating is dominated by evaporation, and sticky liquid state of resist solution is initially applied, strict environmental control is required. The factors to be controlled are temperature, humidity, airflow, and cleanliness.

  Ambient air temperature range : 18 – 30°C

  Ambient air temperature control : ± 0.1°C

  Air humidity : 35 – 45 %

  Air humidity control : ± 0.5 %

  Environment : class 0.1

- **DP1.3.6**: Environmental factor controller is primarily responsible for controlling temperature, humidity, airflow, and particle contamination.

[18] If imaging is needed on the outer edges, removal of the bead is not acceptable.

- **FR1.3.7**: Each of the coating process functions should be coordinated and controlled. The requirement involves sequencing/synchronizing the functions and setting local process parameters.
- **DP1.3.7**: Local process controller, category of CCA type II, is responsible of coordinating FR1.3.# and their sub FRs. It should be in conjunction with top level controller, DP4 and sub DP4, and also with lower level controller, which is included in decomposition of FR1.3.#/DP1.3.#.

Design matrix

$$\begin{bmatrix} FR1.3.1 \\ FR1.3.2 \\ FR1.3.3 \\ FR1.3.4 \\ FR1.3.5 \\ FR1.3.6 \\ FR1.3.7 \end{bmatrix} = \begin{bmatrix} X & O & O & O & O & O & O \\ O & X & O & O & O & O & O \\ O & O & X & O & O & O & O \\ O & O & O & X & O & O & O \\ O & O & O & O & X & O & O \\ O & X & X & X & X & X & O \\ X & X & X & X & O & O & X \end{bmatrix} \begin{bmatrix} DP1.3.1 \\ DP1.3.2 \\ DP1.3.3 \\ DP1.3.4 \\ DP1.3.5 \\ DP1.3.6 \\ DP1.3.7 \end{bmatrix} \quad \textbf{(5.6)}$$

The information given at this level is summarized below.

- DP1.3.2, resist delivery/deposit system, and DP1.3.4, solvent delivery/deposit system, are placed over a wafer during its deposit process.
- DP1.3.3, spin system, generates certain amount of heat and particles.
- DP1.3.5, material disposal system, has to enclose the main body of spin coater to block splashing resist solvent.
- DP1.3.6, environmental factor controller, is primarily in charge of temperature, humidity, airflow, and particle contamination control throughout the process.

(6, 2): Resist delivery/deposit system, specifically deposit device, has effect on airflow control when it is located on top of wafer surface.

(6, 3): Spin system generates certain amount of heat while spinning, which affects temperature control. Particle generation during its spinning process affects particle contamination control

(6, 4): Solvent delivery /deposit system, specifically deposit device, has effect on airflow control when it is located on top of wafer surface. The effect of DP1.3.4, however, is smaller than that of DP1.3.2, because DP1.3.4 operates after the resist coating process is done. Airflow at that point may not be as significant as that in the midst of resist coating.

(6, 5): The enclosure of material disposal system around spinning system affects the pattern of airflow. Recirculating flow may also cause particle contamination problem.

(7, 1) – (7, 4): Local process controller is designed to control the operation of each DP. Change of any of those DPs – or sub DPs – potentially requires redesign of certain feature of DP1.3.7.

Design matrix is determined to be a decoupled design matrix. Foregoing decomposition needs to follow the sequence indicated in the design matrix. It should be noted that coating process is highly sensitive to temperature and humidity of the environment and the temperature of the resist solution. All of the sources to affect them need to be minimized. In terms of design matrix, the off-diagonal 'X's in the 6th row, (6, 2) – (6, 5), must be minimized.

Constraints

At the 2nd level design, the constraint related to coater module is;

Neither mechanical vibration nor particle generation from DP2.2 must be transferred to DP2.1, DP2.2, DP2.3, and DP2.4.

The above constraint is not specifically addressed in 3rd level design, and needs to be considered for next level decomposition.

(3, 1): Wafer I/O mechanism should not affect the performance of spinning function.

As mentioned in design matrix discussion, these elements of (6, 2) – (6, 5) must be minimized to obtain required quality of photoresist film.

(6, 2): Resist solution dispense device should have minimum effect on airflow over the wafer surface. Constraint can be imposed on the shape of dispense device and size of the device.

(6, 3): Heat generation in spin system, either from actuator or from friction, must be minimized. Or the heat should not be transferred to wafer processing area. The spin system design should prevent any particle from coming out to the environment.

(6, 4): Solvent dispense device should have minimum effect on airflow over the wafer surface. Constraint can be imposed on the shape of dispense device and size of the device.

(6, 5): The enclosure of the disposal system should be designed such that it generates minimum recirculation of airflow.

	Support the system physically	System layout	
Support	Keep TAKTprocess,i below TAKTsys	# of each process module	
Support	Maintain # of movements by main robot not to degrade target throughput	# of IBTA	
Support	Locate process modules into 200-APS frame	Layout(module arrangement)	

Table 12. 3rd Level FR/DP - System layout branch

Description

- FR3.1.1: We define TAKT time as the time interval (in seconds) between two consecutive wafers coming out of the system. Thus, the TAKT time of the system is directly driven from required throughput.

$$\text{TAKT}_{\text{system}}(\text{sec}) = \frac{1}{\textit{Throughput}(\text{WPH})} \times 3600 \qquad (5.7)$$

For example, if the required throughput is 60WPH, then TAKT time of the system is 60 sec.

