Characterization of Variance in the High Volume Production of Air Bag Control Units

by

Anthony von Behring Reese

B.S., Industrial Technology
California State University, Los Angeles, 1991

Submitted to the Department of Mechanical Engineering and the Sloan School of Management in partial fulfillment of the requirements for the degrees of

Master of Science in Mechanical Engineering
and
Master of Science in Management

at the
Massachusetts Institute of Technology
June 1995

Signature of Author ______________________________

Certified by ______________________________
Anantaram Balakrishnan, Associate Professor
Sloan School of Management
Thesis Advisor

Certified by ______________________________
John G. Kassakian, Professor
Department of Electrical Engineering and Computer Science
Thesis Advisor

Certified by ______________________________
Woodie C. Flowers, Professor
Department of Mechanical Engineering
Thesis Reader

Accepted by ______________________________
Ain A. Sonin, Chairman
Committee on Graduate Students, Mechanical Engineering

Characterization of Variance in the High Volume Production of Air Bag Control Units

by

Anthony von Behring Reese

B.S., Industrial Technology
California State University, Los Angeles, 1991

Submitted to the Department of Mechanical Engineering and the Sloan School of Management in partial fulfillment of the requirements for the degrees of

Master of Science in Mechanical Engineering
and
Master of Science in Management

ABSTRACT

This thesis examines the difficulty of characterizing variance in the high volume production of air bag control units. In the production of air bag controllers several problems exist that prevent understanding of test and repair feedback:

1. Important characteristics of electronic circuits (e.g., circuit timing) are not directly observable or understood without the aid of special test equipment.
2. Batch processing generates multiple assembly combinations.
3. Circuit boards change positions within batches.
4. Circuit boards leave and re-enter the production flow.

The process problems (2–4) are classified as contributing directly or indirectly to multiple assembly combinations. Multiple assembly combinations are shown to prevent rapid and accurate process feedback. "Process Attribution" is introduced a simple means of feedback that clarifies the manufacturing environment and enables the workforce to make timely and relevant decisions about the performance of the process. To illustrate the application of Process Attribution, specific examples from air bag controller assembly lines are used.

Thesis Advisors

Dr. Anantaram Balakrishnan, Associate Professor of Management
Dr. John G. Kassakian, Professor of Electrical Engineering and Computer Science

Acknowledgments

I gratefully acknowledge the support and resources provided to me through the MIT Leaders for Manufacturing Program, a partnership between MIT and major U.S. manufacturing companies.

I wish to thank my advisors, Anantaram Balakrishnan and John Kassakian for their advice and constructive criticism throughout the thesis process. Their support was essential to its success.

I also thank the many engineers and technicians at Delco Electronics Vehicle Systems whose assistance allowed the project to be completed.

Finally, I would like to thank my wife, Teri, for her love, support, and encouragement during the internship and throughout the writing of this thesis.

Table of Contents

Table of Figures and Tables

1. Introduction and Motivation

1.1 Introduction

This thesis examines the difficulty of characterizing variance in the high volume production of air bag control units. Decisions about the state of control can be made using test and repair data and the product only if some thought has been put into matching the feedback mechanism to the process. Once real time feedback has been established, testing becomes a value adding activity and quality (product and process) can improve.

This study asserts that "Process Attribution" is a simple means of feedback that clarifies the manufacturing environment and enables the workforce to make timely and relevant decisions about the condition of the process. Process Attribution is defined as visibly marking the product to identify process steps. Recording the process steps on the product makes visible the specific details of the process used to manufacture a product. To illustrate the application of Process Attribution, specific examples from air bag controller assembly lines are used. These examples are based on interviews, experiments, and assembly work performed by the author and others at Delco Electronics. Some data presented in this thesis has been disguised for proprietary reasons, but the relationships have been maintained to illustrate key points.

In this thesis we will show that feedback problems exist, that the lack of feedback raises manufacturing costs, and that Process Attribution is a simple way to provide feedback. Further, we will examine some of the wider implications for Process Attribution and explain some problems found during implementation.

1.2 Motivation

Quality and Costs

The desire for increased quality and reduced cost make visualizing the process important. Air bag control units are safety devices. With any safety device, reliability is a concern. Shawn [1] reports that Delco Electronics produces a "best in class" air bag controller. Providing high levels of quality is important because it reduces the direct product cost, waste, testing, and rework.

In an effort to reduce costs, specific process variances must be understood. Making the process visible helps characterize process variances. Characterizing variances in the production process will focus improvement activities, increase productivity, and should be part of a long term testing strategy.

Feedback May Be Captured By Visualizing The Production Process

Delco Electronics assembles on the order of 1,000,000 air bag controllers per year, with very short cycle times. Operators monitor complex equipment, and the short cycle times can make it extremely challenging to detect and diagnose out of control processes. Information systems designed to capture the complexity and identify out of control situations are the contemporary means for dealing with this. But these are just tools. The business of quality improvement relies on ever-improving knowledge of the manufacturing process, and skilled problem solvers on the shop floor [2].

Process Attribution helps the workforce understand how the process is performing. Making the production process performance visible to the workforce is in line with many of today's broader improvement activities [3] [4]. These activities encourage worker involvement in the improvement of production systems. Workforce involvement rests on the assumption that the worker can receive timely feedback about production defects in order to understand the process and suggest improvements. From the shop floor, the process of high volume electronic assembly production is not easy to understand. One way to capture feedback is by making the process visible. This makes the process clear and stimulates improvement in a production environment.

Problems in Feedback

In the production of air bag controllers several problems exist that prevent understanding of test and repair feedback:

1. Important characteristics of electronic circuits (e.g., circuit timing) are not directly observable or understood without the aid of special test equipment.
2. Batch processing generates multiple assembly combinations.
3. Circuit boards change positions within batches.
4. Circuit boards leave and re-enter the production flow.

While problem (1) relates to the nature of the product, problems (2-4) relate to the layout and operation of certain processes. These problems make observations of process performance difficult. This lack of process transparency prevents feeding back test and repair information in a timely manner and reduces process control. With little process transparency and no real-time process feedback, systematic sources of waste reduce productivity, add faults to the product and increase the cost of production.

1.3 Thesis Organization

Chapter Two introduces the electronics assembly process. Chapter Three provides examples of each process problem identified in the high volume production of air bag controllers. Chapter Four asserts that Process Attribution is a method for helping close the process feedback loop in high volume electronics assembly. Examples of Process Attribution at Delco Electronics and findings are described. The benefits associated with providing real-time feedback are presented in Chapter Five. Chapter Six looks at the wider implications of Process Attribution, and the conclusions are in Chapter Seven.

2. Background

2.1 Introduction

This chapter outlines the electronics assembly process for air bag controllers. Discussing volumes, geometry and packaging, process and information flows provides the framework for the four problems identified in Chapter Three. Defining the terms sequence, order, and assembly combination completes the background material.

2.2 Overview of the Electronics Assembly Process

Volumes

Production of air bag controllers is accomplished in lot sizes (batches) of 1–5 thousand circuit boards with multiple product changeovers scheduled daily. Delco Electronics builds half a dozen unique air bag circuit board types in high volume (100,000 to 500,000 per year). General Motors car divisions purchase the majority of these air bag controllers.

Geometry and Packaging

An air bag controller is typically built on a single four-layer printed circuit board. The majority of the smaller components (resistors and capacitors) are surface mounted to the bottom side of the circuit board. The larger surface mounted components and the thru-hole parts are on the top side of the circuit board. The assembled and tested circuit boards are packaged in a die cast assembly and mounted in various locations within the passenger compartment of the vehicle.

Process Flow

The process (Figure 2.1) starts with the bare circuit board. Each board is visually inspected as it is loaded into the solder paste machine. The solder paste machine applies a thin coating of solder paste to the topside circuit board pads. The board passes through top side chip placement and then the reflow oven. The reflow oven liquefies the solder paste creating the joints that complete the electrical and mechanical connections. The circuit board with top side components is then tested at x-ray for conformance.

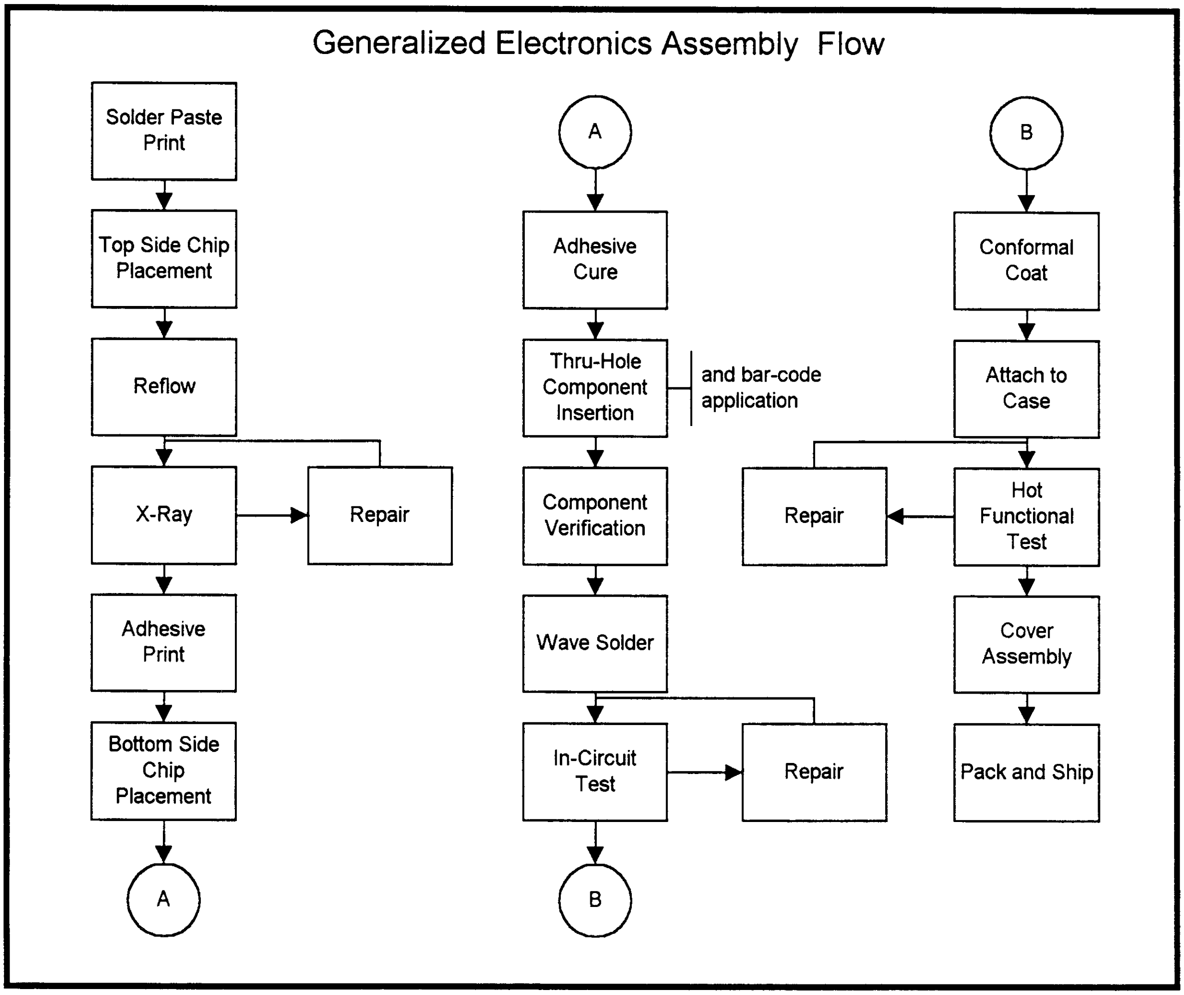

Figure 2.1 Generalized electronics assembly flow for air bag controllers.

The circuit boards are then turned over and loaded into the adhesive printer. A glue die transfers glue to locations between circuit pads where components will be placed. The board passes through bottom side chip placement and then the adhesive cure oven. The cure oven hardens the adhesive and creates the joints that secure the bottom side components until they are soldered.

At the thru-hole insertion area, all the large leaded components and a solid state accelerometer are placed on the top-side of the circuit board. A wave solderer solders both the bottom side and thru-hole components. The completed board assemblies are tested for conformance at in-circuit testing before a protective conformal coating is placed over the board. The coated circuit board is then assembled in a case and functionally tested before shipping. Using both surface mounted and thru-hole components is called "mixed" or "Type II" printed circuit board assembly. This flow is consistent with "mixed" printed circuit board assembly as documented in Prasad [5].

Information

An information system collects data from the manufacturing process. Product tracking starts at thru-hole component insertion where a serialized product bar-code is applied (Figure 2.1). The bar-code data and information system assure that tests are passed as the product continues through the production system. In-circuit testing scans each product bar-code, performs a test, and sends the test results to the information system. Functional test electronically transfers the bar-code data to the product memory, tests the assembly, and sends the results to the information system. The information system creates a test file for each assembly. Repair technicians send repair data to this test file. This test file captures the date and time of tests, cause of failure, and repair.

2.3 Definitions

Two terms that are easily confused are sequence and order. Changes in order and sequence produce assembly combinations. These terms are defined here to assist the reader in understanding the details of the electronics assembly process and the related problems.

Sequence: as it is used in this thesis is the arranging process. Sequence describes rules for successive actions (e.g., place this board here and that board there)

Order: as it is used in this thesis is the result or output of a sequence (e.g., the resulting arrangement of boards).

Assembly Combination: a specific mix of interchangeable components and processes that produces an air bag controller (e.g., boards have same part or process, but use a different source or machine).

3. The Problems

3.1 Introduction

The process has four problems which inhibit identification of specific process variances and real time feedback of test and repair data.

1. Important characteristics of electronic circuits (e.g., circuit timing) are not directly observable or understood without the aid of special test equipment.
2. Batch processing generates multiple assembly combinations.
3. Circuit boards change positions within batches.
4. Circuit boards leave and re-enter the production flow.

This chapter examines the four problems identified in the high volume production of air bag controllers. The process problems are then classified as contributing directly or indirectly to multiple assembly combinations. Multiple assembly combinations are shown to prevent rapid and accurate process feedback.

3.2 Problem One: The Nature of the Product

Many of the important qualities of the assembled electronic circuit board are not observable. There are hundreds of parts on each side of a circuit board. Many of these parts have no markings. Circuit functionality requires the interaction of several parts, connected by traces on the circuit board. The only way to verify electrical properties of the circuit (e.g., continuity, timing, resistance, or capacitance) is to perform an electrical test. This testing requires either an in-circuit tester (component level test) or a functional tester (circuit level test) or both.

In addition to electrical characteristics, physical characteristics affect functionality and reliability of the assembled electronic circuit board. Visual inspection (manual and automated) and x-ray tests check for component presence, positioning, and solder joint quality.

3.3 Problem Two: Multiple Assembly Combinations within Surface Mount

During the production of air bag controllers many assembly combinations occur. Because of the nature of electronic circuit board assemblies these combinations are largely unobservable on the shop floor. The assembly combinations hide the effect of specific process variances (e.g., one source of resistors contains defective parts). These combinations must be made explicit for test data to be useful in solving problems related to specific component parts or processes.

From a general perspective (Figure 2.1) products pass from one operation to the next in a fixed sequence. At a finer level of detail, however, the surface mount process at Delco Electronics does not rigidly fix the production sequence and thus produces multiple assembly combinations within the same batch of circuit boards.

Figure 3.1 shows the layout of two chip placement machines performing top side chip placement. Both machines simultaneously populate two boards, one in position A and another in position B. Each machine performs the same operation on both boards, but uses different tooling, and parts from different part reels for each position. The top side component placements are divided between the two machines. This configuration provides

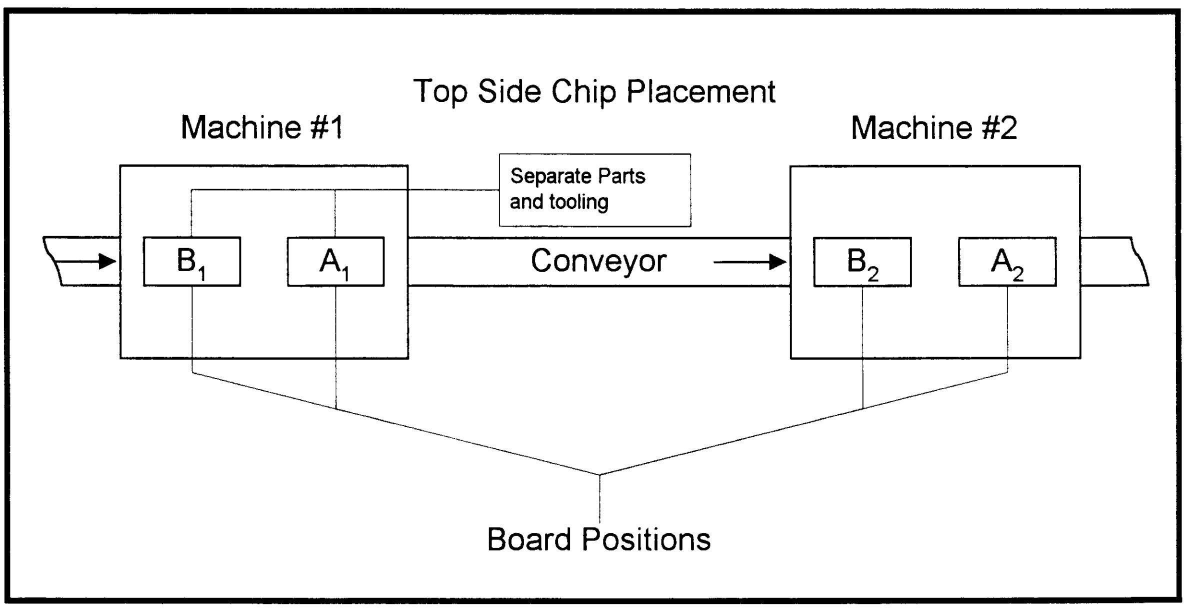

Figure 3.1 Layout of two top side chip placement machines.

a minimum of two board assembly combinations: circuit boards populated with parts and tooling from positions A_1 and A_2, denoted as (A_1,A_2), and circuit boards populated with parts and tooling from positions B_1 and B_2, denoted as (B_1,B_2). Because both machines provide half of the parts for a complete assembly, only these two combinations are assumed to occur, and when faults are detected diagnosis is performed to determine whether the effected boards are (A_1,A_2) or (B_1,B_2). Observations of this process, however, indicate that though circuit boards are typically built as (A_1,A_2) or (B_1,B_2) they are also built as either (A_1,B_2) or (B_1,A_2).

The additional assembly combinations (A_1,B_2) and (B_1,A_2) present a problem in the identification of specific machine-position variances. In practice, operators diagnose the process by clearing all machines of work-in-process (i.e., interrupting throughput) and allowing two boards, marked as A and B to be processed. The A and B boards ((A_1,A_2) and (B_1,B_2) assembly combinations) are compared with the defect data (the defective product has typically been repaired). This procedure often does not allow timely identification of process variances for several reasons. First, the procedure assumes only a limited set of assembly combinations and forces an assembly sequence that may not have produced the actual defect. Second, the diagnosis depends on a product manufactured during a different time than the original defect. Third, unique interactions due to assembly combinations are not understood. The position specific variances of each top side chip placement machine are lost in the confusion produced by the many assembly combinations. Because throughput is lost during diagnosis, a timely diagnosis is important.