In order to meet the throughput requirement, the TAKT time for every process step must be equal to or less than system TAKT time.

- DP3.1.1: TAKT time for each process is computed for given process time, number of modules, and transport time. It is termed as a Fundamental Period (FP)[19]. The concept of FP is presented Perkinson, et al.[28], and further discussed by Oh and Lee [29]. For process i, FP_i is defined as;

$$FP_i = \frac{PT_i + 4T}{N_i} \qquad (5.8)$$

,where PT is process time, N is number of the process modules, and T is transport time. T is more precisely defined as the sum of translation time and the time for wafer handling before being ready to move to another module.

[19] Concept of fundamental period is presented in reference

The condition to satisfy system throughput is,

$$FP_i \leq TAKT_{system} \tag{5.9}$$

Substituting equation (5.8) into (5.9),

$$N_i \geq \frac{PT_i + 4T}{TAKT_{system}} \tag{5.10}$$

Therefore, N, number of module, is selected as DP3.1.1 to satisfy FR3.1.1.
However, as noticed in equation (5.10), it involves transport time, which is determined from DP3.2.#. This is the source of the coupling indicated in the 2nd level decomposition for system configuration in section 4.2.2.2.

- FR3.1.2: Transport module is simply a process module with certain process time, from the perspective of system throughput. Yet, TAKT time for a transport module is slightly different from that of process module.

  Within a period of $TAKT_{system}$, a transport module must complete all required wafer moves, TAKT time of a transport module is defined as the sum of transport time for each moves. For example, suppose that there are 2 process steps. Then, total of 3 moves must be performed in one cycle, which is $TAKT_{system}$. TAKT time of a transport module is $\tau_1+\tau_2+\tau_3$, or for simplicity, $3\times\tau$. If $TAKT_{system}$ is 30 seconds, necessary condition[20] to meet the throughput requirement is $\tau \leq 10$sec.

  Now that the transport time is mostly limited by transport module design – although the time is also function of location, which is DP3.1.3 –, there are some instances when $TAKT_{transport}$ exceeds $TAKT_{system}$. In that case, $TAKT_{transport}$ must be reduced until it gets lower than system TAKT time.

- DP3.1.2: The definition of $TAKT_{transport}$ is;

  $$TAKT_{transport} = N \times \tau \tag{5.11}$$

  , where N is number of moves, and τ is 2T.

  In order to reduce $TAKT_{transport}$, we have to either 1) decrease τ, or 2) decrease N. Option 1) is related to DP3.1.2 performance improvement, and is not easily achieved. Option 2) is to

[20] Note that this is necessary condition. Additional conditions are needed to meet the throughput requirement, and those are discussed in reference [29].

decrease number of moves that one (main) transport module has to perform by assigning small and cheap transporter for some of the moves. It should be noted that equation (5.11) involves transport time, and both of the options are dependent on the transport time: another source of the coupling indicated in 2nd level decomposition.

In the company's design, they've selected option 2) due to some reasons such as cost of main transport device, layout design (DP3.1.3), etc. They added a number of subsidiary transport devices to the system. The device is named as IBTA (Inter Bay Transfer Arm). IBTA is a relatively simple transport device, which handles wafer moves between two consecutive thermal process modules, e.g. SB → SBC. Yet, it turns out that introduction of IBTA causes another coupling resulting inflexibility in system configuration.

- FR3.1.3: A number of process modules (DP3.1.1) need to be allocated into proper geometric position within the system.
- DP3.1.3: The system analyzed adopts 'bay-and-spin station' frame structure. It is divided to two portions; thermal stack on one side and spin station on the other. Previous level constraints, minimization of thermal cross-talk, must be considered in design of the frame. The modules arrangement in the frame is also considered.

Design matrix

$$\begin{bmatrix} FR3.1.1 \\ FR3.1.2 \\ FR3.1.3 \end{bmatrix} = \begin{bmatrix} X & O & O \\ O & X & \mathbf{X} \\ X & X & X \end{bmatrix} \begin{bmatrix} DP3.1.1 \\ DP3.1.2 \\ DP3.1.3 \end{bmatrix} \quad \textbf{(5.12)}$$

(2, 1), (1, 2): The number of modules is dependent only on the throughput requirement and process time (plus transport time), and the number of moves a transport device must make is dependent on process flow. In mathematical expression,

$$(2,1) = \frac{\partial(\mathrm{FR3.1.2})}{\partial(DP3.1.1)} = \frac{\partial(N\tau)}{\partial \mathrm{N}} = 0 \quad \textbf{(5.13)}$$

$$(1,2) = \frac{\partial(\mathrm{FR3.1.1})}{\partial(DP3.1.2)} = \frac{\partial\{(\mathrm{PT_i} + 4\mathrm{T})/\mathrm{N_i}\}}{\partial(\mathrm{Number\ of\ IBTA})} = 0 \quad \textbf{(5.14)}$$

(3, 1): This element indicates the information of how many modules are in the system. DP3.1.1 is the direct object which FR3.1.2 addresses.

(3, 2): DP3.1.2, IBTA, is a simple device performing transport between two thermal modules. Because of the limited capability, two consecutive modules must be located side by side once IBTA is selected as a means of wafer transfer. The limitation affects FR3.1.3 and decreases the freedom in allocating the modules in the frame.

(2, 3): Since the space for juxtaposition of consecutive thermal modules is limited in 'bay-and-spin station' structure, determining the number of IBTA is not free from DP3.1.3.

The design matrix is coupled at this level. It turns out that the coupling results from the adoption of IBTA, and inflexible nature of bay-and-spin station structure. Note that the previous level design matrix is also coupled one, and causes of the coupling are mentioned in the description of FR/DP above.

	Command/control system operation	Operation controller	
CCA	Initialize the system	Ini.-mode algorithm	
CCA	Acquire operational data	Read/Refer to the recipe database	
CCA	Compile module-level information	Process data collector	
CCA	Command the 'action'	Event handler/Scheduler	
CCA	React to malfunction	Error handler	

Table 13. 3rd Level FR/DP - Operational controller branch

Description

Operation of the system involves largely 3 activities: Initialization, Routine operation, and Error handling. FRs identified here address those 3 activities.

- **FR4.2.1**: Before starting the operation, the system must be initialized. It include both hardware and software initialization.
- **DP4.2.1**: Ini.-mode algorithm is responsible for coordinating the hardware/software during startup procedures.
- **FR4.2.2**: In order to perform the operation, the operation controller must have data regarding all the operations.

- **DP4.2.2**: Software module of reading/referring is responsible for acquiring the data from recipe database (DP4.1.1.1 & DP4.1.1.2).

- **FR4.2.3**: All of the necessary information, which is related to each process module states, needs to be known to the operation controller.
- **DP4.2.3**: This software module is responsible for collecting the information.

- **FR4.2.4**: Various commands for the operation are to be generated.
- **DP4.2.4**: These software modules (algorithm) are responsible for generating system commands.

- **FR4.2.5**: When there is malfunction within the system, the operation controller has to react to the malfunction.
- **DP4.2.5**: Error handler is primarily responsible for coordinating the system during malfunctioning.

Design matrix

$$\begin{bmatrix} FR4.2.1 \\ FR4.2.2 \\ FR4.2.3 \\ FR4.2.4 \\ FR4.2.5 \end{bmatrix} = \begin{bmatrix} X & O & O & O & O \\ O & X & O & O & O \\ O & O & X & O & O \\ O & X & X & X & X \\ O & O & X & O & X \end{bmatrix} \begin{bmatrix} DP4.2.1 \\ DP4.2.2 \\ DP4.2.3 \\ DP4.2.4 \\ DP4.2.5 \end{bmatrix} \quad \textbf{(5.15)}$$

(4, 2): DP4.2.2 acquires recipe data and transfer it to DP4.2.4 such that it can generate operation command based on the data.

(4, 3): Process data are transferred to DP4.2.4, Event handler.

(4, 5): Event handler should acquire error information from DP4.2.5, Error handler, and deal with error situation.

(5, 3): Process data are transferred to DP4.2.5, Error handler.

4.2.3 *Graphical represent of System Architecture*

Part of the SA for the track system is represented using a Flow chart. Figure 16 shows the SA with the three levels decompositions. The flow chart provides visual information in its simple format. Without re-examining all the design matrices, the structure of the design decisions is visualized.