The circuit board flow through two chip placement machines can be conceptualized by thinking of each machine as two separate processors in parallel with a service preference for processor A (Figure 3.2). The dotted lines represent the preferred board flow, where position B_1 or B_2 is used only if position A_1 or A_2 is taken. Because of the preference for processor A, additional assembly combinations (A_1,B_2) and (B_1,A_2) occur when circuit boards do not arrive as rapidly as the chip placement machine can process them. The relationship between arrival time and processing time determines the board combination. Processing time has two components, the cycle time (time to populate

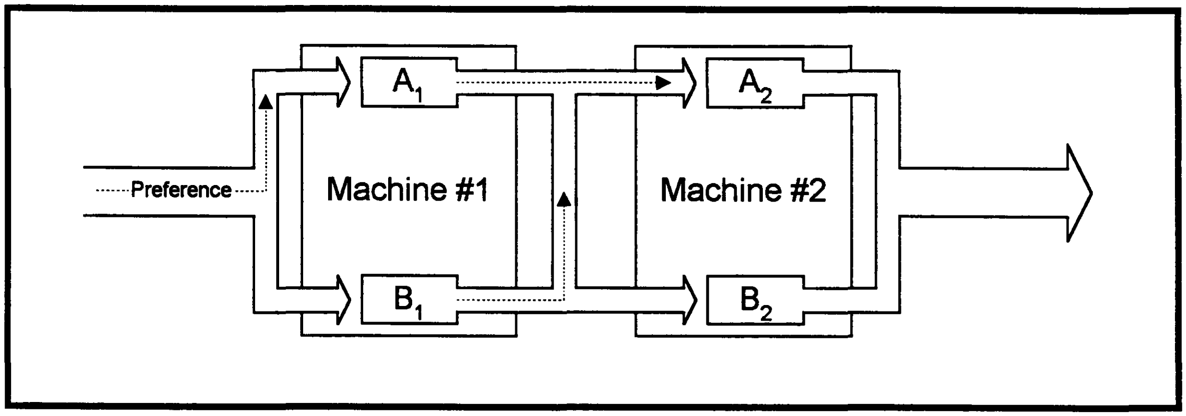

Figure 3.2 Conceptual diagram of circuit board flow.

the circuit board), defined as P, and the delay time that elapses if only one or zero boards are available to load into the machine, defined as D. As long as the inter-board arrival time (time between consecutive circuit board arrivals), defined as T, is small compared to the processing time $(P+D)$ only the standard combinations (A_1,A_2) and (B_1,B_2) are produced. The number of boards between the machines also affects assembly combinations. Once an odd number of boards are on the conveyor (Figure 3.1) then all the circuit boards are produced as combinations (A_1,B_2) and (B_1,A_2).

Measuring The Likelihood of Producing (A_1,B_2) and (B_1,A_2) Combinations

To determine the likelihood of producing non-standard board combinations requires tracking assembly combinations under varying conditions of circuit board availability. Since stationary states of operation are rarely observed in operation it is difficult to experiment with the physical system. Law and Kelton [6] write that "the opportunity to estimate the performance of the system under some projected set of operating conditions," is one of the reasons for the wide spread appeal of simulation. Additionally, we desire to capture the variability of inter-board arrival times that prevents this system from being purely deterministic. A Witness™ [7] simulation model can describe the two topside machines and three conveyors in top side chip placement process.

Since board combinations are a function of inter-board arrival times T, delay times D, and cycle times P, the parameter $\alpha = \frac{T}{D+P}$ can be used to understand how each of these might affect assembly combinations. To experiment with board combinations, parameter α was varied over a reasonable range. In practice, process problems can increase inter-board arrival times temporarily to levels where $\alpha > 1$. The arrival rate is not assumed to be constant over the trial interval and a Poisson process was used to generate inter-board arrival times. The Poisson process provides the variability associated with the solder paste operation which supplies circuit boards to machine #1. Figure 3.3 graphs the results of the experiment.

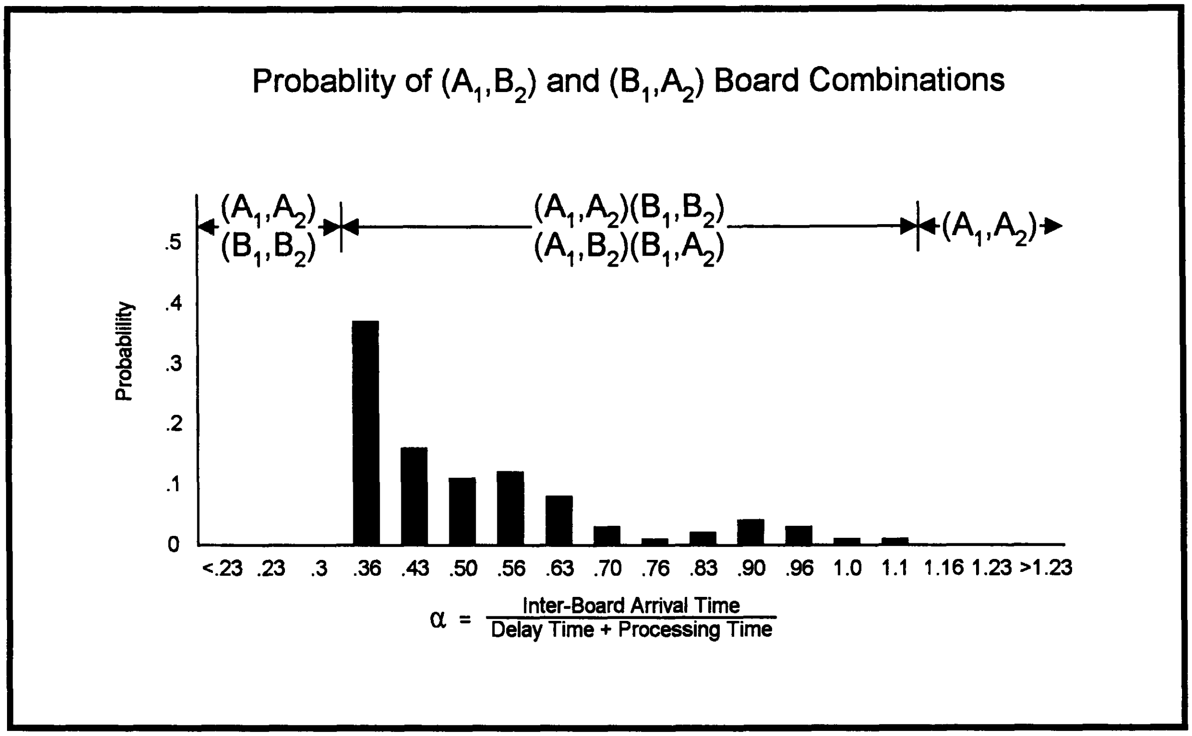

Figure 3.3 Probability of (A_1,B_2) and (B_1,A_2) board combinations at top side chip placement.

Several conclusions may be drawn from the simulation results. There is a range of α over which certain assembly combinations will occur. With α less than .3, and the conveyor between machines containing an even number of circuit boards, only $(A_1,A_2)$and (B_1,B_2) circuit boards are produced. With α between .36 and 1.1 a mix of all four combinations is produced and finally values of α greater than 1.1 produces only (A_1,A_2) boards. If α increases to .36, there is a 37% chance that the boards produced will be of the combinations (A_1,B_2) and (B_1,A_2). As α increases, the likelihood that the boards produced will be (A_1,B_2) and (B_1,A_2) reduces as the majority of boards become (A_1,A_2). This system will not actively recover from having an odd number of boards on the conveyor. If α increases only briefly to above .36, then possibly all circuit boards will be of the (A_1,B_2) and (B_1,A_2) variety. A discussion of machine logic, the complete Witness™ model, and simulation details are in Appendix A.

The simple case of two machines each processing two parts can be extended to include the bottom side surface mount process which requires three surface mount machines each processing three circuit boards. There are $2^2 \times 3^3 = 108$ possible combinations for the top and bottom side surface mount operations on a single circuit board type. The surface mount process maintains the first-in first-out relationship, but permits many assembly sequences. Since high speed machines process these circuit boards and the circuit boards appear identical (Problem One), the many assembly combinations are unnoticed in production.

3.4 Problem Three: Changing Positions Within Batches During Handling

As described above some operations do not limit the assembly sequence (e.g., top side and bottom side chip placement). Other operations, such as reflow and adhesive curing, do constrain the sequence, but the sequence itself alters the order of circuit boards within batches.

To balance the line capacity, the circuit board flow splits from series to parallel for some operations and consolidates from parallel to series for others. Depending on the method of board handling, splits and consolidations change the order of boards within a batch. Figure 3.4 details the board handling at adhesive curing. In this case, the automated board handlers fix the sequence, but change the order of the circuit boards.

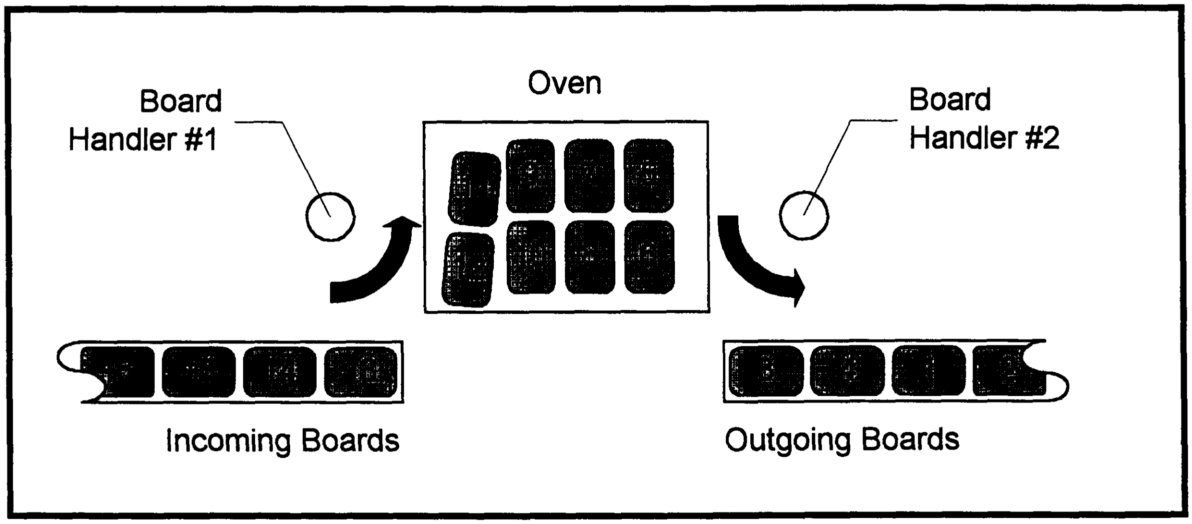

Figure 3.4 Board handling at adhesive cure illustrating board flow change from series to parallel and back to serial.

The changing order of boards throughout the manufacturing process presents a problem in the identification of specific machine variances. The production order establishes the time-specific process events. Changing the order of processes causes two problems. First, flow splits after a change in order produce additional assembly combinations (e.g., boards have the same part from different sources). Second, series flows after a change in order produce time specific assembly combinations (e.g., boards have same part and source, but at different times). At adhesive cure, the flow splits and produces additional assembly combinations (adhesive cure left, or right) which are determined by arrival order. The change in board order during handling at adhesive cure produces time specific combinations of the upstream and downstream operations. Table 3.1 shows combinations in time due to changed board order at adhesive curing.

Time Specific Combinations Due to Changed Board Order at Adhesive Cure

Board Number	Bottom Side Chip Placement	Thru-hole Component Insertion
1	First	Second
2	Second	First
3	Third	Fourth
4	Four	Third

Table 3.1 Time specific combinations of bottom side chip placement and thru-hole component insertion due to changed board order at adhesive curing.

Combinations in time make the identification of specific machine-time variances difficult to identify.

If the sequence is fixed by the board handling equipment, the ordering problem can be resolved by changing the sequence (e.g., make board handler #2 at adhesive cure rotate clockwise and move the outbound conveyor to the other side of the board handler).

The problem of changing board order exists whenever flows are split or consolidated and is not restricted to automated handling systems. Many operations require that operators sequence boards (either splitting or consolidating the flow). In these situations the sequencing procedure and resulting order may change by the week, day, shift, hour or minute. Table 3.2 categorizes the splitting and consolidating of flows in the production of air bag controllers.

Flow Splits and Consolidations

	Split by		Consolidated by	
Process	Operator	Board Handler	Operator	Board Handler
Solder Paste			2 into 1	
Reflow Oven		1 into 2		
x-ray	2 into 4		4 into 2[1]	
Cure Oven		1 into 2		2 into 1
Thru-hole	1 into Many		Many into 1	
In-Circuit Test		1 into 3	3 into 1	
Conformal Coat	1 into 8		8 into 2	
Attach to Case				2 into 1

Table 3.2 Flow splits and consolidations in the production of air bag controllers.

Measuring Product Order

To measure the overall change in product order, arrival data at the last step in the manufacturing process was collected. Air bag controllers are serialized at thru-hole component insertion (Figure 2.1) where a bar code, with serial number, is applied to the assembled circuit board. An arrival sample ($n = 150$) of serialized controllers was tracked

[1] An inconsistency appears here as product is placed into inventory

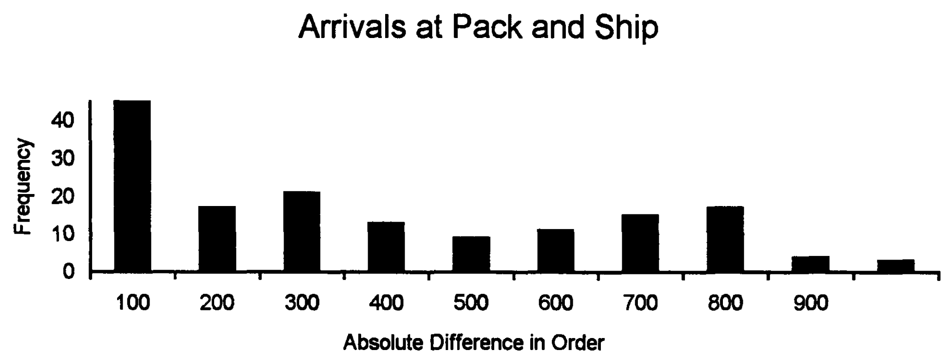

Figure 3.5 Absolute change in order from thru-hole component insertion through pack and ship.

(differences in bar-code numbers) at the last process step. Figure 3.5 graphs the distribution of the absolute change in order between consecutive arrivals at pack and ship. Forty percent of the arrivals are out of order by more than 500 serial numbers. With such large changes in production order, the production of defects and their arrival time at test and repair adds no meaningful information about specific process variability. Again, the identical appearance of the circuit boards (Problem One) makes these changing board orders inconspicuous in production.

3.5 Problem Four: Exiting and Re-entering Production Flow

Throughout the electronics assembly process there are test and repair loops which remove defective circuit boards from production (Figure 2.1). Failed circuit boards are processed through the repair area and then return to production. Additionally, some boards are removed for inspection and then returned to the process. These exits and re-entries are a special case of splitting and consolidating flows with delays. Repair and inspection delays spread the re-entry over several batches of circuit boards, change the ordering within multiple batches, and reduce the speed of feedback to the system.

To learn about the delays in feedback of test and repair information, a time study was performed at each test and repair loop. Testers provide a printout of the defect information with a date and time of the failure. After synchronizing the testers with a time clock the technicians were asked to clock each receipt upon start and finish of the circuit board repair. The receipts were collected for five days at each test and repair loop. While the average time to repair a circuit board can be measured in minutes, the average delays between failing a test and reaching a repair technician are measured in hours (Figure 3.6). These delays prevent any information that can be gathered from the product about the process from being useful.

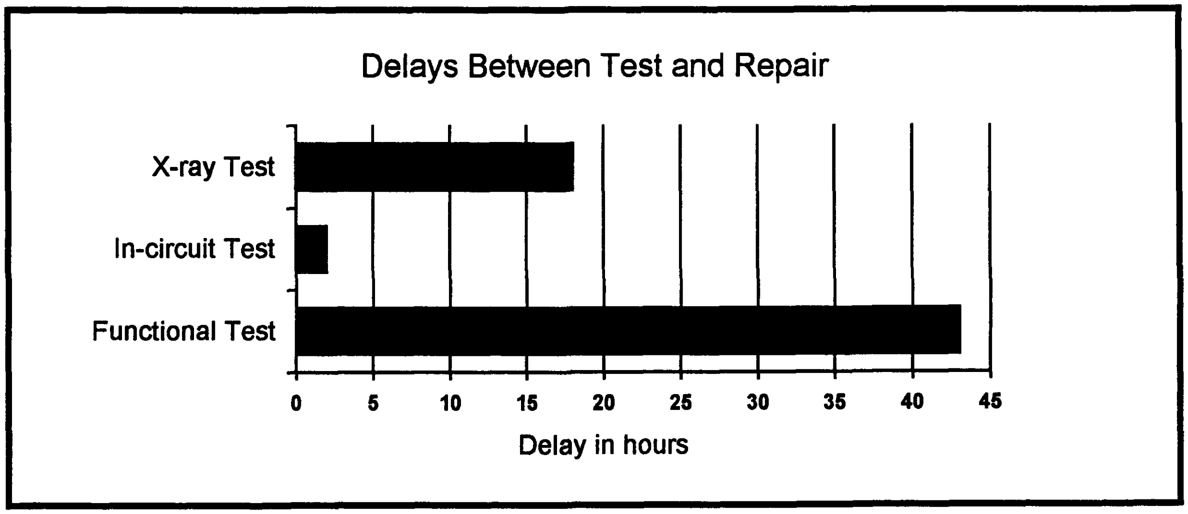

Figure 3.6 Delays between test and repair.

3.6 Classification of Problems

Each of the four problems described above contributes to the prevailing problem of many assembly combinations. Problem One prevents identification of the various assembly combinations. Problem Two directly produces multiple assembly combinations. Problems Three and Four indirectly produce multiple assembly combinations through handling of the product. The prevailing problem of many assembly combinations prevents understanding specific process variances in a useful way. Understanding specific process variances is necessary for problem solving, process improvements, and cost reductions.

Understanding specific process variances helps to solve three types of production problems. By knowing that defects are related to an isolated machine, or a position within a machine, a specific process variance has been identified. Knowing that similar defects appear randomly distributed over several machines and processes provides evidence of systematic problems (e.g., design problems). Without this evidence, one cannot know that the collected defects actually account for all the possible assembly combinations. Finally, defects that result from interactions among certain process combinations become identifiable. Locating the cause of defects (e.g., machine specific, design related or an interaction) is the first step in process improvement.

3.7 Feedback

While testing of the air bag controllers provides quality assurance, the test data alone can provide no insight to quality improvement. Real time feedback is needed in process control and improvement activities. The nature of electronics is such that ensuring quality at the source is often not possible. Because key quality characteristics are not visible (Problem One) assembly errors can occur that are only detectable with test equipment. With no real-time feedback, process corrections must occur in two steps. First, product level information must identify when a process problem exists. Then, through process diagnosis, information is collected to perform a root cause analysis. Figure 3.7 diagrams the separate operations of feedback. Loop **A** represents product defect identification. Feedback loop **B** represents process diagnosis needed to identify a root

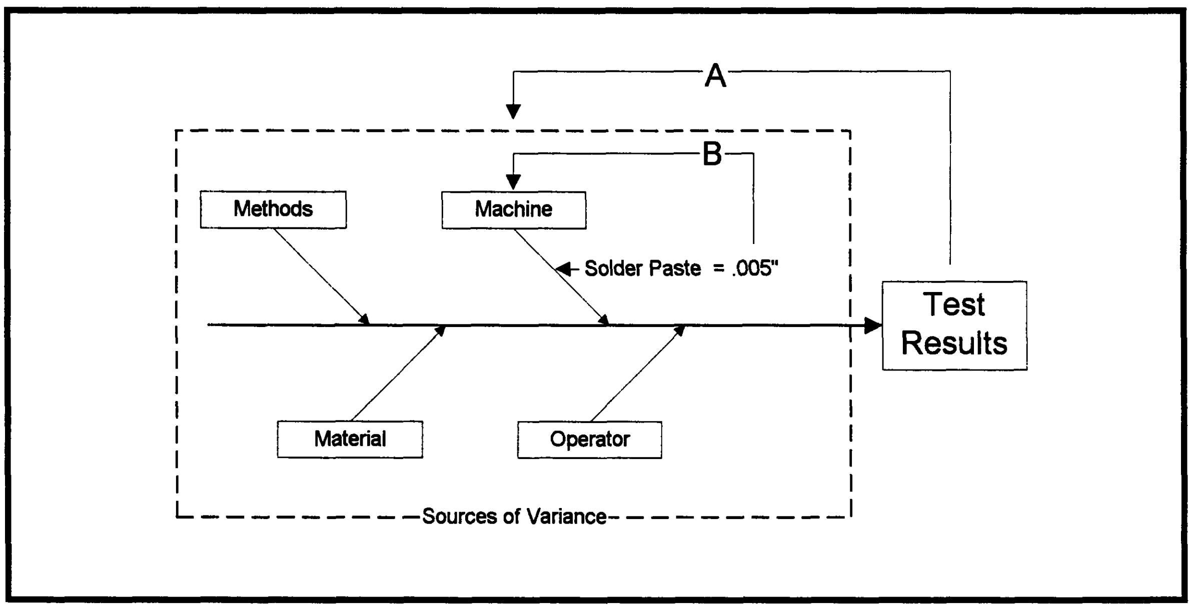

Figure 3.7 Feedback Diagram Showing Separate Operations of Defect Identification (**A**) and Process Diagnosis (**B**).

cause. The optimal case of only using measurements of the process as control variables (only loop **B**) requires an exact understanding of the relationship between the process and the product characteristics. While this may be possible in some mature electronics assembly operations, new products, such as air bag controllers, create new relationships.