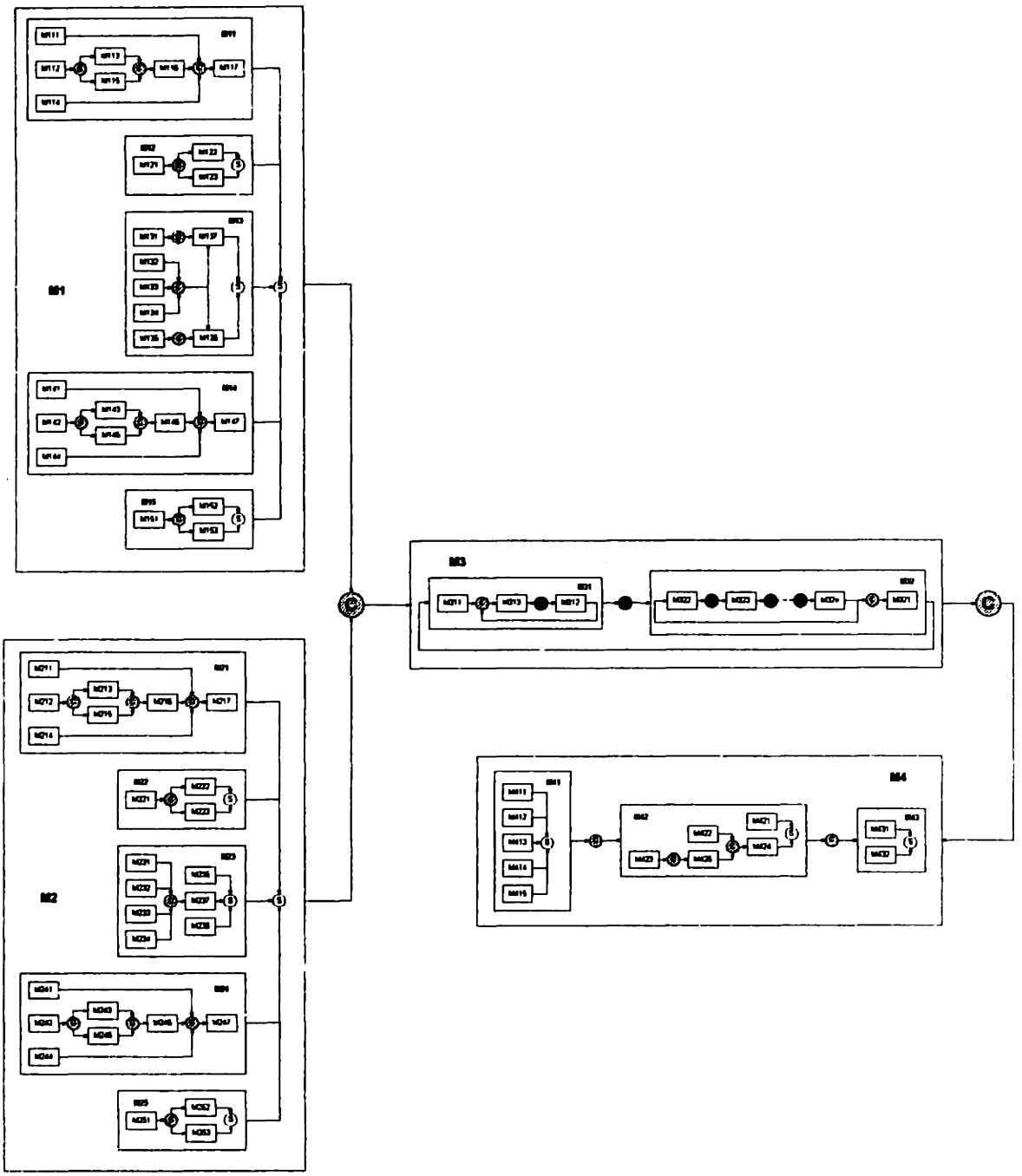


Figure 16. Flow chart for track system (3rd level)

4.3 Examples and related hypothesis

Section 4.2 showed part of the decomposition for the track system. It shows how AD design activities are applied to an actual system design and analysis, and illustrates some of the concepts presented in the hypothesis: system representation, top-down approach for system design, mapping/zigzagging at high-level and low-level, capturing design knowledge, etc.

This section presents some of the examples that focus on specific problems encountered during the case-study. These examples address the following two hypotheses, and validate the hypotheses:

- 4.3.1 - SA is useful as a diagnostic tool for a system.
- 4.3.2 - A system is designed using top-down approach based on the fundamental concept of AD. (Small scale example)

4.3.1 System Diagnosis with SA – Diagnosis for wafer off-centering

During its testing period, the track system encountered various problems to be fixed immediately. The first step to solve a problem is to figure out cause(s) of the problem. One of the problems, 'wafer off-centering at VP module' is investigated using the SA to demonstrate its capability as a diagnostic tool.

4.3.1.1 Description of the problem

Once a wafer is processed in a Vapor Prime module, it is to be picked up and transported to the next process module, a Chiller module. Ideally, a wafer sits on the exact center of a Vapor Prime process position, and a transport robot picks it up assuming it is in the right position. However, it is observed that the center of the transport robot end-effector does not coincide with the center of the wafer in the Vapor Prime module.

The problem occurs only in wafer pick-up of Vapor Prime module, which indicates that the problem is caused most likely by abnormal functioning of Vapor Prime module.

4.3.1.2 Diagnosis procedure

In order to diagnose a problem, the 4-step diagnosis procedure using SA is developed. Since the procedure is entirely based on the SA created for the system, a comprehensive and accurate SA is imperative for the diagnosis.

STEP 1: LOCATE PROBLEM INTO SA

The first essence of AD approach is to identify the functions a system has to perform. The complete set of FRs specifies every function in hierarchical structure. Once the functions are stated explicitly, it is possible to see a problem as failure of a specific function. The problem is located in the SA: for example, "FR#.#.#.# is not properly achieved".

In case of the 'off-centering' problem, the FR/DP for the Vapor Prime module is explored. Table 14 shows the decomposition for the Vapor Prime module branch.

	Improve resist adhesion to wafer surface	*Vapor Prime module*	
Process	Receive wafer	3 actuated lift pins for robot access-'Receive' mode	
Process	Isolate heating space from exterior	Sealed/insulated lid-type chamber	
Process	Supply selected gas	Selective, gas supplying module	
Process	Heat wafer from 20C to 150C with +/- 0.25C uniformity	Casted resistive hot plate	
Process	Exhaust inside gas	Exhaust plumbing system	
Process	Prepare wafer for hand-off	3 actuated lift pins for robot access-'Send' mode	
Control	Control Vapor Prime process functions	Local process controller (CCA II)	

Table 14 . FRs/DPs for Vapor Prime module branch

The problem description says that the center of transport robot end-effector does not match with the center of a wafer at the time of picking-up. The problem is clearly the failure of FR1.1.6, 'prepare wafer for hand-off.'

STEP 2: CHECK EACH 'X' IN DESIGN MATRIX

Once the failed FR is identified, the next step is to search for the cause based on the design matrix involved. The design matrix shows the design decisions previously made regarding which of DPs affect the FR and which do not. Design matrix for Vapor Prime module decomposition is shown below:

$$\begin{bmatrix} \mathbf{FR1.1.1} \\ \mathbf{FR1.1.2} \\ \mathbf{FR1.1.3} \\ \mathbf{FR1.1.4} \\ \mathbf{FR1.1.5} \\ \boldsymbol{FR1.1.6} \\ \mathbf{FR1.1.7} \end{bmatrix} = \begin{bmatrix} X & O & X & O & O & O & O \\ O & X & O & O & O & O & O \\ O & O & X & O & O & O & O \\ O & O & O & X & O & O & O \\ O & X & O & O & X & O & O \\ \mathbf{O} & \mathbf{O} & \mathbf{X} & \mathbf{O} & \mathbf{X} & \mathbf{X} & \mathbf{O} \\ X & X & X & X & X & X & X \end{bmatrix} \begin{bmatrix} DP1.1.1 \\ DP1.1.2 \\ DP1.1.3 \\ DP1.1.4 \\ DP1.1.5 \\ \boldsymbol{DP1.1.6} \\ DP1.1.7 \end{bmatrix} \quad \textbf{(5.16)}$$

Firstly, DPs corresponding to 'X' elements in row 6 are examined. There are two off-diagonal 'X's and a diagonal 'X.' It must be answered whether the effects indicated by off-diagonal 'X's are compensated by DP1.1.6.

The off-diagonal 'X' elements are in (6, 3) and (6, 5). There are a series of instance of gas flow within the chamber, and the gas flow created is known to have some effects on wafer position during the process. DP1.1.3, selective gas supplying module, and DP1.1.5, exhaust plumbing system, are involved in the flow generation. By close examination, it is revealed that the effect of 'X' in (6, 3) are so large that DP1.1.6 can not compensate for it. That is, the gas flow for HMDS bake process – decomposition of FR1.1.3/DP1.1.3 – makes a wafer float and slip slightly. DP1.1.6 does not have an active holding DP, therefore, the effect can not be compensated by DP1.1.6.

STEP 3: CHECK EACH 'O' IN DESIGN MATRIX

'O' elements in design matrix should guarantee no effect to the FR from other DP. Many problems result from the fact that the actual design violates the conceptual decision on 'O' elements in design matrix. Those elements must be thoroughly examined to make sure the actual design coincides with the decision.

Design matrix is revisited, and 'O' elements in the 6th row are examined. It turns out that no clear evidence of a violation is found.

STEP 4: PROPOSE ALTERNATIVE DESIGN SOLUTIONS

Exploring the design matrix is followed by a design solution proposal. Conceptually, a design solution is proposed such that the observed problems in step 2 and 3 are eliminated.

The potential solutions are either adjusting DP1.1.6 to be able to compensate for the effect, or redesigning certain aspect of DP1.1.3 – or DP1.1.5 – so that the effect is minimized. The proposed solutions are:

1) Add additional sub FR for DP1.1.6 – 'supply holding force during the period of gas flow', and devise appropriate DP for the sub FR.

2) Adjust the aspect of DP1.1.3 which cause excessive effect. Operate at optimal pressure level such that gas flow does not cause wafer floating.

3) Sense wafer location and notify the location to robot. It may require robot design change as indicated in the 1st level design matrix, and make the system more complex.

4.3.1.3 Conclusion

It is demonstrated that the SA, once created, is used in a systematic way for diagnosing a system. A typical system diagnosis involves comprehensive knowledge base and rule-based inference engine. The knowledge base can vary from a shallow level, where the set of rules has been derived purely from the experience of human experts, to a deep level, where the set of rules has been derived from theoretical understanding of the domain [30]. A diagnosis based on an SA is a combination of deep and shallow knowledge based approach. Stating FRs, devising DPs, and analyzing the relationships involves both aspects. Strength of diagnosis using SA is that the SA itself is a diagnosis database, otherwise we have to create something new for diagnosis purpose. The 4-step procedure presented above may serve as the frame of a system diagnosis tool. With more rigorous features, an SA diagnosis system will be realized.

4.3.2 *System design from SA – synchronous algorithm*[29]

In section 4.2.2.3, it is noted that the design of the transport system must be consistent with the higher-level design decision: no effect from transport system, DP3.2, to FR1 and FR2. In order to maintain the consistency, four constraints (constraint 4 through 7) are identified. Those constraints essentially concerns quality deterioration due to transport delays. It is observed that, within the track system, those constraints are violated to cause the inconsistency in the design decisions.