To achieve real time feedback in this environment requires a method of connecting loop **A** with the appropriate loop **B**. Making assembly combinations explicit links these two loops, allowing real time feedback.

3.8 Summary

Four problems have been identified; one in perceiving assembly combinations of electronic products and three that increase assembly combinations. These problems of observing the many assembly combinations and the constantly changing assembly order, make specific process variability difficult to identify. Without identifying specific process variability, machine specific, design related, or interaction problems are difficult to locate. Without real-time feedback, process correction occurs by identifying product defects and then collecting specific process information.

Hitoshi Kune [8] reminds us that a structured approach to problem solving involves identifying the main causes of a specific problem. In order to identify specific process variances, reducing assembly combinations is one such approach. The creation of multiple assembly combinations between processes can be reduced by planning and organizing the processes. But reducing the direct creation of multiple assembly combinations will require redesigning process equipment. Without redesigning equipment and processes, however, a method for making the assembly combinations explicit is needed to connect product defects with specific processes. Process Attribution can be one of the links between product information and process corrections.

4. Process Attribution and Feedback

4.1 Introduction

The problems identified in Chapter Three compound the complexity of the manufacturing process. The high speed nature of the assembly process, the many assembly combinations, and the constant changes in assembly order are hard to track from the shop floor. Specific process variances are lost in the noise of production sequences. Process Attribution makes the noise of the process apparent and allows for extraction of specific process signals. In this Chapter, three examples of Process Attribution show a measure of specific process variances. Process Attribution and bar-coding are compared and the strengths and weaknesses discussed.

4.2 Process Attribution

Expanded Definition

Process Attribution refers to the marking of a product during manufacturing to make the specific assembly combination explicit. Process attribution creates a visual process record on the product. The defective product contains much of the contextual information regarding a defect. Keeping the process record with the product centralizes relevant information in one location.

In theory, making the assembly combinations observable will have two effects. First, with contextual knowledge of defects and explicit process information, operators can rapidly perform a root cause analysis. The identification of a root cause eliminates the need to diagnose the process. Second, operators observing that the circuit boards are not all the same can gain a better understanding of how the mix of boards changes over time. This understanding creates a reference set of useful experiences. Over time this experience can identify process improvements and reduce diagnostic time further.

While the majority of the workforce is not aware of the complexities identified in Chapter Three, some workers have adopted their own Process Attribution methods for visualizing the process. Individual methods for coding processed circuit boards (by pen marking) are being used by workers.

These methods of tracking work-in-process typically involve simple markings on the assembled electronic circuit board. Interviews with the operators identified two sources for this strategy, the prototype lab and engineering floor support. Because the prototype lab lacks an information system or automated board handlers, pen markings are used to track the product through the process (e.g., which processes are completed and by whom). In production, engineering floor support often requests operators to specifically mark production units to confirm visual inspection (e.g., presence of a fastener). As the workforce rotates through different jobs they continue to employ these marking techniques at new positions in the assembly operation. One operator explained "I need to mark the work-in-process to distinguish work completed and work waiting to be processed (e.g., x-ray test completed) so I don't make errors and so I won't be held responsible for something that I didn't do." By marking the circuit boards specific workers, x-ray testers, and rework can be identified. A straight diagonal mark down the side of a stack of circuit boards also makes the changing board order visible. Later when the circuit boards are re-stacked the marks no longer form a line. The concept of Process Attribution is further motivated by examples of experiments.

Example One: Process Attribution of Solder Paste Process

In an example of a solder paste Process Attribution, one operator was asked to draw a black mark down the side of a stack of bare circuit boards. This marked the edge of all boards in the stack. These boards were then run in production from one of two paste machines (Figure 4.1). The flow of these boards throughout the assembly line was discussed with operators at various stages.

Qualitatively the experiment was successful. Changes in the distribution of boards were noted by the operators, and the repair technicians could identify during diagnosis which paste machine supplied a defective circuit board. The repair technician identified that more defects were coming from one paste station. The defects where pointed out to the paste station operator who then corrected the paste screen registration. The normal delays to x-ray repair (Figure 3.6) where bypassed during this experiment.

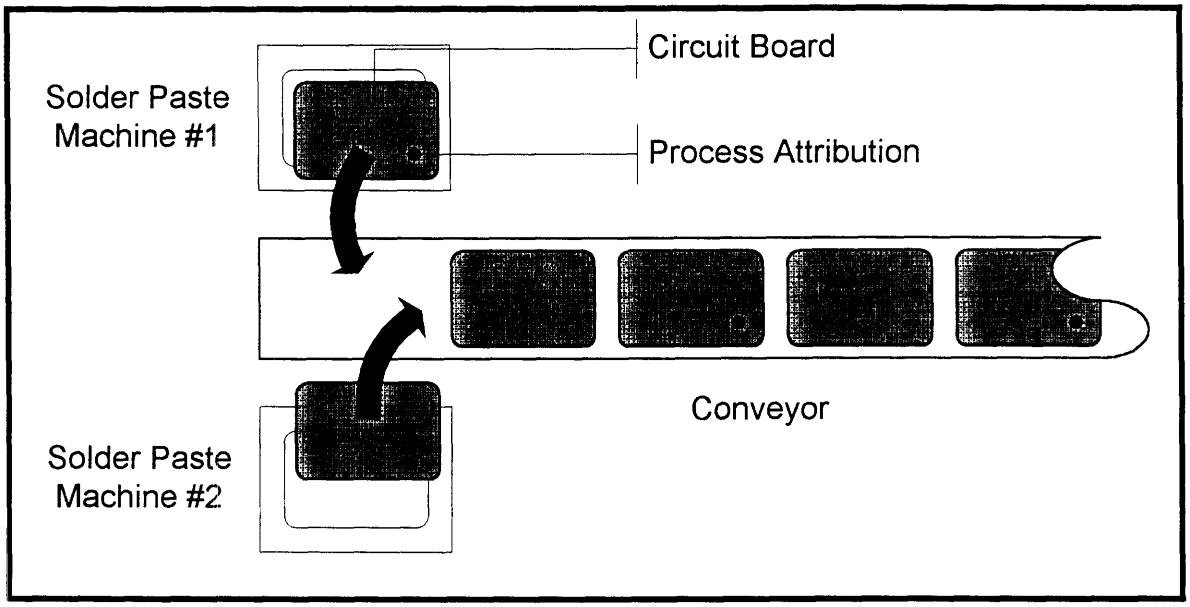

Figure 4.1 Example of process attribution at a solder pasting operation.

approach the end of their scheduled shift they consolidate the remaining circuit boards to reduce the number of handled circuit boards left over. This redistribution of marked circuit boards causes the last marked boards to come from both paste stations. This last minute shuffling defeats the purpose of the Process Attribution and worse, it may cause unwanted feedback. The conclusion drawn is that attributions should be applied by the machine as part of the process to reduce errors.

In the real case of flow consolidation at solder paste (Table 3.1), once test and repair identify defective products (Figure 3.7, loop **A**), it is not possible to resolve which of the two machines is producing the defects (Figure 3.7, loop **B**). Defects that are caused by solder paste include smearing, skipping, ragged edges, and misalignment. To identify the root cause requires inspecting solder paste viscosity; print thickness; squeegee wear, pressure and hardness; print speed and mesh tension at both stations. In practice this analysis reduces throughput of the solder pasting operation. The solution to closing the process feedback loop requires that each machine leave its "signature" on the boards it processes. This can be done by adding a reference hole in the solder screen of machine #1 so boards pasted at machine #1 have an additional identifying paste mark on the circuit board.

Example Two: Process Attribution of Board X Panel Position

Delco Electronics fabricates it's own circuit boards. The air bag controller circuit boards are produced on one large panel. Depending on the board size there are several panel configurations. Board X requires a three by five array (Figure 4.2). Numbers identify each board position within the panel.

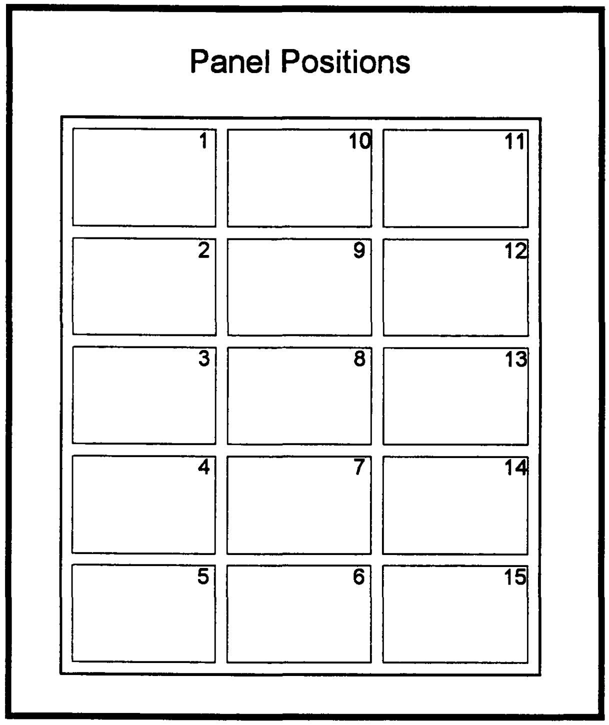

Figure 4.2 Circuit board panel positions.

Panel position was selected for Process Attribution for two reasons. First, useful Process Attribution already exists because position numbers are part of the board. Board fabrication has its own set of flows and consolidations that mix the board order. To provide feedback to the fabrication process, all board positions must be numbered. Second, the circuit board is a custom part (different for every product) and typically has lower quality levels than component parts which are standardized, increasing the likelihood that this experiment would detect a problem.

Circuit board defects include trace width and spacing, plating thickness variances, and solder mask quality. To understand if Process Attribution could identify any panel position-specific defects for board X (a specific process variance), repair technicians were asked to record the panel position of circuit boards identified as defective. Samples of board X were collected from in-circuit repair (Figure 2.1). As part of the data collection the technicians verified that the circuit board had not been repaired previously and that the circuit board was defective.

A representative sample ($n = 488$) of defective boards from in-circuit repair was collected over several weeks. The panel position of each defective circuit board was tracked with a checksheet. Each checksheet recorded panel position of defects for one shift. The in-circuit repair technicians were interviewed before, during, and after the data collection. During this period nothing outside of normal operations occurred in production.

Whenever process data is analyzed it is important to consider the relevant sub-groupings. Because data was collected at the shift level (finest grain) it can be analyzed for shift, day, week, or overall trends. The first question to ask is, "can Process Attribution identify panel position-related defects during a shift on which immediate corrective action could take place?" (best case scenario). From the author's perspective the answer is "no." While there were no significant board-related defects during the trial period, shift level data typically clumped randomly at various panel positions. Figure 4.3 shows data from a sample shift mapped to panel positions. A further analysis of the sample data is supplied in Appendix B.

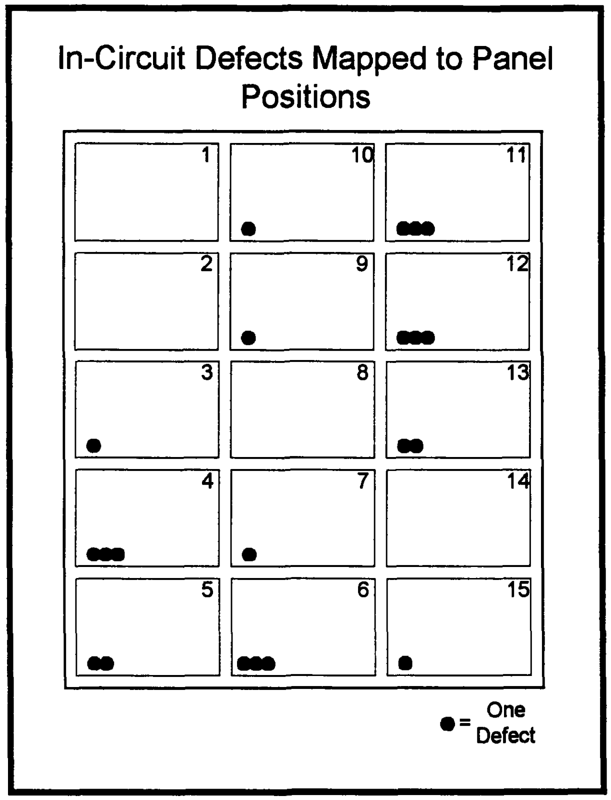

Figure 4.3 Sample defect data from one shift mapped to panel position.

However, when total panel defect data is mapped to panel position there is evidence of systematic effects. The cluster of defects located at the bottom of the panel (Figure 4.4) indicates a higher likelihood of failure for those circuit boards of panel positions 5, 6, 7, and 15. Process engineers responsible for production of the circuit board panels explained "lower panel positions often experience higher plating thickness variability." The Process Attribution data and observations of the plating process indicate a systematic circuit board problem. The effect of plating thickness variances on co-planarity and quality are documented by Prasad [9] and support this conclusion. Another observation made is that circuit boards in corner positions have higher failure rates. This trend analysis indicates that the circuit board assembly process is sensitive to variations in circuit board geometry. Put another way, the assembly process is not robust enough to overcome process variation in board fabrication.

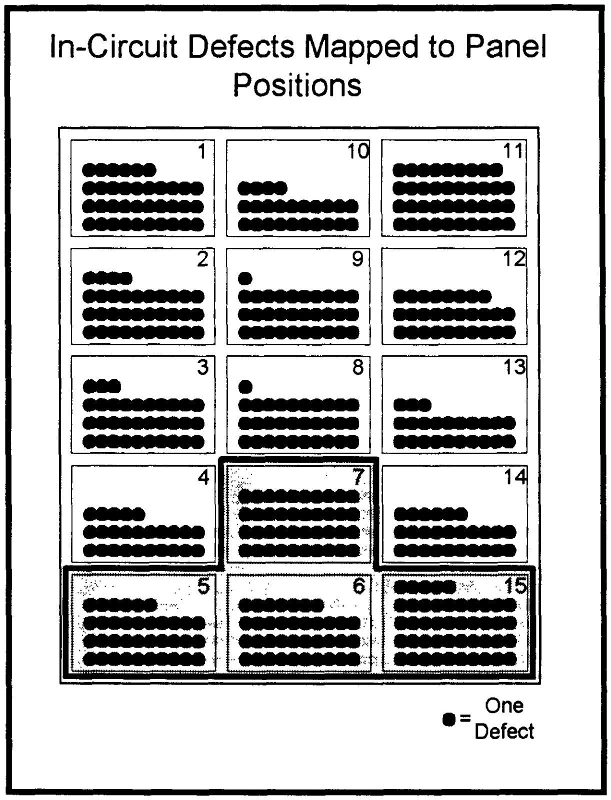

Figure 4.4 Total defect data mapped to panel position.

Example Three: Process Attribution by Pallet Position

During this same study another Process Attribution was implemented to identify the pallet position of defective circuit boards. The circuit boards are placed two at a time on pallets at thru-hole component insertion. Because board dimensions vary from product to product, pallets are board type specific.

Pallets travel down the thru-hole component insertion line as operators insert leaded components. The pallet controls the height of both circuit boards as they travel through the wave soldering process and in-circuit testing. Because the pallets fix the order of the boards traveling through the wave solderer, boards can be seen as trailing or leading in the process (Figure 4.5).

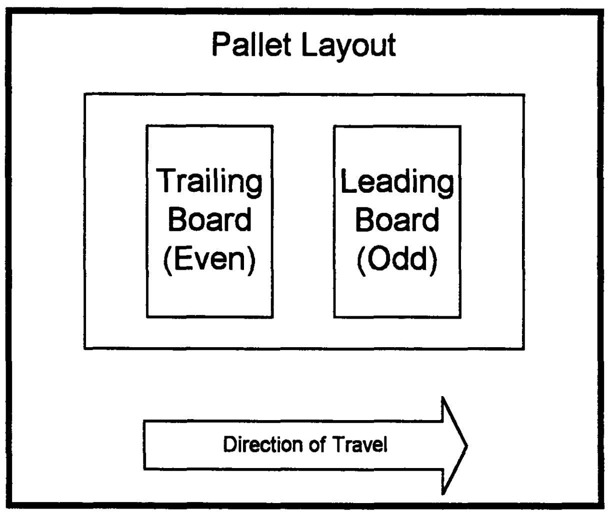

Figure 4.5 Leading and trailing board positions on pallet.

To identify if the positions of the board affected solder quality, the boards need to be identified as leading or trailing in the process. Because the boards are serialized prior to this operation, a simple way to identify the original pallet position is to place the bar-coded label on each circuit board so that the even numbered labels are on the trailing board and the odd numbered labels are on the leading board. Once the work instructions had been altered to provide this ordering, defect data by pallet position could be tracked.

The same representative sample ($n = 488$) of Board X can be evaluated for effects due to pallet position. No significant effects were identified ($\text{Even} = 246$, $\text{Odd} = 242$).

While it is unclear what was learned from this attribution, the Process Attribution clearly visualizes pallet position. The marking is simple enough to provide anyone with immediate information about the pallet position, and allows relevant comparisons of defects. The sample data are supplied in Appendix B.

4.3 What Process Attribution Provides

The preceding examples of Process Attribution provide solid evidence of specific process variances and help make several points.

- Making the manufacturing process visible with Process Attribution centralizes relevant data.
- Process Attribution visually records the specific mix of assembly combinations and stimulates learning through comparisons.
- Process Attribution links each defective circuit board, with all of its contextual information, to a specific process position (e.g., panel position or machine) and greatly reduces the confusing effects of multiple assembly combinations.

We will now discuss each of these points.

Centralizing Relevant Data

The value of Process Attribution is that the worker, first line supervisor, and engineer can at a glance connect product defects with specific processes. Sources of relevant data include test results, repairs performed, specific assembly combinations, and operational information about the product failure.

In practice, test results are provided to the repair technician, in the form of a receipt, with the defective board. These test results along with observations of the actual circuit board allow the technician to effect timely repairs. With the repairs completed the receipts are thrown away and the product re-enters production. While this procedure results in timely repairs of the defective product, much important information needed for process improvements is lost. Contextual information is not captured in the test data that resides in the information system. The contextual information about the failure (for example, the board was not seated correctly on the pallet when it arrived at in-circuit test) is important.

When process information (in the form of specific assembly combination) and the test results are provided with the product, much of the data needed to identify trends in the assembly process is available in one central location. By collecting contextual product defect information, data from test, and specific assembly combination, Process Attribution centralizes information. Centralizing the process and product information improves the chances that a root cause can be identified. For example, suppose that boards from panel position 15 develop an extraordinary string of failures. The panel number (process information) identifies that they are all from the same panel position, and any similarity in the actual defects (product information) helps root cause identification.

Learning Through Comparisons

In understanding the manufacturing process, relevant comparisons are important. The air bag controller assembly line is a unique combination of equipment and processes. Each product has special processing requirements. Comparisons made between assembly lines assume a parity that may not exist. Comparing differences in machines, materials, and methods is not easy. In the high volume production of air bag controllers Process Attribution allows comparisons among processes within one assembly line. With explicit process and product information at hand, operators can rapidly perform a root cause analysis based on meaningful comparisons. These meaningful comparisons provide information about the symptom and cause. The examples of Process Attribution allowed comparisons of product defects by solder paste machine, panel position, and pallet position. In Addition, Process Attribution provides a visual environment that stimulates learning. Through comparisons the technicians learned that not all boards are "created equal."

Reducing confusion

By using specific position information (Example Two and Three) workers could understand and talk about the manufacturing process problems. One technician commented "If I find a problem related to a specific board position it would be simple to explain to a shift supervisor." By improving awareness of the process problems and communication, Process Attribution reduces the confusing effect of multiple assembly combinations.

4.4 What Bar-Coding Provides

Because the contemporary means for tracking products is through the use of bar-coding, the possibility of using bar-coding to capture specific events should be considered. Bar-coding, as discussed here, refers to applying a bar-code and scanning the bar-code to collect and organize data in an information system. As seen from Example Three, the act of applying the bar-code can be a Process Attribution if there is a visibly distinguishable marking (e.g., serial number).