This section begins with a discussion about the transport delays, and shows the approach and solution to the problem. It shows the problem identification from the SA and the design of a sub-system, which is a part of the top-down system design using AD.

4.3.2.1 Problem description

Completing a wafer transport requires three conditions to be satisfied: 1) a sending module, in which the wafer has just been processed, is ready, 2) a receiving module, in which the wafer will next be processed, is empty, and 3) a transport robot is available. Those three conditions are related to multiple branches of FR3.1/DP3.1 and FR3.2/DP3.2. With the system as it is, which has a limited number of transport robots, the three conditions are not always satisfied at the same time.

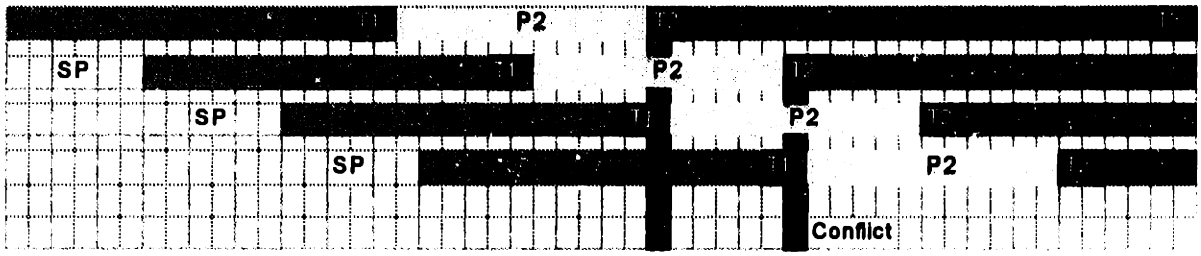


Figure 17. Illustration of transport conflict: P1, P2, and P3 refer to process 1, process 2, and process 3, respectively. T1, T2, and T3 stand for transport from process1 module to process2 module, and so forth. SP is sending period of the system

Figure 17 illustrates the situation when condition 1) and 3) has conflict. The second condition is guaranteed by employing sufficient number of process modules. DP3.1.1 is the determination of the number of modules: in other words, condition 2) is satisfied by DP3.1.1. Equation (5.10) is used to determine the number of modules. *Necessary* condition for the third condition is explained in description of FR3.1.2/DP3.1.2: $TAKT_{transport} \leq TAKT_{system}$. Note that it is not a sufficient condition but a necessary one. As shown in Figure 17, there are some instances where the third condition is not satisfied while the first condition is satisfied. Those conflicts are indicated as black vertical columns.

When a conflict occurs, the wafer pick-up has to be delayed. The original system handles the conflict by a reactionary algorithm. The reactionary algorithm determines which one of the simultaneous transport requests is fulfilled based on the process priorities. For example, at the first conflict in Figure 17, if T2 has higher priority than T1, then the algorithm decides to keep a wafer held in process1 module until T2 is done.

The reactionary algorithm requires many priority statements among the transport requests and rules to react to various situations. For instance of such a rule, T2 has priority over T1 until the delay of a wafer in P1 is less than 10 seconds. It also generates irregular delay for a certain process, which violates the constraints 4 through 7 imposed by the decision consistency checking matrix in section 4.2.2.3. Also, the wafer routes in the system are many and unpredictable causing the combinatorial complexity[5].

4.3.2.2 Synchronous algorithm

As one may notice already from Figure 17, transport conflicts are the function of sending period, total process time, and transport time. Also, each of the transports, T1, T2 and T3, is periodic function whose period is sending period. In mathematical formula,

$$
\begin{aligned}
T1 &= \text{Step}(t - (P1 + n^{*}SP)) - \text{Step}(t - (P1 + L1 + n^{*}SP)) \\
T2 &= \text{Step}(t - (P1 + L1 + P2 + m^{*}SP)) - \text{Step}(t - (P1 + L1 + P2 + L2 + m^{*}SP)) \\
T3 &= \text{Step}(t - (P1 + L1 + P2 + L2 + P3 + k^{*}SP)) - \text{Step}(t - (P1 + L1 + P2 + L2 + P3 + L3 + k^{*}SP))
\end{aligned}
\qquad \textbf{(5.16)}
$$

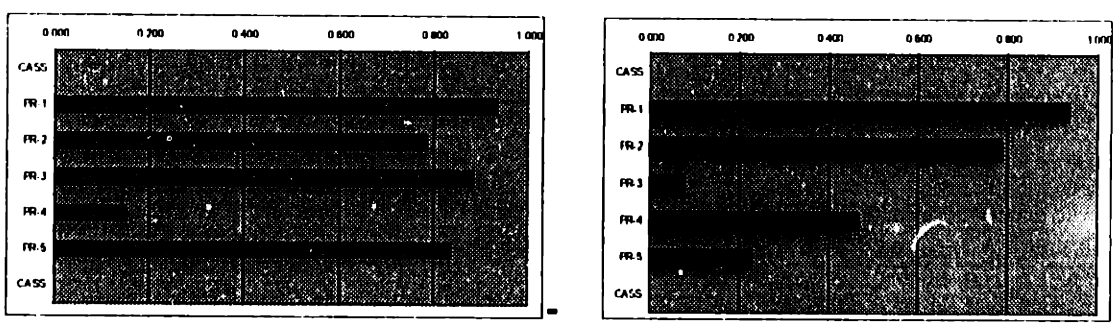
Where *Step(t)* is 0 if $t<0$ and 1 if $t>0$, *SP* is sending period, *Pi* is total process time of process step i, *Li* is transport time, and *n*, *m* and *k* are integer.

The fact that we want to avoid conflicts can be stated in formula using the above functions. If any of the below conditions happens, there will be conflict between some transports. Equation (5.17), therefore, is the mathematical expression of the coupling between condition 1) and 3).

$$
\begin{aligned}
&(n-m)SP - (L1+L2) < P2 < (n-m)SP \\
&(n-k)SP - (L1+L2+L3) < P2+P3 < (n-k)SP - L2 \\
&(m-k)SP - (L2+L3) < P3 < (m-k)SP
\end{aligned}
\qquad \textbf{(5.17)}
$$

The fact that the transport is a periodic function enables us to know exactly when the transport occurs within sending period as in Figure 18(a). Figure 18(a) is a normalized transport timing diagram of a system which consists of five process steps. Within one sending period, all of the transports occur at certain moments. The figure shows transport conflicts between T2 and T5, T5 and T3, T3 and T1.

If, with given parameters of SP, Li, Pi, one of the equation (5.17) is true, a queue times are needed to break the relationship. The aim of synchronous algorithm is to intentionally insert delays at non-critical process steps to break the coupling. These queue times can be found through the iteration steps with Pi substituted with Pi+Qi. Alternatively, instead of finding queue times that break the relationships such as equation (5.17), the queue times can be found using genetic algorithm. The latter method to solve for the queue times is discussed in reference [29]. Figure 18(b) shows the transports timing diagram after the conflicts are resolved with the queue times solution obtained using the latter method.



(a) (b)

Figure 18. (a) Location of transport requests, normalized by SP (b) Transports with no conflict with solved queue times

It should be noted that the constraints, e.g. no queue at coating process and no queue at bottleneck process, may make the iteration diverge: in some cases, there is no solution to match the throughput rate. We have to increase sending period, thus decrease throughput rate, to make the problem solvable. It is not because we are using intentional queue times. If the system runs on reactionary algorithm, the throughput rate will also slow down. The difference is that with the synchronous algorithm, why and where it is caused is known while we don't know in the latter case.

4.3.2.3 Conclusion

This section discussed the coupling of the three transport requirements and its undesirable effects - violation of the decision consistency. It briefly described the decoupling process. Three requirements for transport are identified: 1) sending module is ready, 2) receiving module is empty, and 3) transport robot is available. By recognizing there is a conflict – coupling – between the first and the third requirement, the essence of the problem was quickly approached. The decoupling solution chosen here is the synchronization of the two requirements. The synchronous algorithm computes queues for some of the processes that allow queue insertion. The process of coming up with synchronous algorithm illustrates how the AD approach helps with identifying a problem and finding a proper solution.

5 Conclusions

This thesis discusses various aspects of the concept of a System Architecture as a systematic design approach. The concept of System Architecture is reviewed in the context of Axiomatic Design theory. A list of hypotheses is presented to formalize the issues concerning system design and System Architecture, and are discussed based on Axiomatic Design theory. In the last chapter, a case study is presented. The discussion of the case study illustrates the process of creating a System Architecture. Two specific examples of the application are also given for the system diagnosis and the design of wafer flow management algorithm. The case study that shows the creation of System Architecture of the track system and the examples give evidence for the validity of the hypotheses presented in this thesis.

5.1 System design using the concept of System Architecture

System design is basically the same as the design of simple products when considered from the perspective of Axiomatic Design. The design of a system begins with top level FRs, and the mapping and zigzagging process continue until all the leaf modules are identified so that the design is readily implemented. During the design at high levels, a designer has to deal with abstract concepts rather than solid physical solutions, and the design activities at these levels are mostly to impose conceptual directions for further design. Assumptions for detailed features and constraints must be made in proceeding with the decomposition at high-levels, and their satisfaction should be checked afterward.

The System Architecture is an effective way of representing design decisions made during the design process. The essential elements of the System Architecture are various design decisions made during system design. It captures and organizes every FR and DP, and clarifies the relationship among these functional and physical elements.

A list of hypotheses is presented for the discussion of system design and System Architecture. Those hypotheses address largely three areas: 1) system representation and System Architecture, 2) system design and System Architecture, and 3) system design control and System Architecture. They are discussed based on the Axiomatic Design theory, and shown to be reasonable.

5.2 Application of System Architecture

In the last chapter, a case study is presented. As an illustration of generating a System Architecture, or equivalently designing a system based on Axiomatic Design theory, the photoresist processing system is analyzed.

The decomposition process is shown to the third level of the system hierarchy. Since the decomposition for those levels are at an abstract level of design, the issues discussed in chapter 2 and chapter 4 regarding system design are relevant during the decomposition process. For example, it is demonstrated that the design with insufficient – or abstract – information can be done within the framework of Axiomatic Design.