The data that is captured by Process Attribution (specific assembly combinations) can be collected through bar-coding and scanning of the bar-code. In addition, bar-coding can collect timing information related to specific processes. The combination of specific events and their timing creates a de-facto "product history." By itself, information systems might look like the best solution to the problem of multiple assembly combinations for two reasons. One, performing analysis off line impacts the production systems less. Two, the detail of the data might seem more than adequate to identify relevant process problems.

Though information systems capture the specific timing not captured by Process Attribution and allow off-line analysis, information systems do not always improve problem solving. From the practical position of solving problems on the shop floor, information systems can add complexity and separate process information from relevant product information.

The "mixed" electronic assembly process as outlined in Chapter Two is complex. It is out of this complexity that the problems of multiple assembly arise. Prasad [10] notes that "It (mixed assembly) is the most difficult assembly to manufacture because it has the most process steps." Adding the infrastructure to record bar-code data at each process increases the complexity of the assembly process. In addition to setting up and maintaining product and process flows, the operator or an additional worker has to manage information flows. While bar-coding is a solution for tracking problems, it does not make the production process visible. Because the bar-code data is coded and not directly readable by operators, understanding specific assembly combinations (the original problem) remains a challenge.

Separating the process data from the product causes much relevant information to be lost. Information systems have a finite data management capacity. While the information system can capture detailed process information, it is unable to capture the rich data that the actual defective circuit board contains. For instance, to assure quality during air bag controller assembly, in-circuit testers (Figure 2.1) send "pass–fail"data for every board to the network. However, the in-circuit testers send only a subset of the test data to the network. The subset includes failure data relating to every failed board and data on every 20th good board. This trimmed data set prevents overloading the information system at current production rates. Even with significantly larger data management capacity, much of the contextual information cannot be captured by the information system.

By capturing only part of the data, separation of process information from the product can mislead problem solvers. With only the test data, analysis performed off-line can confuse correlation (a trend) with causation (a cause and effect relationship). Causes identified by trends in the data amount to nothing if another variable, that has not been captured in the data, is the root cause.

What is needed to identify specific process problems is detailed product failure information and some process information. Once a specific process has been targeted for improvement, however, timing information is critical to understanding its performance over time. In this capacity a detailed "process history" adds value.

4.5 Process Attribution and Bar-Coding

Contemporary thinking is that bar-coding the product (scanning it at all process positions) is the solution for product tracking problems. While information systems can provide a detailed process history they cannot capture the rich defect data that the actual defective circuit board contains. Characterizing process variances requires understanding both specific product and process defects in detail. Because the quality of information decays as contextual information is lost, to identify specific process variances it may be more efficient to place some process information directly on the defective product. Once a specific source of variance has been identified, a detailed "process history" becomes important in further understanding why this variance exists. In this capacity, bar-coding and Process Attribution are complementary.

Visualizing the process through Process Attributions places the relevant process information with the actual defect. Information is accessible to anyone. Furthermore, all the relevant information is readily portable. Figure 4.6 summarizes the main differences between bar-coding and Process Attribution. Along the information axis bar-coding is stronger at collecting process information. Process Attribution is stronger at collecting product information. Bar-coding separates process and product information. Process Attribution centralizes process and product information.

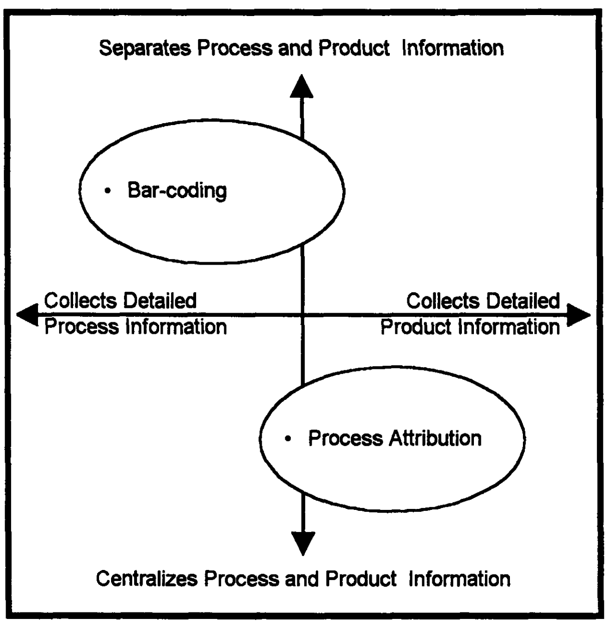

Figure 4.6 Strengths and weaknesses of process attribution and bar-coding.

4.6 Summary

Three examples of process attribution and the value of making the process visible have been presented. Process Attribution was shown to collect and organize relevant data and stimulate comparisons. Bar-coding as a method of collecting a detailed "process history" was discussed along with the complementary roles that bar-coding and Process Attribution can play in collecting information.

The practical value of simply organizing relevant data cannot be overstated. In recommending simplifying manufacturing operations Hayes, Wheelwright, and Clark [11] point out that "More information is not necessarily better than less; it may simply serve to confuse people. The real objective is to have the necessary information in the right place at the right time." By making assembly combinations explicit, Process Attribution places the necessary information in the right place at the right time.

5. Benefits of Process Attribution

5.1 Introduction

From Chapter Three we know that without real-time feedback, corrections to the process occur in two stages. The second stage requires diagnosis of the process to identify root causes of problems. Chapter Four provided examples of how Process Attribution organizes relevant data. This organized data stimulates comparisons that can rapidly identify the root cause of problems. This chapter will examine the costs and benefits associated with making the process visible, and present a model for understanding the value associated with providing real-time feedback. A simulation is performed and the findings are discussed.

5.2 Effects of Information on Throughput and Delays

Information about the manufacturing process directly affects throughput. When relevant data is not available, lengthy diagnosis must be performed to identify a root cause solution. In the high volume production of air bag controllers, process diagnosis reduces the throughput of the manufacturing system. Throughout this study the author observed the methods and effects of actual process diagnosis. Through discussions with the operators and engineers the effects of diagnosis on throughput were estimated. Table 5.1 organizes observed diagnostics and their estimated effect on throughput.

A primary concern for production system throughput is often the bottleneck operation. In the production of airbag control units the bottleneck operation is the top side and bottom side chip placement. Because the assembly line is closely balanced, however, most processes can easily become bottleneck operations. The estimated effects of diagnosis (Table 5.1) suggest that diagnostic work on any process can potentially reduce the entire production system throughput. For example, if one paste station is being diagnosed 40% fewer boards start the top side surface mount line.

One benefit of visualizing the production process with Process Attribution is increased throughput. Because of the complexity of the production system, problems occur. By linking the feedback loops (Chapter Three) corrective actions may be made with

Throughput Lost During Diagnosis

Process	Diagnostic Procedure	Throughput lost
Solder paste	Visually inspect paste screen and solder paste quality.	40% if only one machine is inoperative for inspection and the other increases throughput
Top side chip placement	Stop production and let machines clear all boards. Visually inspect boards that are run through as AA and BB combination.	100% of the surface mount line throughput
X-ray Test	Re-test circuit boards in both positions of tester and then reload the software. If problem persists, check calibration of tester.	50% of test throughput if one machine is down during diagnosis
Glue dye	Initially pick sample boards out after glue and inspect quality and placement of glue. If problems persist, stop production and inspect physical dye for defects.	10%–100% of surface mount throughput if problem persists
Bottom side chip placement	Stop production and let machines clear all boards. Visually inspect boards run through as AAA and BBB and CCC combination.	100% of the surface mount line throughput
Wave Solder	Initially parts and pallets may be inspected off line, but detailed diagnostics uses control (empty) pallets that travel through the process under careful observance.	0%–100% of thru-hole component placement throughput
In-Circuit Test	Typically only one tester is inoperative during diagnosis.	33% of thru-hole component insertion throughput

Table 5.1 Observed diagnostics and estimated effects on throughput.

reduced diagnostic work. Interrupting the production process less for diagnostic work increases throughput. Any time the production system throughput can be increased without the addition of capacity the unit manufacturing cost decreases. The simplest method of estimating the cost of diagnostic work is to consider every minute lost at a specific process as one minute less production.

Another benefit of making the production process visible with Process Attribution is reduced rework. If diagnosis is attempted at the first sign of trouble, less rework is generated. Delaying diagnosis of the process to collect additional information generates added rework. Any decrease in throughput or increase in rework raises the cost of manufacturing air bag controllers.

A Simple Model

A simple process model illustrates the cost of defect identification delays and diagnostic interference. Consider a process that is corrected after some defect identification delay and with some diagnostic interference, as illustrated in Figure 5.1. This model assumes that the correction after diagnosis is immediate and process repairs that are instigated by a mis-diagnosis are considered part of the diagnosis time. This simple process produces at a rate R and generates defects with some probability P. The process generates a defect with probability P_{ic} while in control and probability P_{oc} when out of control. This process has a defect identification delay time of τ_1 and diagnosis stops throughput for a

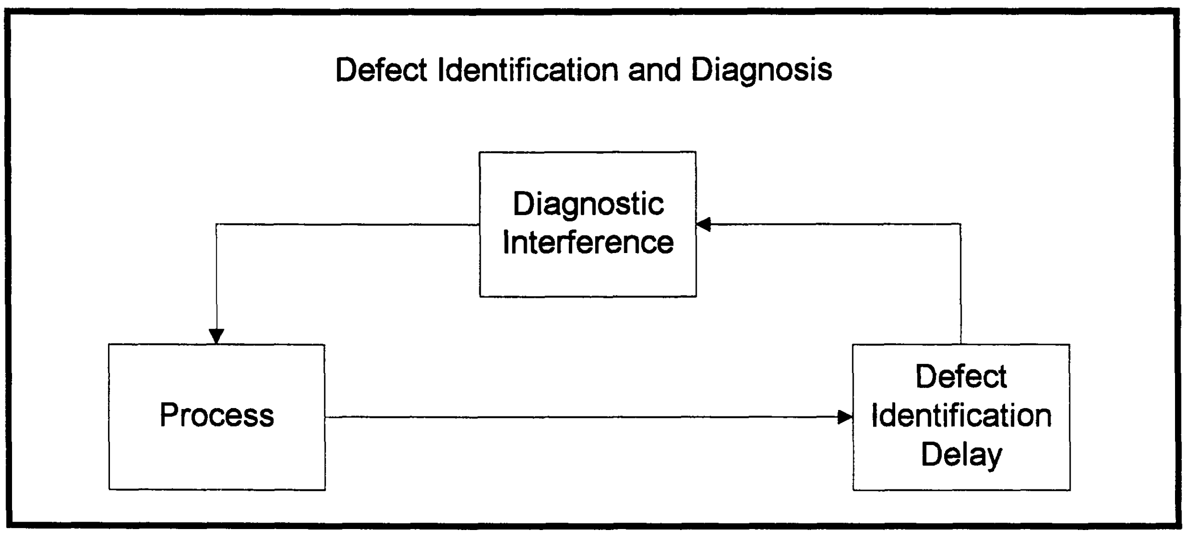

Figure 5.1 A simple model of defect identification and diagnosis.

time equal to τ_2. This model captures the steps in corrective action identified in Section 3.6. The cost of one out of control event (Cost_1) is the cost to rework each defective unit generated before detection $\$_{re}$ and the cost of each unit not produced during diagnosis $\$_{th}$. For the simple model, cost of one out of control event is:

$$\text{Cost}_1 = \underbrace{(P_{oc} - P_{ic})(\tau_1)(R)(\$_{re})}_{\text{Cost of additional rework}} + \underbrace{(\tau_2)(R)(\$_{th})}_{\text{Cost of lost throughput}}$$

From Chapter Four we know that Process Attribution provides relevant and timely information. Assuming that this reduces the time to detect problems (τ_1) and diagnostic interference time (τ_2) the total cost to correct a problem is reduced. Figure 5.2 diagrams Case #1 (without Process Attribution) and Case #2 (with Process Attribution). Depending on how much relevant feedback information is available different delays and process interference result.

The benefits from reducing defect identification delays and diagnostic interference

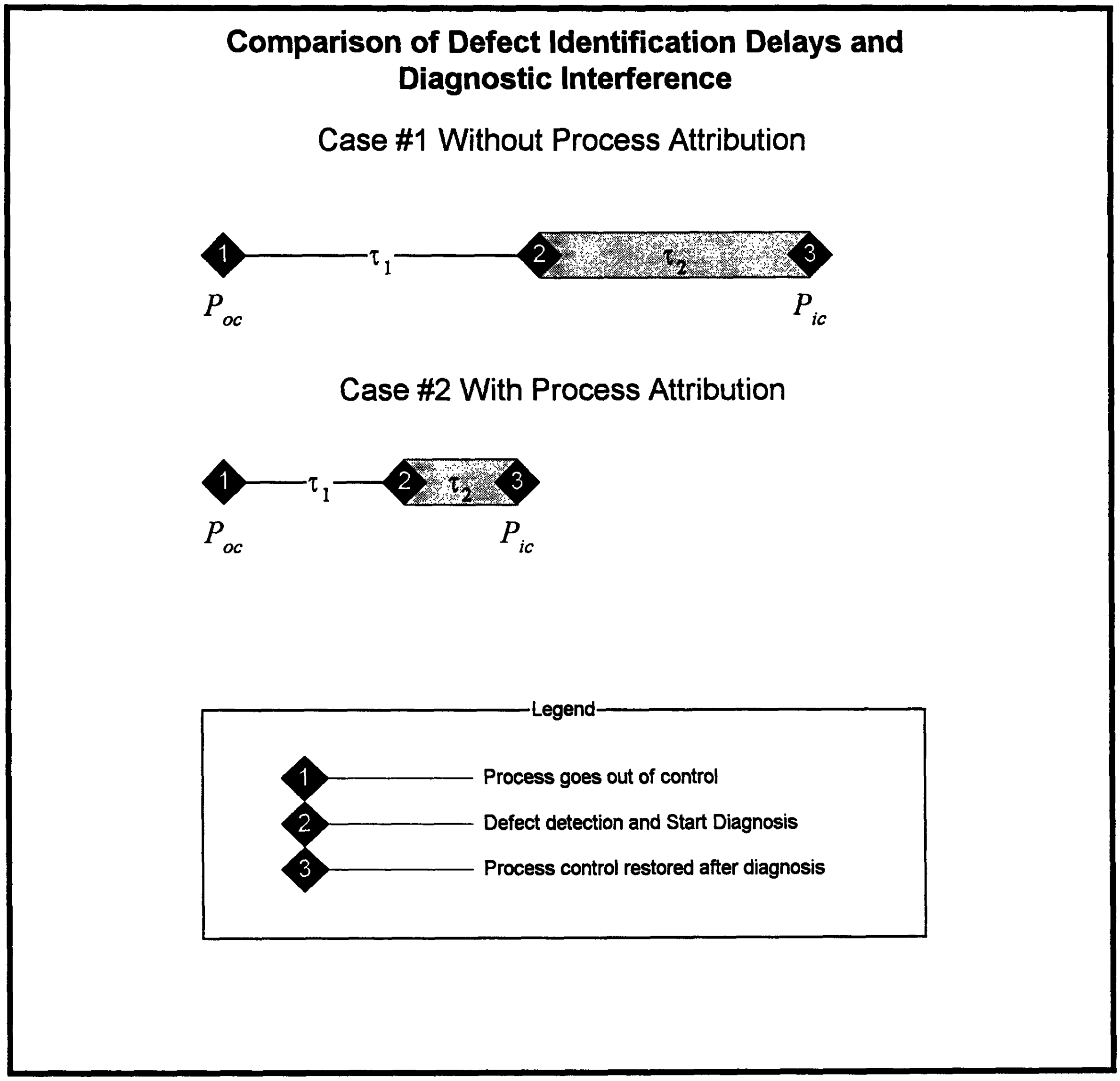

Figure 5.2 Comparison of defect identification delays and diagnostic interference showing reduced time to resolve problems with process attribution.

must be traded off against the costs of implementing Process Attribution.

5.3 Simulation of Top Side Chip Placement

To compare the two cases of identification delays and intervention (Figure 5.2) on a production system we revisited the Witness™ model of top side chip placement. Amending the model to include a rework station, allows a simulation of the two cases.

Assumptions

To compare throughput for several states of system reliability requires making some assumptions about the systems performance. These assumptions include:

- Reliability of the system
- Control limits for defect identification
- Rework rates
- Diagnostic intervention time (τ_2)

For this simulation, position B_2 (Figure 3.1) of top side chip placement machine #2 could fail. When B_2 failed it produced 60% defective boards (30% of machine #2 capacity). Position B_2 was assigned a failure rate based on an exponential ($\lambda = 1$) function. This function provides the "memoryless" property associated with component lifetimes. Trivedi [12] provides a complete treatment of the exponential ($\lambda = 1$) function. Once a failure occurred defective boards were produced until some corrective action was taken. Corrective action was initiated when the rework station exceeded its control limit.

The control limits were set at 75 defective units for Case #1 and 8 defective units for Case #2. The limit of 75 defective control units for Case #1 assumes that, even without Process Attribution, if defects pile up at rework they will be noticed. The control limit provides the defect identification delay (τ_1). The delays between test and repair documented in Chapter Three are assumed to be part of the delays to identification of defects and are captured in the control limits. The rework rate for Case #1 was assumed to be one board per minute (consistent with actual rework rates noted in Section 3.4). Because additional work is being performed by the repair technician (i.e., gathering process information) the rework rate for Case #2 was assumed to be 50% of the Case #1 rate.

Finally, diagnostic intervention time for Case #1 (15 min.) was assumed to be three times as long as the diagnostic intervention time of Case #2 (5 min.). This is consistent with the reasoning that Process Attribution allows reduced diagnostic times. With the above assumptions, the simulation was run for both cases and the results are shown in Figure 5.3.

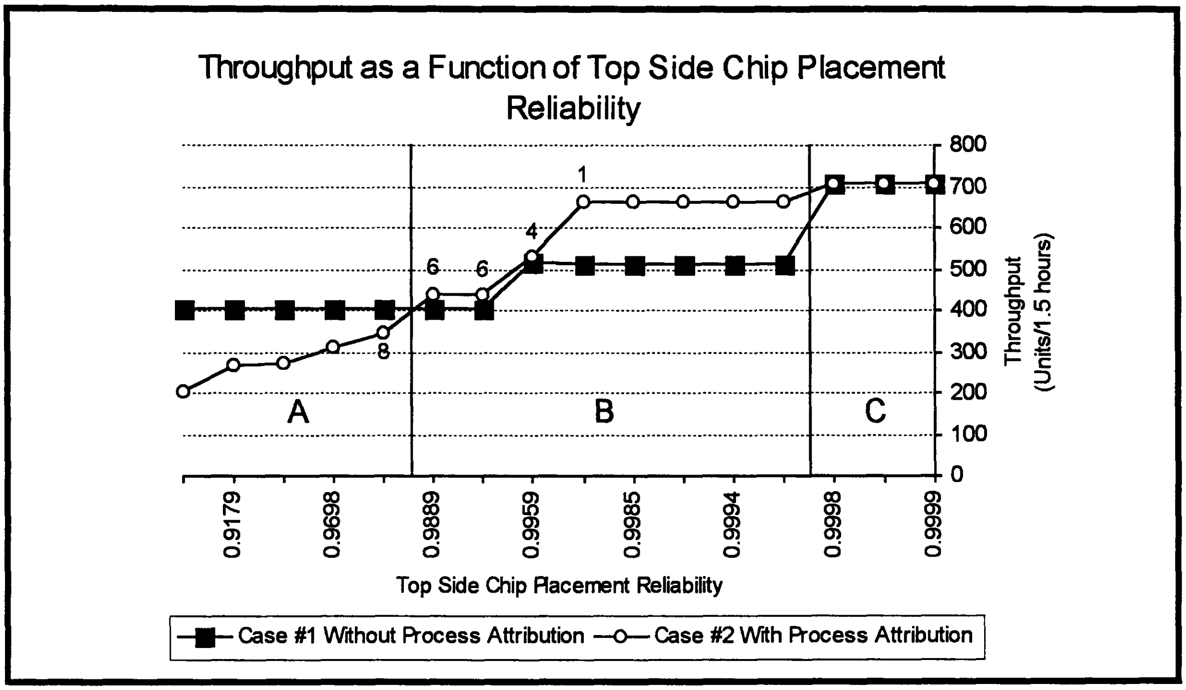

Figure 5.3 Throughput as a function of top side chip placement reliability.