The last two examples concern the more specific problem of system diagnosis and system design from decoupling, respectively. Although not all of the details are presented in this thesis, since some of the contents are proprietary to the research sponsor, they are sufficient for illustrative purposes.

6 Reference

1. Suh, N.P., *The Principles of Design*, Oxford University Press, 1990
2. Altshuller, G.S., *Creativity as an Exact Science*, New York: Gordon and Breach, 1988
3. Pahl, G., Beitz, W., *Engineering Design: A Systematic Approach*, London: Springer-Verlag, 1996
4. Hubka, V., Eder, W.E., *Principles of Engineering Design*, London: Butterworth Scientific, 1982
5. Suh, N.P., *Axiomatic Design: Advances and Applications*, to be published, 1999
6. Albano, L.D., "An Axiomatic Approach To Performance-Based Design", *Ph.D. Thesis, Department of Civil Engineering*, Massachusetts Institute of Technology: Cambridge, 1992
7. Templeman, A.B., *Civil Engineering Systems*, London: Macmillan Press Ltd., 1982
8. Ulrich, K.E., Eppinger, S., *Product Design and Development*, New York: McGraw-Hill, Inc., 1995
9. Tate, D., "A Roadmap for Decomposition: Activities, Theories, and Tools for System Design", *Ph. D. Thesis, Department of Mechanical Engineering*, Massachusetts Institute of Technology: Cambridge, 1999
10. Nordlund, M., Tate, D., "Research methods for design theory", working paper, MIT: Cambridge, 1996
11. Suh, N.P., "Axiomatic Design Theory for Systems", working paper, MIT: Cambridge, 1998
12. International Council on Systems Engineering, *Systems Engineering Handbook*, International Council On Systems Engineering (INCOSE): San Francisco, 1998
13. Rechtin, E., *Systems Architecting in Manufacturing*, speech note, San Jose, 1998
14. Suh, N.P., "Design and Operation of Large Systems", *Journal of Manufacturing Systems*, Vol.14, No.3, pp.203-213, 1995
15. Ulrich, K., "The role of product architecture in the manufacturing firm", *Research Policy*, Vol.24, pp.265-293, 1993
16. Baldwin, D.F., "Microcellular Polymer Processing and the Design of a Continuous Sheet Processing System", *Ph.D. Thesis, Department of Mechanical Engineering*, Massachusetts Institute of Technology: Cambridge, 1994
17. Hintersteiner, J.D. "A Fractal Representation for Systems", *Proceedings of the 1999 International CIRP Design Seminar*, Enschede, the Netherlands, 1999
18. Kim, S.J., Suh, N.P., Kim, S.-K., "Design of software systems based on axiomatic design", *Annals of the CIRP*, Vol. 40, pp.165-170, 1991
19. Clausing, D., *Total quality development*, New York, NY: ASME press, 1993
20. Nordlund, M, Tate, D., Suh, N.P., "Growth of Axiomatic Design through Industrial Practice", *3rd CIRP Workshop on Design and the Implementation of Intelligent Manufacturing Systems*, pp. 77-84, 1996

21. Do, S.H., "Design of Axiomatic Design Software", working paper, MIT: Cambridge, 1998

22. Nordlund, M., "An Information Framework for Engineering Design based on Axiomatic Design", *Ph.D. Thesis, Department of Manufacturing Systems*, Royal Institute of Technology (KTH), Stockholm, Sweden, 1996

23. Tate, D., "A Roadmap for Decomposition: Activities, Theories, and Tools for System Design", *Ph. D. Thesis, Department of Mechanical Engineering*, Massachusetts Institute of Technology: Cambridge, 1999

24. Tate, D., Nordlund, M., "A Design Process Roadmap as a General Tool for Structuring and Supporting Design Activities", *Proceedings of the 2nd world conference on integrated Design and Process Technology*, Austin, TX, 1996

25. Thompson, L.F., Willson, G., Bowden, M., *Introduction to Microlithography*, Washington, DC: ACS Professional Reference Book, 1993

26. Hintersteiner, J.D., Tate, D., "Command and Control in Axiomatic Design Theory: Its Role and Placement in the System Architecture", *the 2nd International Conference on Engineering Design and Automation*, Maui, HI, 1998

27. Lee, T.S., *Standardizing the 200-APS*, internal report, Silicon Valley Group, Inc.: San Jose, CA, 1998

28. Perkinson, T.L., Gyurcsik, R.S., McLarty, P.K., "Single-Wafer Cluster Tool Performance: An Analysis of the Effets of Redundant Chambers and Revisitation Sequences on Throughput", *IEEE Transactions on Semiconductor Manufacturing*, Vol.9, pp.384-400, 1996

29. Oh, H.L., Lee, T.S., "Synchronizing Wafer Flow To Achieve Quality In a Single-Wafer Cluster Tool", working paper, 1999

30. Leang, S., Spanos, C.J., "A General Equipment Diagnostic System and its Application on Photolithographic Seuences",*IEEE Transactions on Semiconductor Manufacturing*, Vol.10, pp.329-343, 1997