Findings

The simulation results demonstrate that using the above stated assumptions, providing feedback increases throughput of the production system under **some** conditions. There are three distinct regions of performance. Region (B), characterized by reasonable reliability, shows that there are gains to throughput by taking corrective action sooner. Where process intervention is low (Figure 5.3 data from Case #2 is labeled with the number of line stoppages) throughput gains are substantial. Region (C), characterized by extremely high reliability, shows no improvement with Process Attribution. This phenomenon makes sense. If the system does not produce defects then process attribution is not useful. Region (A), characterized as low reliability, shows that throughput can actually be negatively impacted by too much process intervention. This was an unexpected result. The reduction of throughput shows that too much diagnostic intervention even with relevant information may be a bad thing. It is important to note, however, that the rework to be processed remained manageable (9 defects left to be processed). The regions defined by the simulation are a function of the assumed parameters. If the control limit for Case #2 (8 defective boards) are decreased then more interventions will occur and throughput will drop off faster. The simulation results are supplied in Appendix C.

There are other related benefits that stem from increased throughput. Increasing throughput of the manufacturing system can reduce work-in-process and shorten lead times. Uncertainty of throughput at any particular process causes inventories of partially assembled circuit boards to be kept on hand. When variance is reduced these inventory requirements shrink. The costs associated with holding these inventories (value of material and additional lead time) add to the cost of manufacturing. Increased throughput may increase available production time if order shortages require additional setups.

5.4 Further Improvements

If cycle times are constant then τ_1 represents work-in-process before test. Reducing the amount of work-in-process or moving tests closer to the source of defects, reduces τ_1. Reducing τ_1 reduces the delay between production of defect and detection. While reducing τ_1 is in line with improving quality by shortening defect identification time, without a feedback system it makes no difference if defects are identified sooner. The consensus is that there is no one optimal test strategy, but several possible alternatives. Bateson [13], for instance, does not suggest a single most efficient form of electronics testing, but rather a "product" specific method. But, Delco Electronics' need for high volume restricts the choice of testers and their layout.

Standard test equipment has been developed for certain process locations. The design of in-circuit and functional testers determine their location in the process. In-circuit is designed to test assembled and soldered circuit boards for electrical flaws. Functional test is designed to test cased circuit boards for system functionality. These testers are not easily moved without eliminating certain processes. While types of automated vision systems could be placed closer to individual assembly processes, the value added by additional testing is zero if no real-time feedback is available.

The quality assurance requirements of air bag controller production have resulted in the development of a substantial test infrastructure. The marginal cost of using this data for process feedback is small. What is needed is a method for feeding back this information to the process.

5.5 Learning effects

Another benefit identified in Chapter Four was the ability of workforce to learn about the process. By making the process visible the workers are able to understand the normal flow of work. This knowledge provides a reference set of experiences that, over time, provide insight into process improvements. While τ_1 is fixed by cycle time and work-in-process, making the process visible would allow preemptive actions by the worker. With the additional information there is a concern about over controlling a process (or over interfering with the process as in Region (A) of the simulation), but actions based on clear feedback provide the opportunity to learn.

5.6 Summary

As part of a testing and feedback strategy, making the production process visible is important because it provides needed feedback to the process. By providing feedback Process Attribution reduces rework and increases throughput of the production system. An example of improved throughput at top side chip placement has been provided along with the insight that if process reliability is low Process Attribution feedback may be counter productive to throughput. Process Attribution can reduce the cost of manufacturing air bag controllers by identify defects and reducing process diagnosis if the production system is reasonably reliable. Additional savings result from reduced rework, scrap, and work-in-process.

6. Wider Implications for Process Attribution

6.1 Introduction

In Chapter Four we showed that making the process visible provides meaningful feedback. A source of feedback by itself will solve only local problems that are targeted for improvement. For feedback to be an asset, the workforce must adopt a *change culture*, where everyone can take responsibility for making improvements. Having identified four problems that inhibit feedback (Chapter Three), it is important to examine the wider organizational effects that result from this lack of feedback. There are at least four observable organizational artifacts that result:

1. Worker self control of problem solving is reduced
2. Analytical and operational work are separated
3. Large delays in repair are tolerated
4. Workers resist process diagnosis

The purpose of examining these artifacts is to understand how Process Attribution can contribute to the organizational shift needed to bring about a *change culture*.

6.2 Worker Control

The quality improvement programs of Deming, Juran and Crosby differ in decision tools and rules. However, they all require participative management and worker involvement. [See Fine (1987) for a comparison of quality programs and Juran (1988), Deming (1982) and Crosby (1979) for specific details.] Worker involvement is required to close the feedback loop from data analysis to process improvement. Any feedback that is directed toward process improvements depends on worker self control. Juran [14] divides control into two categories: worker and management. Worker control is said to exist if three essential criteria are met. These criteria are, providing workers with a means for:

1. knowing what they are supposed to do,
2. regulating the process,
3. knowing what they are actually doing.

Without these three criteria, improvement activity rests with management. Delco Electronics' workers receive extensive training about process operation, maintenance, and setup. Workers are expected to regulate the process. Managers must provide the workers with complete information to use in process control and improvement. By making the process explicit, workers can understand the performance of their operation. Visualizing the process provides a means of overcoming problems 1–4 of Section 6.1. By providing the third criteria, Process Attribution gives the worker self control. In example two, a trend was identified because checksheets allowed workers to know what they were doing. Visualizing the process brings problem solving into the domain of worker self control.

6.3 Analytical and Operational Activities

In high volume production of air bag controllers, knowledge and information are separated. Workers know how to operate the equipment and how to perform setups. The engineering support staff know why the equipment is selected and why it is important to balance the assembly line. Testing is the engineering domain and the diagnosis is the repair technicians domain. Because the process is not explicit, knowing why does not easily translate into knowing how (and vice versa). Figure 6.1 shows the operational–analytical

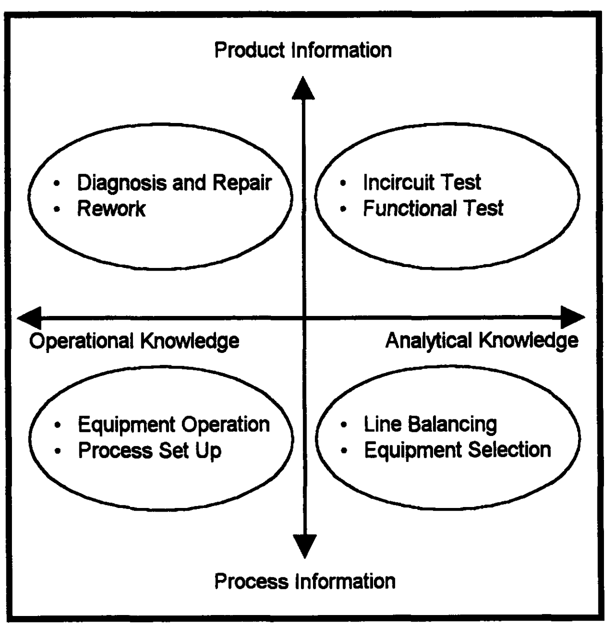

Figure 6.1 Separation of knowledge and information in the production of air bag control units.

separation. The operational–analytical separation impairs feedback and the development of problem solving skills. Klein [15] advocates linking these two work areas (operational and analytical) for continuous improvement and learning. Non-transparency of the process keeps the operational world of "how to" separated from the analytic world of "why." While discussing benefits of subject matter (operational) knowledge, and strategy (analytical) knowledge, Box [16] echoes this concern for integrating both types of work "…one does best (solving problems) by using both subject matter knowledge and strategy."

Visualizing the process provides for a *change culture* by integrating the operational–analytical activities of production. Physically marking the product allows specific assembly combinations to be seen and understood by everyone. In this capacity, process attribution connects the "how-to" with the "why." The common language of process understanding supports a *change culture*.

6.4 Delays in Repair

The large delays in repair, referred to on page 28 are tolerated because information gathered does not correlate with specific processes. The repair activity is seen as not adding value to the production process. It is hard for workers to see bottom line contributions of test and repair activities as anything other than increasing daily output. Repair technicians have no reason to think that one control unit is different from any other. This promotes a random repair order amongst the defective units. When all the control units appear identical, the particular repair order is unimportant. Here again visualizing the process makes every board different. These differences provide an incentive for organizing repairs in a First-In-First-Out order.

6.5 Resistance to Process Diagnosis

The difficulty posed in identifying a root cause of operational problems induces the workers to resist process diagnosis. Workers are reminded of their impact on the company's bottom line every day. A good job equals "product rolling out the door." Workers trade off the cost of lost production due to process diagnosis against the cost of additional rework. Because of the pressure to produce and the obvious impact diagnosis has on the production systems, operators favor producing additional rework.

Diagnosis is rarely initiated when individual products are found defective. This happens for two reasons. First, the lack of relevant information makes diagnosis risky. Without clear feedback mechanisms operators are not interested in trying to diagnose the process and risk lost production. Second, there is no way to predict the magnitude of the problem. Many process problems appear to generate only a few additional defects as they drift from in-control to out-of-control. The actual defects are mixed throughout the process and, as one worker pointed out, these defects arrive in "dribs and drabs," at the tester. Investigations only start when problems persist or when a batch of defectives arrive at a tester. As problems persist, the record of defective units is captured in the information system. However, by the time diagnosis begins much of the important contextual data is no longer available.

Not only do workers resist diagnosing themselves they resist others trying to diagnose. For example, staff are not encouraged to run experiments on the production floor. When maintenance or engineering support is called in to identify a problem, workers fuss about the fact that they will produce less if any testing is performed on the process. Typically, they try to negotiate the diagnosis away from their shift.

6.6 Problems of Implementation

While changing the manufacturing process to enable visualization is a direct physical change that can be observed, the organizational structures that have evolved because of the problems take a longer term view to correct. Every strategy should begin by taking stock of the current situation. For "organizational change" to be implemented Beckhard [17] suggests identifying key players and assessing their current commitment to change. His approach uses a "commitment chart." The commitment chart identifies the critical mass group, and each individuals' current level of commitment. Figure 6.2 shows an example of one possible commitment chart. This chart identifies levels of commitment that need to change (increased and decreased), and creates a map of those changes. Change may not require that everyone be at the "make it happen" level, but test and repair

	No commitment	Let it happen	Help it happen	Make it happen
Plant Manager			X →	O
Shift Supervisor		X →	O	
Repair Technician A			O ←	X
Repair Technician B				XO
Repair Technician C	X →	→	O	
Operator A		XO		
Operator B		XO		
Engineering Support A		X →	→	O
Engineering Support B				XO

X = Current commitment level   O = Target commitment level

Figure 6.2 Example of a commitment chart.

personnel are key individuals. If the needed change is going to occur, repair operators must have an understanding and ownership of the upstream processes.

This thesis provides examples of feedback problems and asserts that Process Attribution is one solution. Collecting data for the Process Attribution examples identified the depth to which change affects the organization. For example, to implement the bar-coding of pallet positions as odd or even required working with the operators at thru-hole as well as many of the other line operators. This apparently simple change in work instructions for one operator quickly identified that 25% of the workers are indirectly affected by this one change.

6.7 Summary

Visualizing the process is a form of feedback, that supports the broader organizational issue of implementing a *change culture*. It does this by providing for worker self control; integrating analytical and operational activities; generating incentives for reducing repair delays, and for diagnosing the process. Because Process Attribution addresses both the technical problem of multiple assembly combinations and the resulting organizational behaviors, it is an elegant solution for characterizing variance in the high volume production of air bag controllers.

7. Conclusion

While there is no formal research specifically addressing "Process Attribution," visualizing the manufacturing environment has been documented as a method for motivating continuous improvement. Greif Shimbun and Imai have written about visualizing the manufacturing environment. They refer to numerous examples of inventory control, KanBan systems, and visible management. In his book, "Kaizen," Imai [18] defines Visible Management as "the technique of providing information and instruction about elements of a job in a clearly visible manner so that the worker can maximize his productivity." Shimbun [19] defines visual controls in the manufacturing workplace as follows:

> "Visual control is a type of control that will enable even persons such as the company president, or other upper-level executives who know very little about the plant, to apprehend a certain amount of important information about the plant (namely, the progress status of the manufacturing processes, the amount of raw materials and work-in-progress being held in inventories, the number of defects being generated, which machines and equipment are out of production and why, and the like) merely by walking through the plant and observing it; this, in turn, will allow executives to point out problems and make suggestions concerning how to deal with them."

These authors demonstrate the value added by visualizing the manufacturing environment at the macro level. This thesis addresses visualizing the production process at the micro (work cell) level. Process Attribution helps the workers, first line supervisors and engineers understand the production process. While "visual controls" increase the workers consciousness of problems and costs, Process Attribution raises the workers awareness of the process. As a visual tool, Process Attribution is a subset of the broader range of visual controls (Figure 7.1).

This thesis demonstrates that feedback problems exist, that these feedback problems raise the cost of manufacturing and that Process Attribution is a simple method of providing feedback. Using Process Attribution we identified out of control processes (Section 4.2 Example One) and systematic circuit board problems (Section 4.2 Example Two). Process Attribution allows problems to be solved rapidly by explicitly identifying

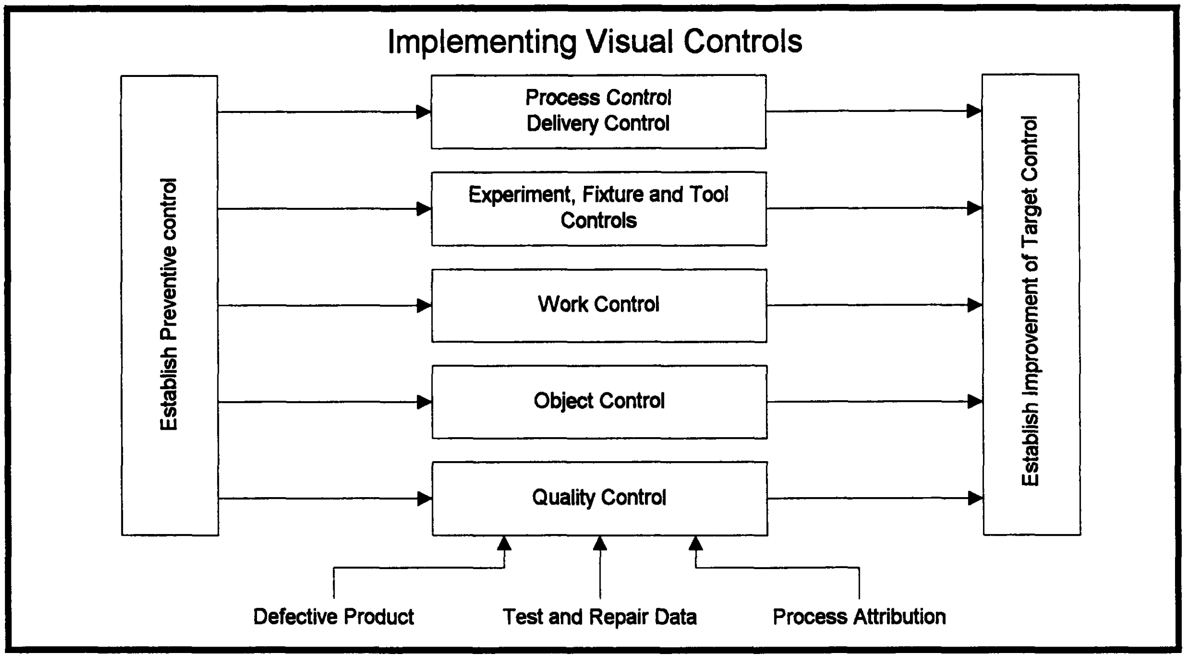

Figure 7.1 Steps of visual control implementation showing how defective product, test and repair data, and process attribution are part of visual quality control.

specific assembly combinations, which provides timely, accurate and relevant data. Understanding the process visually benefits any manufacturing environment where lengthy diagnostics reduce throughput.

This thesis is about understanding and simplifying process and information flows. The problems outlined in this thesis will be overcome as the pressure for reduced cost continues to increase. The solution will involve either redesigning the products, processes and equipment; implementing an information system based solution; or visualizing the process. The eventual solution may be some blending of these three. The learning from this work is that simple visual methods of transmitting information are one possible solution for providing real-time feedback.

It is important to consider how the changing product will influence information and process flows. For instance, as products get smaller, the assembled electronic circuit board will also shrink. How will this effect process and information flow? If smaller products are assembled on a single board and then separated at some point in the process, the point and method of separation is critical to process and information flows. The point and method of separation can either help provide real-time feedback or hinder it.

References

[1] Shawn, B., "State Farm Passive Restraint Study," State Farm Insurance Companies, Feb., 1992.

[2] Stubee, E., "Implementing Automated Statistical Process Control in Electronics Assembly," Master's Thesis, MIT 1993, pg. 87

[3] Deming, W.E., "Quality, Productivity, and Competitive Position," Massachusetts Institute of Technology, Massachusetts, 1982, pg. 4.

[4] Juran, J., "Juran's Quality Control Handbook," McGraw Hill, New York, 1988, pg. 17.4.

[5] Prasad, R., "Surface Mount Technology," Van Nostrand Reinhold, New York, 1989, pg. 9.

[6] Law, Averill M., "Simulation Modeling and Analysis," McGraw Hill, New York, 1991, pg. 114.

[7] "Witness release 6.0," AT&T ISTEL Limited, 1994.

[8] Kune, H., "Statistical Methods for Quality Improvement," AOTS, Japan, 1985, pg. 198.

[9] Prasad, R., "Surface Mount Technology," Van Nostrand Reinhold, New York, 1989, pg. 523.

[10] Prasad, R., "Surface Mount Technology," Van Nostrand Reinhold, New York, 1989, pg. 10.

[11] Hayes, Wheelwright and Clark., "Dynamic Manufacturing," Free Press, New York, 1988, pg. 204.

[12] Trivedi, K., "Probability and Statistics with Reliability, Queuing, and Computer Science Applications," Prentice Hall. New Jersey, 1982, pg. 114.

[13] Bateson, J., "In-Circuit Testing," Van Nostrand Reinhold, New York, 1985, pg. xiii.

[14] Juran, J., "Juran's Quality Control Handbook," McGraw Hill, New York, 1988, pg. 17.4.

[15] Klein, J., "Maintaining Expertise in Multi-Skilled Teams," in Advances in Interdisciplinary Studies of Work Teams, JAI Press Inc., 1994, pg. 145-165.

[16] Box, G., "Statistics For Experimenters," John Wiley & Sons, New York, 1978, pg. 6.

[17] Beckhard, R., "Organizational Transitions," Addison Wesley, Massachusetts, 1987, pg. 91.

[18] Imai, M., "Kaizen: The Key To Japan's Competitive Success," McGraw Hill, New York, 1986, pg. xxv.

[19] Shimbun, N., "Visual Control Systems," Productivity Press Oregon, 1995, pg. 4-5.

Appendix A Simulation and Experiment Data

Modeling the Machine Logic

To effectively model the simple case of two surface mount machines the exact machine logic must be understood. The author interviewed operators and support engineers to establish the basic machine logic.

An important part of the surface mount machine logic is the "load-unload cycle." The surface mount machine unloads two processed boards and loads two bare boards simultaneously. Finished boards wait until two bare boards arrive before unloading. If a delay time elapses, however, the surface mount machine will unload the finished boards and load only one or no boards. The loading of one or no circuit boards allows completed boards to move onto the next process (e.g., final circuit boards in a batch).When either machine loads only one board the assembly sequence changes and boards are assembled in new combinations. The basic machine logic (represented in Figure 1) was reached after several iterations.

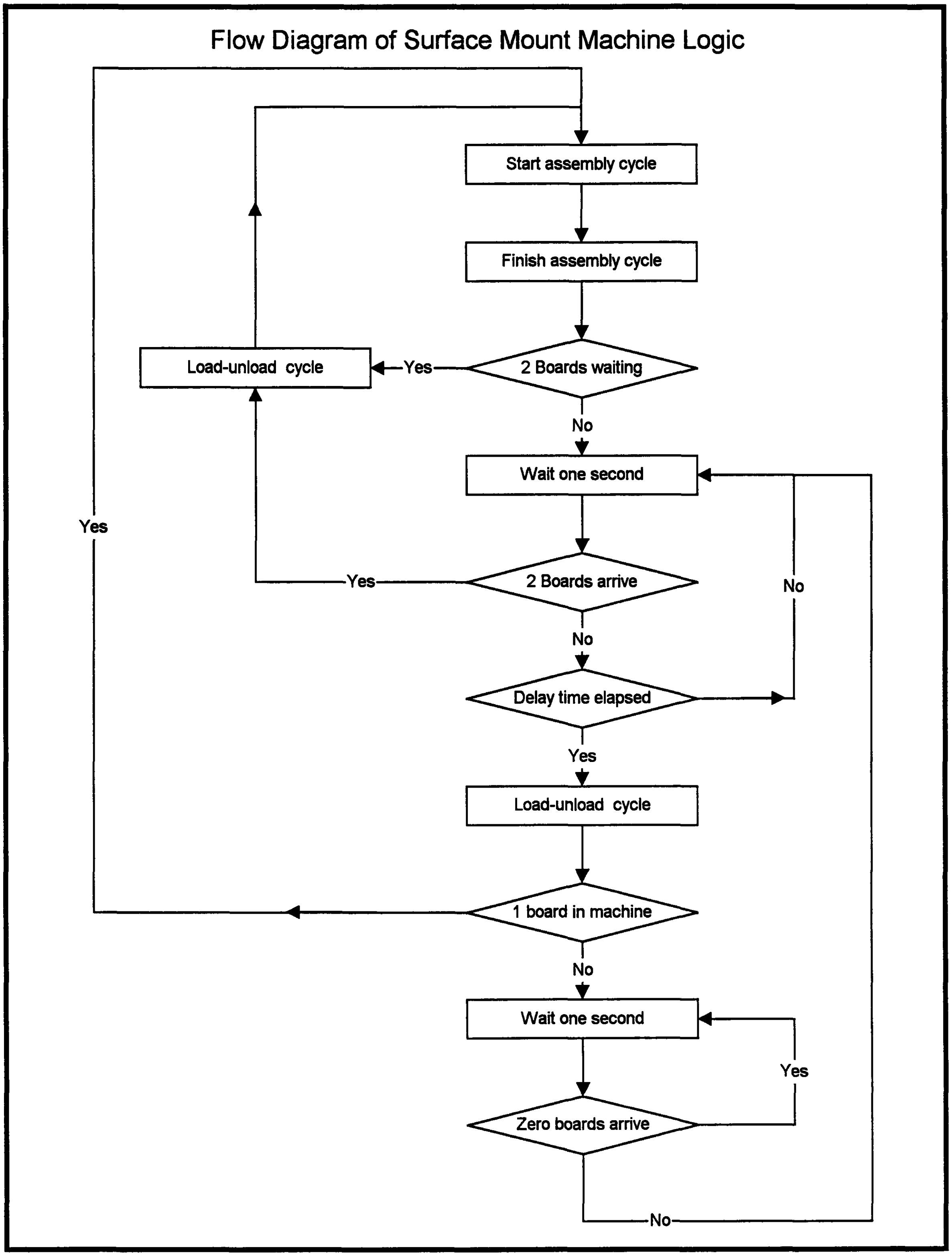

Figure 1 Surface Mount Machine Logic Diagram

Once the basic machine logic was understood and agreed upon the simulation model was built. Since Witness™ simulation software does not directly support the "load-unload" logic the author built the logic using 13 elements. The following list defines those 13 Witness™ elements necessary to simulate the behavior of a real surface mount machine:

```
PART: board,Variable attributes;
```

This represents the circuit board in the simulation. It has the attributes of machine position for each machine and board number for panel position attribution.

```
MACHINE: SMT_RSS1,1,Batch,0,0;
```

Defines the surface mount machine with batch size = 2. Input is controlled by the function "load1" and output is controlled by function "unload1."

```
FUNCTION: unload1:Name,0,;
```

The unload function sets flag1=1, which starts timer1, and waits for two parts to arrive in b1. The unload function returns the correct element to push to depending on part type. Scrap if part = dummy, and conveyor if part = board.

```
FUNCTION: load1:Name,0,;
```

The load function will check for two parts in the buffer b1. If zero parts then return none (equal to wait). If one then start timer1 and return none. If two parts then check for both being dummy parts, if so then start dummy1b (using flag1b) to pull the parts out. If two parts are not both dummy parts set flag1c and return b1. Setting flag1c allows SMT_RSS1 to continue loading parts.

```
BUFFER: b1,1,2;
```

Defines the input buffer to "SMT_RSS1" and has a capacity of two parts. The buffer is passive and has parts pushed in and pulled out of it.

```
MACHINE: timer1,1,Single,0,0;
```

This machine will pull parts out of the world and scrap them in one second cycle times. It is started when flag1=1, and stopped when flag1=0 actions on finish include counting to 19 and then setting flag1a=1

```
VARIABLE: flag1,1,1,Integer;
```

Flag1 is monitored by timer1. It is reset only by timer1. No reporting. It is set to 1 by unload1 and 2 by load1

```
VARIABLE: flag1a,1,1,Integer;
```

Flag1a is monitored by dummy1a and is set to 1 by timer1 and is reset to zero by dummy1a.

```
VARIABLE: flag1b,1,1,Integer;
```

Flag1b is monitored by dummy1b and is set to 1 by load1 and reset to zero by dummy1b.

```
VARIABLE: flag1c,1,1,Integer;
```

Flag1c is monitored by SMT_RSS1 and is set to 1 by load1 and reset to zero by SMT_RSS1.

```
VARIABLE: cnt1,1,1,Integer;
```

This variable is used to count completed boards from timer1 (equivalent to seconds). It is reset to zero only by timer1.

```
MACHINE: dummy1a,1,Batch,0,0;
```

This machine will pull two dummy parts from world after the "delay time," process them and push them to b1. On finish it will turn itself off by setting flag1a=0

```
MACHINE: dummy1b,1,Batch,0,0;
```

The purpose of this machine is to remove two dummy parts from b1 if both parts are dummy parts. The two dummy parts must arrive in b1 so SMT_RSS1 unloads, but they should not be loaded into SMT_RSS1.

Experimental Data

The simulation was run with three trials at each α. Each trial seeded the random number generator with a new number to remove any effects due to the random number stream. The experiment results are tabulated below.

	Number of (A_1,B_2) and (B_1,A_2) Boards				
α	Trial 1	Trial 2	Trial 3	Average Fraction	Average Total Boards
.23	0	0	0	0.00	694
.30	0	0	0	0.00	546
.36	164	36	298	0.37	446
.43	82	23	80	0.16	377
.50	39	24	47	0.11	325
.56	30	41	30	0.12	288
.63	22	20	17	0.08	257
.70	6	4	14	0.03	232
.76	4	2	2	0.01	213
.83	6	0	6	0.02	195
.90	8	7	8	0.04	180
.96	9	2	4	0.03	168
1.0	2	4	0	0.01	158
1.1	0	2	2	0.01	147
1.16	0	0	0	0.00	139
1.23	0	0	0	0.00	131

The Witness™ Model

```
! WITNESS MODEL: COMBINE

  * Title  : Surface Mount
  * Author : Anthony Reese
  * Date   : Sat Apr 29 21:57:32 1995
  * Version: WIN-207 Release 6.0


DEFINE

  PART: board,Variable attributes;
  PART: dum,Variable attributes;
  ATTRIBUTE: BRD_NUM,1,Integer,1;
  ATTRIBUTE: SMTPRSS1,1,Integer,1;
  ATTRIBUTE: SMTPRSS2,1,Integer,1;
  BUFFER: Board_AA,1,1000;
  BUFFER: Board_BB,1,1000;
  BUFFER: Board_AB,1,1000;
  BUFFER: Board_BA,1,1000;
  BUFFER: b1,1,2;
  BUFFER: b2,1,2;
  CONVEYOR: CNV_RSS1,1,Queuing,11;
  CONVEYOR: CNV_RSS2,1,Queuing,4;
  CONVEYOR: CNV_RSS3,1,Fixed,4;
  VARIABLE: flag1,1,1,Integer;
  VARIABLE: flag1a,1,1,Integer;
  VARIABLE: flag1b,1,1,Integer;
  VARIABLE: flag1c,1,1,Integer;
  VARIABLE: flag2,1,1,Integer;
  VARIABLE: flag2a,1,1,Integer;
  VARIABLE: flag2b,1,1,Integer;
  VARIABLE: flag2c,1,1,Integer;
  VARIABLE: cnt1,1,1,Integer;
  VARIABLE: cnt2,1,1,Integer;
  FUNCTION: unload1:Name,0,;
  FUNCTION: unload2:Name,0,;
  FUNCTION: load1:Name,0,;
  FUNCTION: load2:Name,0,;
  FUNCTION: load4:Integer,0,;
  MACHINE: SMT_RSS1,1,Batch,0,0;
  MACHINE: SMT_RSS2,1,Batch,0,0;
  MACHINE: dummy1a,1,Batch,0,0;
  MACHINE: dummy2a,1,Batch,0,0;
  MACHINE: dummy1b,1,Batch,0,0;
  MACHINE: dummy2b,1,Batch,0,0;
  MACHINE: timer1,1,Single,0,0;
  MACHINE: timer2,1,Single,0,0;

END DEFINE


REPORT_MODE ON_SHIFT_TIME GRAPHICAL STANDARD

DISPLAY
  OPTIONS
    TIME_SCALE FACTOR : 1.00,Off;
    WALK_TIME : Slow;
    TIME INCREMENT : 1;
    BATCH INCREMENT : 10;
  END OPTIONS

  DEFAULTS
    NAME COLOR: White;
    BACKGROUND COLOR: Black;
    TEXT SIZE: Standard;
    PART DISPLAY SIZE: 1;
```

```
  LABOR DISPLAY SIZE: 1;
  VEHICLE DISPLAY SIZE: 1;
  CONVEYOR: GAPS: 96,....;
  TRACK: GAPS: 16,....;
  MACHINE: GAPS: 45,....;
END DEFAULTS

KEY
END KEY

SCREEN 1
END SCREEN

SCREEN 2
  TEXT     : 48,32,White,32,MODULE 7643 ;
  BOX      : 776,368,776,368,Red,28,0;
  BOX      : 640,184,640,184,Red,28,0;
  ICON     : 2392,232,34,Yellow,257,0,0;
  BOX      : 2824,472,2824,472,Grey,28,0;
  BOX      : 2496,256,2544,264,Red,28,0;
  BOX      : 2416,256,2464,264,Red,28,0;
  BOX      : 2576,256,2624,264,Red,28,0;
END SCREEN

SCREEN 3
END SCREEN

SCREEN 4
END SCREEN

SCREEN 5
END SCREEN

SCREEN 6
END SCREEN

SCREEN 7
END SCREEN

SCREEN 8
END SCREEN

SCREEN 9
END SCREEN

CLOCK
  UNIT : Seconds;
  MULTIPLE : 1,Time ,60,0;
  MULTIPLE : 2,Day  ,24,1;
  MULTIPLE : 3,Week ,7,1;
  RATIO : 1:1;
  DISPLAY : USER;
END CLOCK

WINDOW_TITLES
  TITLE : 1,7643 Layout View
  TITLE : 2,TOPSIDE SURFACE MOUNT
  TITLE : 3,BOTTOMSIDE SURFACE MOUNT
  TITLE : 4,STICKLEAD
  TITLE : 5,Window 5
  TITLE : 6,Window 6
  TITLE : 7,Window 7
  TITLE : 8,Window 8
  TITLE : 9,Designer Elements
  TITLE : 10,Designer Elements Display
END WINDOW_TITLES

LAYER_STATUS
  LAYER : 0,On,Simulation Layer
  LAYER : 1,On,Layer 1
```

```
    LAYER : 2,On,Layer 2
    LAYER : 3,On,Layer 3
    LAYER : 4,On,Layer 4
    LAYER : 5,On,Layer 5
    LAYER : 6,On,Layer 6
    LAYER : 7,On,Layer 7
    LAYER : 8,On,Layer 8
    LAYER : 9,On,Layer 9
  END LAYER_STATUS

  BAR_SELECTOR_POSITION : 6,93,140,126;

  LIST_SELECTION_FORM_POSITION : 325,122;

SELECT

  board

  STYLE: Icon,6,1;
  PART: 64,248;

  END board

  dum

  STYLE: Icon,6,51;
  PART: 120,272;

  END dum

  Board_AA

  NAME: Standard,White,88,424;
  PART: Count,White,0,8,3,All,112,408;

  END Board_AA

  Board_BB

  NAME: Standard,White,176,424;
  PART: Count,White,0,8,3,All,200,408;

  END Board_BB

  Board_AB

  NAME: Standard,White,272,424;
  PART: Count,White,0,8,3,All,296,408;

  END Board_AB

  Board_BA

  NAME: Standard,White,352,424;
  PART: Count,White,0,8,3,All,376,408;

  END Board_BA

  b1

  BUFFER ICON: Status,95,240,152,1,1,0,0;
  PART: Right,White,-16,0,2,All,256,160;

  END b1

  b2

  BUFFER ICON: Status,95,464,152,1,1,0,0;
  PART: Right,White,-16,0,2,All,480,160;
```

```
END b2

CNV_RSS1

GAPS: 96,....;
PART: Right,White,-16,0,2,All,224,160;

END CNV_RSS1

CNV_RSS2

GAPS: 96,....;
PART: Right,White,-32,0,4,All,432,160;
VIEW: 2,157,56,-1;

END CNV_RSS2

CNV_RSS3

GAPS: 96,....;
PART: Right,White,-32,0,4,All,656,160;
VIEW: 2,157,56,-1;

END CNV_RSS3

flag1

NAME: Standard,White,272,248;
VALUES: Standard,White,1,320,256,0,0,1;

END flag1

flag1a

NAME: Standard,White,264,264;
VALUES: Standard,White,1,320,272,0,0,1;

END flag1a

flag1b

NAME: Standard,White,264,280;
VALUES: Standard,White,1,320,288,0,0,1;

END flag1b

flag1c

NAME: Standard,White,264,296;
VALUES: Standard,White,1,320,304,0,0,1;

END flag1c

flag2

NAME: Standard,White,496,248;
VALUES: Standard,White,1,544,256,0,0,1;

END flag2

flag2a

NAME: Standard,White,488,264;
VALUES: Standard,White,1,544,272,0,0,1;

END flag2a

flag2b

NAME: Standard,White,488,280;
```

```
VALUES: Standard,White,1,544,288,0,0,1;

END flag2b

flag2c

NAME: Standard,White,488,296;
VALUES: Standard,White,1,544,304,0,0,1;

END flag2c

cnt1

NAME: Standard,White,280,232;
VALUES: Standard,White,2,320,240,0,0,1;

END cnt1

cnt2

NAME: Standard,White,504,232;
VALUES: Standard,White,2,544,240,0,0,1;

END cnt2

unload1

NAME: Standard,White,320,104;

END unload1

unload2

NAME: Standard,White,544,104;

END unload2

load1

NAME: Standard,White,320,120;

END load1

load2

NAME: Standard,White,544,120;

END load2

load4


END load4

SMT_RSS1

MACHINE ICON: Status,33,272,152,1,1,0,0;
GAPS: 45,....;
PART: Right,White,-32,0,4,All,304,160;
VIEW: 1,230,234,-1;

END SMT_RSS1

SMT_RSS2

MACHINE ICON: Status,33,496,152,1,1,0,0;
GAPS: 45,....;
PART: Right,White,-32,0,4,All,528,160;
VIEW: 1,374,234,-1;
```

```
  END SMT_RSS2

  dummy1a

  NAME: Standard,White,256,120;
  GAPS: 45,....;

  END dummy1a

  dummy2a

  NAME: Standard,White,480,120;
  GAPS: 45,....;

  END dummy2a

  dummy1b

  NAME: Standard,White,256,136;
  GAPS: 45,....;

  END dummy1b

  dummy2b

  NAME: Standard,White,480,136;
  GAPS: 45,....;

  END dummy2b

  timer1

  NAME: Standard,White,256,104;
  GAPS: 45,....;

  END timer1

  timer2

  NAME: Standard,White,480,104;
  GAPS: 45,....;

  END timer2

END SELECT

END DISPLAY

DETAIL
  OPTIONS
    BREAKDOWN MODEL: Actual;
    REPAIR MODEL: Actual;
    LABOR TO UNLOAD : No;
    WARMUP PERIOD : 0.00;
    OUTPUT INTERVAL : None;
    UNBLOCK BASIS : Priority;
    MONITOR STEP : Undefined;
    MIXTURE STEP : Undefined;
    MODULE ELEMENT NAMES : Use local preferences;
  END OPTIONS

SELECT

  unload1

  NAME OF FUNCTION: unload1;
  TYPE: Name;
  PARAMETERS: 0
  ACTIONS, Execute
  Add
```

Appendix A Simulation and Experiment Data (Continued)

```
!THE UNLOAD FUNCTION SETS flag1=1, WHICH STARTS timer1, AND WAITS
!FOR TWO PARTS TO ARRIVE IN b1. THE UNLOAD FUNCTION RETURNS THE CORRECT
!ELEMENT TO PUSH TO DEPENDING ON PART TYPE.
    PRINT "SMT_RSS1 is blocked because b1 does not have parts"
    flag1 = 1
    IF NPARTS (b1) < 2
     RETURN NONE
    ELSEIF TYPE = board
     RETURN CNV_RSS2
    ELSE
     RETURN SCRAP
    ENDIF
  End Actions

  END unload1

  unload2

  NAME OF FUNCTION: unload2;
  TYPE: Name;
  PARAMETERS: 0
  ACTIONS, Execute
  Add
!THE UNLOAD FUNCTION SETS flag2=1, WHICH STARTS timer2, AND WAITS
!FOR TWO PARTS TO ARRIVE IN b2. THE UNLOAD FUNCTION RETURNS THE CORRECT
!ELEMENT TO PUSH TO DEPENDING ON PART TYPE.
    PRINT "SMT_RSS2 is blocked because b2 does not have parts"
    flag2 = 1
    IF NPARTS (b2) < 2
     RETURN NONE
    ELSEIF TYPE = board
     RETURN CNV_RSS3
    ELSE
     RETURN SCRAP
    ENDIF
  End Actions

  END unload2

  load1

  NAME OF FUNCTION: load1;
  TYPE: Name;
  PARAMETERS: 0
  ACTIONS, Execute
  Add
!THE LOAD FUNCTION WILL CHECK FOR TWO PARTS IN THE BUFFER b1. IF ZERO
!PARTS THEN RETURN NONE (EQUAL TO WAIT). IF ONE THEN START timer1 AND
!RETURN NONE. IF TWO PARTS THEN CHECK FOR BOTH BEING DUMMY PARTS, IF SO
!THEN START dummy1b (USING flag1b) TO PULL THE PARTS OUT. IF TWO PARTS
!AND NOT BOTH DUMMY SET flag1c AND RETURN b1. SETTING flag1c ALLOWS
!SMT_RSS1 TO CONTINUE LOADING PARTS.
    IF NPARTS (b1) < 2
     IF NPARTS (b1) > 0
      flag1 = 2
      RETURN NONE
     ELSE
      RETURN NONE
     ENDIF
    ELSE
     IF b1:TYPE = dum AND b1 AT 2:TYPE = dum
      flag1b = 1
      RETURN NONE
     ELSE
      flag1c = 1
      flag1b = 0
      flag1a = 0
      RETURN b1
     ENDIF
    ENDIF
```

```
End Actions

END load1

load2

NAME OF FUNCTION: load2;
TYPE: Name;
PARAMETERS: 0
ACTIONS, Execute
Add
!THE LOAD FUNCTION WILL CHECK FOR TWO PARTS IN THE BUFFER b2. IF ZERO
!PARTS THEN RETURN NONE (EQUAL TO WAIT). IF ONE THEN START timer2 AND
!RETURN NONE. IF TWO PARTS THEN CHECK FOR BOTH BEING DUMMY PARTS, IF SO
!THEN START dummy2b (USING flag2b) TO PULL THE PARTS OUT. IF TWO PARTS
!AND NOT BOTH DUMMY SET flag2c AND RETURN b2. SETTING flag2c ALLOWS
!SMT_RSS2 TO CONTINUE LOADING PARTS.
    IF NPARTS (b2) < 2
     IF NPARTS (b2) > 0
      flag2 = 2
      RETURN NONE
     ELSE
      RETURN NONE
     ENDIF
    ELSE
     IF b2:TYPE = dum AND b2 AT 2:TYPE = dum
      flag2b = 1
      RETURN NONE
     ELSE
      flag2c = 1
      flag2b = 0
      flag2a = 0
      RETURN b2
     ENDIF
    ENDIF
End Actions

END load2

load4

NAME OF FUNCTION: load4;
TYPE: Integer;
PARAMETERS: 0

END load4

board

NAME OF PART: board;
TYPE: Variable attributes;
GROUP NUMBER: 1;
MAXIMUM ARRIVALS: Unlimited;
INTER ARRIVAL TIME: POISSON (m,p);
FIRST ARRIVAL AT: 0.0;
LOT SIZE: 1;
OUTPUT RULE: PUSH to CNV_RSS1 at Rear;
PART ROUTE: None
REPORTING: Yes;
CONTAINS FLUIDS: No;
SHIFT: Undefined;

END board
```

The parameter m establishes the mean arrival time.

The parameter p establishes a random number sequence.

```
dum

NAME OF PART: dum;
NOTES:
  This dummy pat is assigned the same group as the real part so it has the
  same attributes allowing the machine to assign them during the process.
```

```
  These parts are pulled from world and scraped. The part is used in two
  places "dummy machine" and "SMT_RSS1".
END NOTES
TYPE: Variable attributes;
GROUP NUMBER: 1;
MAXIMUM ARRIVALS: 0;
OUTPUT RULE: Wait;
PART ROUTE: None
REPORTING: Yes;
CONTAINS FLUIDS: No;
SHIFT: Undefined;

END dum

BRD_NUM

NAME OF ATTRIBUTE: BRD_NUM;
QUANTITY: 1;

END BRD_NUM

SMTPRSS1

NAME OF ATTRIBUTE: SMTPRSS1;
QUANTITY: 1;

END SMTPRSS1

SMTPRSS2

NAME OF ATTRIBUTE: SMTPRSS2;
QUANTITY: 1;

END SMTPRSS2

Board_AA

NAME OF BUFFER: Board_AA;
QUANTITY: 1;
CAPACITY: 1000;
DELAY TIME : Undefined;
INPUT POSITION: Rear;
OUTPUT SCAN FROM: Front;
* Select: First;
REPORTING: Individual;
SHIFT: Undefined,0;

END Board_AA

Board_BB

NAME OF BUFFER: Board_BB;
QUANTITY: 1;
CAPACITY: 1000;
DELAY TIME : Undefined;
INPUT POSITION: Rear;
OUTPUT SCAN FROM: Front;
* Select: First;
REPORTING: Individual;
SHIFT: Undefined,0;

END Board_BB

Board_AB

NAME OF BUFFER: Board_AB;
QUANTITY: 1;
CAPACITY: 1000;
DELAY TIME : Undefined;
INPUT POSITION: Rear;
```

```
OUTPUT SCAN FROM: Front;
* Select: First;
REPORTING: Individual;
SHIFT: Undefined,0;

END Board_AB

Board_BA

NAME OF BUFFER: Board_BA;
QUANTITY: 1;
CAPACITY: 1000;
DELAY TIME : Undefined;
INPUT POSITION: Rear;
OUTPUT SCAN FROM: Front;
* Select: First;
REPORTING: Individual;
SHIFT: Undefined,0;

END Board_BA

b1

NAME OF BUFFER: b1;
NOTES:
  THIS BUFFER REPRESENTS THE TWO POSITIONS THAT MUST BE FILLED BEFORE
  SMT_RSS1 UNLOADS AND LOADS PARTS. THIS BUFFER IS FEED FROM EITHER CNV-RSS1
  OR dummy1b AND FEEDS TO SMT_RSS1 OR dummy1c.

END NOTES
QUANTITY: 1;
CAPACITY: 2;
DELAY TIME : Undefined;
INPUT POSITION: Rear;
OUTPUT SCAN FROM: Front;
* Select: First;
REPORTING: Individual;
SHIFT: Undefined,0;

END b1

b2

NAME OF BUFFER: b2;
NOTES:
  THIS BUFFER REPRESENTS THE TWO POSITIONS THAT MUST BE FILLED BEFORE
  SMT_RSS2 UNLOADS AND LOADS PARTS. THIS BUFFER IS FEED FROM EITHER CNV-RSS2
  OR dummy2b AND FEEDS TO SMT_RSS2 OR dummy2c.

END NOTES
QUANTITY: 1;
CAPACITY: 2;
DELAY TIME : Undefined;
INPUT POSITION: Rear;
OUTPUT SCAN FROM: Front;
* Select: First;
REPORTING: Individual;
SHIFT: Undefined,0;

END b2

CNV_RSS1

NAME OF CONVEYOR: CNV_RSS1;
NOTES:
  Name: Belt conveyor
  Cycle Time: calculated as the period of time needed for the board to
  travel it's length on the conveyor
END NOTES
QUANTITY: 1;
```

Appendix A Simulation and Experiment Data (Continued)

```
TYPE: Queuing;
PART LENGTH: 11;
MAX CAPACITY: 11;
INPUT RULE: Wait;
OUTPUT RULE: PUSH to b1;
CYCLE TIME: 1.0;
BREAKDOWNS: No;
PRIORITY: Undefined;
LABOR:
  Repair: None;
END
REPORTING: Individual;
SHIFT: Undefined;

END CNV_RSS1

CNV_RSS2

NAME OF CONVEYOR: CNV_RSS2;
QUANTITY: 1;
TYPE: Queuing;
PART LENGTH: 4;
MAX CAPACITY: 4;
INPUT RULE: Wait;
OUTPUT RULE: !push these parts to buffer b2
             PUSH to b2;
CYCLE TIME: 1.0;
BREAKDOWNS: No;
PRIORITY: Undefined;
LABOR:
  Repair: None;
END
REPORTING: Individual;
SHIFT: Undefined;

END CNV_RSS2

CNV_RSS3

NAME OF CONVEYOR: CNV_RSS3;
QUANTITY: 1;
TYPE: Fixed;
PART LENGTH: 4;
MAX CAPACITY: 4;
INPUT RULE: Wait;
OUTPUT RULE: IF SMTPRSS1 = 1
              IF SMTPRSS2 = 1
               PUSH to Board_AA
              ELSE
               PUSH to Board_AB
              ENDIF
             ELSEIF SMTPRSS1 = 2
              IF SMTPRSS2 = 2
               PUSH to Board_BB
              ELSE
               PUSH to Board_BA
              ENDIF
             ELSE
              Wait
             ENDIF;
CYCLE TIME: 7.0;
BREAKDOWNS: No;
PRIORITY: Undefined;
LABOR:
  Repair: None;
END
REPORTING: Individual;
SHIFT: Undefined;

END CNV_RSS3
```

```
flag1

NAME OF VARIABLE: flag1;
NOTES:
  flag1 IS MONITORED BY timer1. IT IS RESET ONLY BY timer1. NO REPORTING.
  IT IS SET TO 1 BY unload1 AND 2 BY load1.
END NOTES
QUANTITY: 1;
REPORTING: No;

END flag1

flag1a

NAME OF VARIABLE: flag1a;
NOTES:
  flag1a IS MONITORED BY dummy1a AND IS SET TO 1 BY timer1 AND IS RESET TO
  ZERO BY dummy1a. NO REPORTING
END NOTES
QUANTITY: 1;
REPORTING: No;

END flag1a

flag1b

NAME OF VARIABLE: flag1b;
NOTES:
  flag1b IS MONITORED BY dummy1b AND IS SET TO 1 BY load1 AND RESET TO
  ZERO BY dummy1b. NO REPORTING.
END NOTES
QUANTITY: 1;
REPORTING: No;

END flag1b

flag1c

NAME OF VARIABLE: flag1c;
NOTES:
  flag1c IS MONITORED BY SMT_RSS1 AND IS SET TO 1 BY load1 AND RESET TO
  ZERO BY SMT_RSS1. NO REPORTING.
END NOTES
QUANTITY: 1;
REPORTING: No;

END flag1c

flag2

NAME OF VARIABLE: flag2;
NOTES:
  flag2 IS MONITORED BY timer2. IT IS RESET ONLY BY timer2. NO REPORTING.
  IT IS SET TO 1 BY unload2 AND 2 BY load2.
END NOTES
QUANTITY: 1;
REPORTING: No;

END flag2

flag2a

NAME OF VARIABLE: flag2a;
NOTES:
  flag2a IS MONITORED BY dummy2a AND IS SET TO 1 BY timer2 AND IS RESET TO
  ZERO BY dummy2a. NO REPORTING
END NOTES
QUANTITY: 1;
REPORTING: No;
```

```
END flag2a

flag2b

NAME OF VARIABLE: flag2b;
NOTES:
  flag2b IS MONITORED BY dummy2b AND IS SET TO 1 BY load2 AND RESET TO
  ZERO BY dummy2b. NO REPORTING.
END NOTES
QUANTITY: 1;
REPORTING: No;

END flag2b

flag2c

NAME OF VARIABLE: flag2c;
NOTES:
  flag2c IS MONITORED BY SMT_RSS2 AND IS SET TO 1 BY load2 AND RESET TO
  ZERO BY SMT_RSS2. NO REPORTING.
END NOTES
QUANTITY: 1;
REPORTING: No;

END flag2c

cnt1

NAME OF VARIABLE: cnt1;
NOTES:
  THIS VARIABLE IS USED TO COUNT COMPLETED BOARDS FROM timer1 (EQUIVILENT TO
  SECONDS). IT IS RESET TO ZERO ONLY BY timer1. NO REPORTING.
END NOTES
QUANTITY: 1;
REPORTING: No;

END cnt1

cnt2

NAME OF VARIABLE: cnt2;
NOTES:
  THIS VARIABLE IS USED TO COUNT COMPLETED BOARDS FROM timer2 (EQUIVILENT TO
  SECONDS). IT IS RESET TO ZERO ONLY BY timer2. NO REPORTING.
END NOTES
QUANTITY: 1;
REPORTING: No;

END cnt2

SMT_RSS1

NAME OF MACHINE: SMT_RSS1;
NOTES:
  DATE LAST MODIFIED:11/18/94
  ____________________________________
  PROCESS DESCRIPTION SMT_RSS1:
  THIS PHILIPS FIVE STAR SURFACE MOUNT MACHINE IS A TWO HEADED VERSION
  AND IS MODELED AS A BATCH MACHINE WITH A CYCLE TIME OF 10.66/TWO BOARDS.
  THIS MACHINE PULLS FROM b1 AND PUSHES TO CNV_RSS2 OR SCRAP. ON START THIS
  IDENTIFES WHICH POSITION EACH BOARD OCCUPIED DURING PLACEMENT BY CHANGING
  SMTPRSS1 ATTRIBUTE TO 1 OR 2. POSITION ONE IS THE FORWARD MOST POSITION
  IN THIS MACHINE. ON FINISH THIS MACHINE CHANGES THE ICON ATTRIBUTE TO 3,
  THIS REPRESENTS THE POPULATED BOARD.
  ____________________________________
  PROCESS DATA:
  CYC TIME/UNIT(SEC)=5.33
  ME%=.80
  YLD%=.98
```

```
CAP/MACH=3821
#MACH=1
#OPER=1
CAP/SHIFT=3821
SOURCE=ROLAND McKENZIE
DATE=10/26/94
_________________________________
TO DO: CHANGE THE ICON ATTRIBUTE FOR POSITION ONE ONLY
_________________________________
NOTES:
_________________________________
INPUT RULE
!ON INPUT CHECK TO SEE IF FLAG1C IS 0. IF IT IS, RUN THE NAME FUNCTION
!LOAD. IF FLAG1C IS SET TO 1 CONTINUE PULLING FROM B1 UNTIL SMT_RSS1 IS
!FILLED.
IF flag1c = 0
 PULL from go1 ()
ELSE
 PULL from b1
ENDIF
_________________________________
OUTPUT RULE
!WAIT FOR TWO PARTS IN B1 BEFORE UNLOADING
PUSH to wait1 ()
_________________________________
ACTIONS ON START
!IDENTIFY WHICH POSITION EACH BOARD OCCUPIED DURING PLACEMENT AND SET
!THE POSITION ATTRIBUTE SMTPRSS1 TO 1 OR 2. RESET INPUT FLAG1C TO ZERO.
SMT_RSS1:SMTPRSS1 = 1
SMT_RSS1 AT 2:SMTPRSS1 = 2
flag1c = 0
_________________________________
ACTIONS ON FINISH
!CHANGE ICON FROM PASTED BOARD TO POPULATED BOARD
IF TYPE = board
 ICON = 3
ENDIF




END NOTES
QUANTITY: 1;
TYPE: Batch;
* Batch min: 2;
* Batch max: 2;
PRIORITY: Undefined;
LABOR:
  Repair: None;
END
LABOR:
  Cycle: None;
END
DISCRETE LINKS :
       Fill: None
END
DISCRETE LINKS :
       Empty: None
END
CYCLE TIME: 10.66;
BREAKDOWNS: No;
ACTIONS, Start
Add
!IDENTIFY WHICH POSITION EACH BOARD OCCUPIED DURING PLACEMENT AND SET
!THE POSITION ATTRIBUTE SMTPRSS1 TO 1 OR 2. RESET INPUT FLAG1C TO ZERO.
   SMT_RSS1:SMTPRSS1 = 1
   SMT_RSS1 AT 2:SMTPRSS1 = 2
   flag1c = 0
```

```
  End Actions
  ACTIONS, Finish
  Add
!CHANGE ICON FROM PASTED BOARD TO POPULATED BOARD
    IF TYPE = board
     ICON = 3
    ENDIF
  End Actions
  INPUT RULE: !ON INPUT CHECK TO SEE IF FLAG1C IS 0, IF IT IS RUN THE NAME FUNCTION
              !LOAD. IF FLAG1C IS SET TO 1 CONTINUE PULLING FROM B1 UNTIL SMT_RSS1 IS
              !FILLED.
              IF flag1c = 0
               PULL from load1 ()
              ELSE
               PULL from b1
              ENDIF;
  OUTPUT RULE: !WAIT FOR TWO PARTS IN B1 BEFORE UNLOADING
               PUSH to unload1 ();
  REPORTING: Individual;
  SHIFT: Undefined,0,0;

  END SMT_RSS1

  SMT_RSS2

  NAME OF MACHINE: SMT_RSS2;
  NOTES:
    DATE LAST MODIFIED:11/18/94
    __________________________________
    PROCESS DESCRIPTION SMT_RSS2:
    THIS PHILIPS FIVE STAR SURFACE MOUNT MACHINE IS A TWO HEADED VERSION
    AND IS MODELED AS A BATCH MACHINE WITH A CYCLE TIME OF 10.66/TWO BOARDS.
    THIS MACHINE PULLS FROM b2 AND PUSHES TO CNV_RSS3 OR SCRAP. ON START THIS
    IDENTIFES WHICH POSITION EACH BOARD OCCUPIED DURING PLACEMENT BY CHANGING
    SMTPRSS2 ATTRIBUTE TO 1 OR 2. POSITION ONE IS THE FORWARD MOST POSITION
    IN THIS MACHINE. ON FINISH THIS MACHINE CHANGES THE ICON ATTRIBUTE TO 3,
    THIS REPRESENTS THE POPULATED BOARD.
    __________________________________
    PROCESS DATA:
    CYC TIME/UNIT(SEC)=5.33
    ME%=.80
    YLD%=.98
    CAP/MACH=3821
    #MACH=1
    #OPER=1
    CAP/SHIFT=3821
    SOURCE=ROLAND McKENZIE
    DATE=10/26/94
    __________________________________
    TO DO: CHANGE THE ICON ATTRIBUTE FOR POSITION ONE ONLY
    __________________________________
    NOTES:
    __________________________________
    INPUT RULE
    !ON INPUT CHECK TO SEE IF FLAG2C IS 0, IF IT IS RUN THE NAME FUNCTION
    !LOAD. IF FLAG2C IS SET TO 1 CONTINUE PULLING FROM b2 UNTIL SMT_RSS2 IS
    !FILLED.
    IF flag2c = 0
     PULL from load2 ()
    ELSE
     PULL from b2
    ENDIF
    __________________________________
    OUTPUT RULE
    !WAIT FOR TWO PARTS IN b2 BEFORE UNLOADING
    PUSH to unload2 ()
    __________________________________
    ACTIONS ON START
    !IDENTIFY WHICH POSITION EACH BOARD OCCUPIED DURING PLACEMENT AND SET
    !THE POSITION ATTRIBUTE SMTPRSS2 TO 1 OR 2. RESET INPUT FLAG2C TO ZERO.
```

```
    SMT_RSS2:SMTPRSS2 = 1
    SMT_RSS2 AT 2:SMTPRSS2 = 2
    flag2c = 0
    ________________________________
    ACTIONS ON FINISH
    !CHANGE ICON FROM PASTED BOARD TO POPULATED BOARD
    IF TYPE = board
     ICON = 3
    ENDIF




  END NOTES
  QUANTITY: 1;
  TYPE: Batch;
  * Batch min: 2;
  * Batch max: 2;
  PRIORITY: Undefined;
  LABOR:
    Repair: None;
  END
  LABOR:
    Cycle: None;
  END
  DISCRETE LINKS :
         Fill: None
  END
  DISCRETE LINKS :
         Empty: None
  END
  CYCLE TIME: 10.66;
  BREAKDOWNS: No;
  ACTIONS, Start
  Add
!IDENTIFY WHICH POSITION EACH BOARD OCCUPIED DURING PLACEMENT AND SET
!THE POSITION ATTRIBUTE SMTPRSS2 TO 1 OR 2. RESET INPUT FLAG2C TO ZERO.
    SMT_RSS2:SMTPRSS2 = 1
    SMT_RSS2 AT 2:SMTPRSS2 = 2
    flag2c = 0
  End Actions
  ACTIONS, Finish
  Add
!CHANGE ICON FROM PASTED BOARD TO POPULATED BOARD
    IF TYPE = board
     ICON = 3
    ENDIF
  End Actions
  INPUT RULE: !ON INPUT CHECK TO SEE IF FLAG2C IS 0, IF IT IS RUN THE NAME FUNCTION
              !LOAD. IF FLAG2C IS SET TO 1 CONTINUE PULLING FROM b2 UNTIL SMT_RSS2 IS
              !FILLED.
              IF flag2c = 0
               PULL from load2 ()
              ELSE
               PULL from b2
              ENDIF;
  OUTPUT RULE: !WAIT FOR TWO PARTS IN B1 BEFORE UNLOADING
               PUSH to unload2 ();
  REPORTING: Individual;
  SHIFT: Undefined,0,0;

  END SMT_RSS2

  dummy1a

  NAME OF MACHINE: dummy1a;
  NOTES:
    THIS MACHINE WILL PULL TWO DUMY PARTS FROM WORLD AFTER 19 SECONDS,
```

```
    PROCESS THEM FOR 1 SECOND AND PUSH THEM TO b1.
    ON FINISH IT WILL TURN ITSELF OFF BY SETTING flag1a=0

    ___________________________________
    INPUT RULE
    !AFTER THE timer1 COUNTS TO 19 SECONDS flag1a IS SET TO 1.
    IF flag1a = 1
     PULL from dum out of WORLD
    ELSE
     Wait
    ENDIF
    ___________________________________
    OUTPUT RULE
    !PUSH BOTH DUMMY PARTS TO b1, IF IT IS FULL PUSH TO SCRAP.
    PUSH to b1,SCRAP
    ___________________________________
    ACTIONS ON START
    ___________________________________
    ACTIONS ON FINISH
    !RESET flag1a TO ZERO AFTER COMPLETING ONE BATCH OF PARTS.
    flag1a = 0



  END NOTES
  QUANTITY: 1;
  TYPE: Batch;
  * Batch min: 2;
  * Batch max: 2;
  PRIORITY: Undefined;
  LABOR:
    Repair: None;
  END
  LABOR:
    Cycle: None;
  END
  DISCRETE LINKS :
         Fill: None
  END
  DISCRETE LINKS :
         Empty: None
  END
  CYCLE TIME: 1.0;
  BREAKDOWNS: No;
  ACTIONS, Finish
  Add
!RESET flag1a TO ZERO AFTER COMPLETING ONE BATCH OF PARTS.
    flag1a = 0
  End Actions
  INPUT RULE: !AFTER THE timer1 COUNTS TO 19 SECONDS flag1a IS SET TO 1.
              IF flag1a = 1
               PULL from dum out of WORLD
              ELSE
               Wait
              ENDIF;
  OUTPUT RULE: !PUSH BOTH DUMMY PARTS TO b1, IF IT IS FULL PUSH TO SCRAP.
               PUSH to b1,SCRAP;
  REPORTING: Individual;
  SHIFT: Undefined,0,0;

  END dummy1a

  dummy2a

  NAME OF MACHINE: dummy2a;
  NOTES:
    THIS MACHINE WILL PULL TWO DUMY PARTS FROM WORLD AFTER 19 SECONDS,
    PROCESS THEM FOR 1 SECOND AND PUSH THEM TO b2 OR SCRAP.
    ON FINISH IT WILL TURN ITSELF OFF BY SETTING flag2a=0
    ___________________________________
```

```
  INPUT RULE
  !AFTER THE timer2 COUNTS TO 19 SECONDS flag2a IS SET TO 1.
  IF flag2a = 1
   PULL from dum out of WORLD
  ELSE
   Wait
  ENDIF
  ________________________________
  OUTPUT RULE
  !PUSH BOTH DUMMY PARTS TO b2, IF IT IS FULL PUSH TO SCRAP.
  PUSH to b2,SCRAP
  ________________________________
  ACTIONS ON START
  ________________________________
  ACTIONS ON FINISH
  !RESET flag2a TO ZERO AFTER COMPLETING ONE BATCH OF PARTS.
  flag2a = 0



END NOTES
QUANTITY: 1;
TYPE: Batch;
* Batch min: 2;
* Batch max: 2;
PRIORITY: Undefined;
LABOR:
  Repair: None;
END
LABOR:
  Cycle: None;
END
DISCRETE LINKS :
       Fill: None
END
DISCRETE LINKS :
       Empty: None
END
CYCLE TIME: 1.0;
BREAKDOWNS: No;
ACTIONS, Finish
Add
!RESET flag2a TO ZERO AFTER COMPLETING ONE BATCH OF PARTS.
  flag2a = 0
End Actions
INPUT RULE: !AFTER THE timer2 COUNTS TO 19 SECONDS flag2a IS SET TO 1.
            IF flag2a = 1
             PULL from dum out of WORLD
            ELSE
             Wait
            ENDIF;
OUTPUT RULE: !PUSH BOTH DUMMY PARTS TO b2, IF IT IS FULL PUSH TO SCRAP.
             PUSH to b2,SCRAP;
REPORTING: Individual;
SHIFT: Undefined,0,0;

END dummy2a

dummy1b

NAME OF MACHINE: dummy1b;
NOTES:
  THE PURPOSE OF THIS MACHINE IS TO REMOVE TWO DUMMY PARTS FROM b1 IF
  BOTH PARTS ARE DUMMY PARTS. THE TWO DUMMY PARTS MUST ARRIVE IN b1 SO
  SMT_RSS1 UNLOADS, BUT THEY SHOULD NOT BE LOADED INTO SMT_RSS1.
  ________________________________
  INPUT RULE
  !PULL DUMMY PARTS FROM b1 IF TWO ARE WAITING (flag1b=1)
  IF flag1b = 1
```

```
    PULL from dum out of b1
   ELSE
    Wait
   ENDIF
   ______________________________
   OUTPUT RULE
   PUSH to SCRAP
   ______________________________
   ACTIONS ON START
   !RESET flag1b TO ZERO
   flag1b = 0
   ______________________________
   ACTIONS ON FINISH


  END NOTES
  QUANTITY: 1;
  TYPE: Batch;
  * Batch min: 2;
  * Batch max: 2;
  PRIORITY: Undefined;
  LABOR:
    Repair: None;
  END
  LABOR:
    Cycle: None;
  END
  DISCRETE LINKS :
         Fill: None
  END
  DISCRETE LINKS :
         Empty: None
  END
  CYCLE TIME: 0.1;
  BREAKDOWNS: No;
  ACTIONS, Start
  Add
!RESET flag1b TO ZERO
    flag1b = 0
  End Actions
  INPUT RULE: !PULL DUMMY PARTS FROIM b1 IF TWO ARE WAITING (flag1b=1)
              IF flag1b = 1
               PULL from dum out of b1
              ELSE
               Wait
              ENDIF;
  OUTPUT RULE: PUSH to SCRAP;
  REPORTING: Individual;
  SHIFT: Undefined,0,0;

  END dummy1b

  dummy2b

  NAME OF MACHINE: dummy2b;
  NOTES:
    THE PURPOSE OF THIS MACHINE IS TO REMOVE TWO DUMMY PARTS FROM b2 IF
    BOTH PARTS ARE DUMMY PARTS. THE TWO DUMMY PARTS MUST ARRIVE IN b2 SO
    SMT_RSS2 UNLOADS, BUT THEY SHOULD NOT BE LOADED INTO SMT_RSS2.
    ______________________________
    INPUT RULE
    !PULL DUMMY PARTS FROM b2 IF TWO ARE WAITING (flag2b=1)
    IF flag2b = 1
     PULL from dum out of b2
    ELSE
     Wait
    ENDIF
    ______________________________
    OUTPUT RULE
```

```
    PUSH to SCRAP
    ________________________________
    ACTIONS ON START
    !RESET flag2b TO ZERO
    flag2b = 0
    ________________________________
    ACTIONS ON FINISH


  END NOTES
  QUANTITY: 1;
  TYPE: Batch;
  * Batch min: 2;
  * Batch max: 2;
  PRIORITY: Undefined;
  LABOR:
    Repair: None;
  END
  LABOR:
    Cycle: None;
  END
  DISCRETE LINKS :
         Fill: None
  END
  DISCRETE LINKS :
         Empty: None
  END
  CYCLE TIME: 0.1;
  BREAKDOWNS: No;
  ACTIONS, Start
  Add
!RESET flag2b TO ZERO
    flag2b = 0
  End Actions
  INPUT RULE: !PULL DUMMY PARTS FROM b2 IF TWO ARE WAITING (flag2b=1)
              IF flag2b = 1
               PULL from dum out of b2
              ELSE
               Wait
              ENDIF;
  OUTPUT RULE: PUSH to SCRAP;
  REPORTING: Individual;
  SHIFT: Undefined,0,0;

  END dummy2b

  timer1

  NAME OF MACHINE: timer1;
  NOTES:
    THIS MACHINE WILL PULL PARTS OUT OUT OF THE WORLD AND SCRAP THEM IN ONE
    SECOND CYCLE TIMES. IT IS STARTED WHEN flag1=1, AND STOPPED WHEN flag1=0
    ACTIONS ON FINISH INCLUDE COUNTING TO 19 AND THEN SETTING flag1a=1
    ________________________________
    INPUT RULE
    !CHECK THE VALUE OF flag1, IF IT IS NOT ZERO THEN START PULLING DUMMY
    !PARTS FROM WORLD
    IF flag1 > 0
     PULL from dum out of WORLD
    ELSE
     Wait
    ENDIF
    ________________________________
    OUTPUT RULE
    !PUSH THE DUMMY PARTS THAT WERE USED TO TIME 19 SECONDS TO SCRAP
    PUSH to SCRAP
    ________________________________
    ACTIONS ON START
    ________________________________
```

```
    ACTIONS ON FINISH
    !IF flag1=1 COUNT THE BOARDS THAT ARE PRODUCED (1 SEC), IF SMT_RSS1 IS
    !NOT BLOCKED THEN RESET COUNTER AND STOP PROCESS. IF COUNT REACHES 19
    !RESET COUNTER AND CHECK IF SMT_RSS1 IS STILL BLOCKED IF IT IS THEN SET
    !flag1a=1. THIS WILL START dummy1a WHICH WILL MAKE TO DUMMY PARTS AND
    !PUSH THEM TO b1 OR SCRAP.
    !IF flag1=2 COUNT THE BOARDS THAT ARE PRODUCED (1 SEC), IF SMT_RSS1 IS
    !NOT WAITING THEN RESET COUNTER AND STOP PROCESS. IF COUNT REACHES 19
    !RESET COUNTER AND CHECK IF SMT_RSS1 IS STILL WAITING IF IT IS THEN SET
    !flag1a=1. THIS WILL START dummy1a WHICH WILL MAKE TO DUMMY PARTS AND
    !PUSH THEM TO b1 OR SCRAP.
    IF flag1 = 1
     cnt1 = cnt1 + 1
     IF ISTATE (SMT_RSS1) <> 3
      cnt1 = 0
      flag1 = 0
     ENDIF
     IF cnt1 = 19
      IF ISTATE (SMT_RSS1) = 3
       flag1a = 1
      ENDIF
      cnt1 = 0
      flag1 = 0
     ENDIF
    ELSEIF flag1 = 2
     cnt1 = cnt1 + 1
     IF ISTATE (SMT_RSS1) <> 1
      cnt1 = 0
      flag1 = 0
     ENDIF
     IF cnt1 = 19
      IF ISTATE (SMT_RSS1) = 1
       flag1a = 1
       flag1 = 0
      ENDIF
      cnt1 = 0
      flag1 = 0
     ENDIF
    ENDIF


  END NOTES
  QUANTITY: 1;
  TYPE: Single;
  PRIORITY: Undefined;
  LABOR:
    Repair: None;
  END
  LABOR:
    Cycle: None;
  END
  DISCRETE LINKS :
        Fill: None
  END
  DISCRETE LINKS :
        Empty: None
  END
  CYCLE TIME: 1.0;
  BREAKDOWNS: No;
  ACTIONS, Finish
  Add
!IF flag1=1 COUNT THE BOARDS THAT ARE PRODUCED (1 SEC), IF SMT_RSS1 IS
!NOT BLOCKED THEN RESET COUNTER AND STOP PROCESS. IF COUNT REACHES 19
!RESET COUNTER AND CHECK IF SMT_RSS1 IS STILL BLOCKED IF IT IS THEN SET
!flag1a=1. THIS WILL START dummy1a WHICH WILL MAKE TO DUMMY PARTS AND
!PUSH THEM TO b1 OR SCRAP.
!IF flag1=2 COUNT THE BOARDS THAT ARE PRODUCED (1 SEC), IF SMT_RSS1 IS
!NOT WAITING THEN RESET COUNTER AND STOP PROCESS. IF COUNT REACHES 19
!RESET COUNTER AND CHECK IF SMT_RSS1 IS STILL WAITING IF IT IS THEN SET
!flag1a=1. THIS WILL START dummy1a WHICH WILL MAKE TO DUMMY PARTS AND
```

```
!PUSH THEM TO b1 OR SCRAP.
    IF flag1 = 1
     cnt1 = cnt1 + 1
     IF ISTATE (SMT_RSS1) <> 3
      cnt1 = 0
      flag1 = 0
     ENDIF
     IF cnt1 = 19
      IF ISTATE (SMT_RSS1) = 3
       flag1a = 1
      ENDIF
      cnt1 = 0
      flag1 = 0
     ENDIF
    ELSEIF flag1 = 2
     cnt1 = cnt1 + 1
     IF ISTATE (SMT_RSS1) <> 1
      cnt1 = 0
      flag1 = 0
     ENDIF
     IF cnt1 = 19
      IF ISTATE (SMT_RSS1) = 1
       flag1a = 1
       flag1 = 0
      ENDIF
      cnt1 = 0
      flag1 = 0
     ENDIF
    ENDIF
  End Actions
  INPUT RULE: !CHECK THE VALUE OF flag1, IF IT IS NOT ZERO THEN START PULLING DUMMY
              !PARTS FROM WORLD
              IF flag1 > 0
               PULL from dum out of WORLD
              ELSE
               Wait
              ENDIF;
  OUTPUT RULE: !PUSH THE DUMMY PARTS THAT WERE USED TO TIME 19 SECONDS TO SCRAP
               PUSH to SCRAP;
  REPORTING: Individual;
  SHIFT: Undefined,0,0;

  END timer1

  timer2

  NAME OF MACHINE: timer2;
  NOTES:
    THIS MACHINE WILL PULL PARTS OUT OUT OF THE WORLD AND SCRAP THEM IN ONE
    SECOND CYCLE TIMES. IT IS STARTED WHEN flag2=1 OR 2, AND STOPPED WHEN
    flag1=0 ACTIONS ON FINISH INCLUDE COUNTING TO 19 AND THEN SETTING
    flag2a=1.
    ________________________________
    INPUT RULE
    !CHECK THE VALUE OF flag2, IF IT IS NOT ZERO THEN START PULLING DUMMY
    !PARTS FROM WORLD
    IF flag2 > 0
     PULL from dum out of WORLD
    ELSE
     Wait
    ENDIF
    ________________________________
    OUTPUT RULE
    !PUSH THE DUMMY PARTS THAT WERE USED TO TIME 19 SECONDS TO SCRAP
    PUSH to SCRAP
    ________________________________
    ACTIONS ON START
    ________________________________
    ACTIONS ON FINISH
    !IF flag2=1 COUNT THE BOARDS THAT ARE PRODUCED (1 SEC), IF SMT_RSS2 IS
```

```
!NOT BLOCKED THEN RESET COUNTER AND STOP PROCESS. IF COUNT REACHES 19
!RESET COUNTER AND CHECK IF SMT_RSS2 IS STILL BLOCKED IF IT IS THEN SET
!flag2a=1. THIS WILL START dummy2a WHICH WILL MAKE TWO DUMMY PARTS AND
!PUSH THEM TO b2 OR SCRAP.
!IF flag2=2 COUNT THE BOARDS THAT ARE PRODUCED (1 SEC), IF SMT_RSS2 IS
!NOT WAITING THEN RESET COUNTER AND STOP PROCESS. IF COUNT REACHES 19
!RESET COUNTER AND CHECK IF SMT_RSS2 IS STILL WAITING IF IT IS THEN SET
!flag2a=1. THIS WILL START dummy2a WHICH WILL MAKE TWO DUMMY PARTS AND
!PUSH THEM TO b2 OR SCRAP.
IF flag2 = 1
 cnt2 = cnt2 + 1
 IF ISTATE (SMT_RSS2) <> 3
  cnt2 = 0
  flag2 = 0
 ENDIF
 IF cnt2 = 19
  IF ISTATE (SMT_RSS2) = 3
   flag2a = 1
  ENDIF
  cnt2 = 0
  flag2 = 0
 ENDIF
ELSEIF flag2 = 2
 cnt2 = cnt2 + 1
 IF ISTATE (SMT_RSS2) <> 1
  cnt2 = 0
  flag2 = 0
 ENDIF
 IF cnt2 = 19
  IF ISTATE (SMT_RSS2) = 1
   flag2a = 1
   flag2 = 0
  ENDIF
  cnt2 = 0
  flag2 = 0
 ENDIF
ENDIF


END NOTES
QUANTITY: 1;
TYPE: Single;
PRIORITY: Undefined;
LABOR:
  Repair: None;
END
LABOR:
  Cycle: None;
END
DISCRETE LINKS :
       Fill: None
END
DISCRETE LINKS :
       Empty: None
END
CYCLE TIME: 1.0;
BREAKDOWNS: No;
ACTIONS, Finish
Add
!IF flag2=1 COUNT THE BOARDS THAT ARE PRODUCED (1 SEC), IF SMT_RSS2 IS
!NOT BLOCKED THEN RESET COUNTER AND STOP PROCESS. IF COUNT REACHES 19
!RESET COUNTER AND CHECK IF SMT_RSS2 IS STILL BLOCKED IF IT IS THEN SET
!flag2a=1. THIS WILL START dummy2a WHICH WILL MAKE TWO DUMMY PARTS AND
!PUSH THEM TO b2 OR SCRAP.
!IF flag2=2 COUNT THE BOARDS THAT ARE PRODUCED (1 SEC), IF SMT_RSS2 IS
!NOT WAITING THEN RESET COUNTER AND STOP PROCESS. IF COUNT REACHES 19
!RESET COUNTER AND CHECK IF SMT_RSS2 IS STILL WAITING IF IT IS THEN SET
!flag2a=1. THIS WILL START dummy2a WHICH WILL MAKE TWO DUMMY PARTS AND
!PUSH THEM TO b2 OR SCRAP.
    IF flag2 = 1
```

```
     cnt2 = cnt2 + 1
     IF ISTATE (SMT_RSS2) <> 3
      cnt2 = 0
      flag2 = 0
     ENDIF
     IF cnt2 = 19
      IF ISTATE (SMT_RSS2) = 3
       flag2a = 1
      ENDIF
      cnt2 = 0
      flag2 = 0
     ENDIF
    ELSEIF flag2 = 2
     cnt2 = cnt2 + 1
     IF ISTATE (SMT_RSS2) <> 1
      cnt2 = 0
      flag2 = 0
     ENDIF
     IF cnt2 = 19
      IF ISTATE (SMT_RSS2) = 1
       flag2a = 1
       flag2 = 0
      ENDIF
      cnt2 = 0
      flag2 = 0
     ENDIF
    ENDIF
  End Actions
  INPUT RULE: !CHECK THE VALUE OF flag2, IF IT IS NOT ZERO THEN START PULLING DUMMY
              !PARTS FROM WORLD
              IF flag2 > 0
               PULL from dum out of WORLD
              ELSE
               Wait
              ENDIF;
  OUTPUT RULE: !PUSH THE DUMMY PARTS THAT WERE USED TO TIME 19 SECONDS TO SCRAP
               PUSH to SCRAP;
  REPORTING: Individual;
  SHIFT: Undefined,0,0;

  END timer2

END SELECT

END DETAIL
```

Appendix B Representative Sample of Defects by Panel Position

Panel Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Shift 1	2	3	3	2	3	4	4	3	2	1	7	1	1	3	5
Shift 2	3	3	2			1		1		1	1		1	1	
Shift 3	7	2	4	2		3	3	2	1	1	2	3			6
Shift 4	2	1		2	2	1	2	1	2		1	3	2	3	4
Shift 5	1	8	7	3	6	9	6	5	6	7	5	4	4	1	6
Shift 6	2	1		3	2	2	4	1	1	1	2	1	1	1	3
Shift 7	2	1		2	2	1	2	1	3			2	1	1	2
Shift 8	6	5	6	1	6	3	4	7	4		5	2	2	2	4
Shift 9 (mapped)			1	3	2	3	1		1	1	3	3	2		1
Shift 10	5	3	2	2	3	1		3	2	4	3	2	4	2	1
Shift 11	3	5	4	2	9	8	9	3	6	6	7	4	1	7	9
Shift 12	3	2	4	3	1	1	5	4	3	2	3	3	4	5	4
Total (mapped)	36	34	33	25	36	37	40	31	31	24	39	28	23	26	45

Is there statistical difference between the means of the two groups identified by trend analysis? Yes, the means differ by 9.5 defects and the t-Test below, performed at P(α)= .05, indicates that the within group variance does not account for the large difference in means.

t-Test: Two-Sample Assuming Equal Variances

Mapped to Panel			
36	24	39	
34	31	28	← *Group 1*
33	31	23	
25	40	26	
36	37	45	← *Group 2*

	Group 1	*Group 2*
Mean	30.00	39.50
Variance	27.40	16.33
Observations	11.00	4.00
Pooled Variance	24.85	
Hypothesized Mean Difference	Zero	
df	13	
t Stat	-3.26	
P(T<=t) two-tail	0.006	
t Critical two-tail	2.16	

Appendix B Representative Sample of Defects by Panel Position (Continued)

The sample does not identify a trend in defects in the leading or trailing pallet positions. However, the number four panel position circuit boards show a distict difference between leading and trailing positions.

Panel Position	Odd	Even
1	17	19
2	16	18
3	17	16
4	5	20
5	20	16
6	21	16
7	19	21
8	21	10
9	14	17
10	15	9
11	13	26
12	15	13
13	13	10
14	16	10
15	20	25
	242	246

Appendix C Top Side Chip Placement Simulation Results

	Feedback Control = 75 Case #1			Feedback Control = 8 Case #2		
Reliability	Thru-put	Rework	Line Stops	Thru-put	Rework	Line Stops
99.99%	705	0	0	705	0	0
99.99%	705	0	0	705	0	0
99.98%	705	0	0	705	0	0
99.97%	514	61	1	662	15	1
99.94%	514	61	1	662	15	1
99.91%	514	61	1	662	15	1
99.85%	514	61	1	662	15	1
99.75%	514	61	1	662	15	1
99.59%	517	67	1	534	33	4
99.33%	406	80	2	440	40	6
98.89%	406	80	2	438	40	6
98.17%	406	80	2	348	40	8
96.98%	406	80	2	314	40	9
95.02%	406	80	2	275	40	10
91.79%	406	80	2	268	40	10
86.47%	406	80	2	204	40	11