

Batch Automated Reporting and Analysis in Semiconductor Manufacturing Parametrics

Tom Winter, Texas Instruments, Dallas TX
Willy Waks, 2W Systems, Dallas TX

Abstract

Early semiconductor process development is especially dependent on parametric data and analysis is hampered by the large volumes generated and general inability of most engineers to effectively retrieve the data and then process into summarized tables, reports & graphs. The SAS system developed allows the engineer to simply browse through the pre-generated reports - both high level and low level - and quickly visualize and capture the needed data. Key input is a "specs" file with the desired parameters for analysis, description, upper/lower spec limits, target and category (for certain summarized reports). Reports are HTML with multilevel drill down boxcharts; graphics for yields, loss paretos, wafer maps, probability distributions and histograms; yield tables showing summarized and detailed failures to meet spec; summarized html, text reports & gifs for cpk analysis. Pregenerated reports are created by executing SAS through a series of perl programs and configuration files for different devices. On-the-fly reports are also able to be created with customized options through a CGI interface. The system uses the SAS macro language, SAS/GRAPH and SAS/QC.

Introduction

First a few introductory remarks about semiconductor processing. Semiconductors are manufactured through the clean room in lots of 24 wafers each (typically). Testing is done in-line on these wafer at 4-5 insertion points in the process flow, and then extensively tested at the end of the process. End of the line parametric test usually consists of 9-18 sites per wafer (for 200mm wafers) and anywhere from 750-5,000 parameters in our R&D environment. Early phases of R&D are heavily dependent on parametric test (as opposed to functional test) as unit processes are being developed and debugged; full chips would have no yield until much later. Additional challenges are introduced as many lots are split into variations of one process or another as part of a designed experiment to judge the effect of changes in equipment or processing conditions.

As with most software, the SAS Batch Automated Reports (SAS/BAR) went through several evolutionary cycles before reaching their current state. Initially, all parametric data was extracted from the central database into stand alone SAS data sets in a hierarchal directory structure. Willy Waks was hired as a SAS consultant to take various user SAS 6.12 programs being used to process the data and weld them into a consistent set of macros that could be utilized by the global population. Primarily because of the processing times required, a set of perl and c-shell programs were developed to pregenerate reports with a standard set of options. With the release of SAS8, major updates to the code were possible and the data source was redirected to the main database, rather than the stand alone SAS datasets. Existing reports were converted to HTML and the modularity of the macros encouraged additional reports to be added.

For the most part, this paper will describe the overall report system, rather than all the many (many!) particulars of the code generation that were implemented.

Analysis System Requirements

When the plans for development of this system were first being drawn up, we outlined a set of requirements we wanted to accomplish:

- No SAS, SQL or programming experience should be required
- Flexible to pull as much or as little data as the user requires
- Use spec files to define parms to be extracted/analyzed, parameter limits, categories, descriptions
- Use general purpose rules file to define new variables for analysis, screen out bogus data, transform data, etc...
- Able to do complex data joins for multiple programs, devices
- Incorporate special work request (SWR) definitions for experimental split analysis
- Handle long term (multi-lot) as well as single lot analysis
- Key drill down reports from high level to low level

Data Extraction & Manipulation

A great deal of thought was put into all the options we needed to have in order to pull all the data we wanted (and no more), to do multiple joins & concatenations and to massage the data before actual report processing.

The key input to the SAS/BAR system is a specs file, containing the list of parameters to be analyzed, a brief description, spec limits & target, outlier limits, user defined categories, flags for indicating if the parms are to be used for scrap or Cpk calculations and other items. Some parametric programs containing literally thousands of parameters may only require automated reports on 100-150 parms; a specs file is necessary to avoid pulling 100MB from the database and then throwing 95% of the data. Yield and Cpk calculations require limits, some of which are available in the database, but user-defined limits are needed for "what-if" analysis, waivers to spec, etc...

In addition to pulling the parms from the database, we required the flexibility to add calculated parms, do transformations, screen out bogus data that worms its way into the database, etc... A general SAS "rules file" can be specified to define any user defined conditions. For instance, BETA is defined as the ratio of NMOS I-drive to PMOS I-drive, and is critical to track but not stored in the database. BETA is included in the specs file as a normal parameter, but must be defined in the rules file: `BETA=NMOS_IDS/PMOS_IDS;`

Occasionally a test problem introduces unwanted data errors in the database; in order to get more accurate trends & calculations for Cpk and yield, that data needs to be screened out. However, as a policy we do not manipulate the database directly. So code in the rules file is also used for screening:
`if lot in (<lot list>) then <parm>=.;`

Recent updates to the rules file allows inclusion of more complex code, such as additional data extraction from another data source via SQL or libname statements, and rejoining with the primary SAS/BAR data table. This allows extraction, for example, of a key piece of process equipment for the population being analyzed and then coloring a box plot by the key equipment.

Additionally, lot split information on special processes is pulled from another database and merged in with the parametric data, which allows us to produce reports that exclude any non-baseline wafers or summarize and produce single lot reports comparing multiple splits.

Also, we allowed the inputs to the reports to specify a wide range of data combination options. Each record in the final data is uniquely identified by lot, wafer and site indexes. Since each lot can be tested in multiple locations, we allow the user to specify full joins across multiple test points for a given lot. In addition, there are times we want to be able to do a technology wide roll up for several different devices; “vertical joins can also be specified to merge several datasets together for individual devices to create a global report for all those different devices together.

Lastly, there are instances that the users would like to access the raw data behind a report for further ad hoc analysis. All the SQL code behind all the data for the reports is stored in a series of views and subviews that can be accessed from SAS by mapping those directories as SAS libraries and executing them. In addition, the HTML/Javascript tags behind the phonebook and pipeline reports that point to an XML file generated with each set of reports that contains the specifics necessary to regenerate the SQL for data extraction in a 3rd party software package. Many TI facilities use Spotfire for our interactive data extraction and analysis; specific code is included to allow the raw data being reported on to be extracted into Spotfire (more about this in a later section).

Report Details

Phonebooks

“Phonebook” reports are a snapshot combination of a histogram, cumulative probability plot, box plot and wafer map (figure 1). Individual parms are accessed through an HTML “cover sheet” with a list of parms to select. SAS 6.12 version combined all into a single GIF with GREPLAY, but current version in HTML is a collage of 4 larger images that can be selected independently and displayed as a large GIF (much easier on the eyes!). Statistics on the dataset are included with PROC ANNOTATE on the histogram GIF. However, since the box plot seems to be used the most, the next release will have statistics on this graphic as well. (Note on proc shewhart box charts – main features of the boxes are display of mean, median, Q1, Q3, bars for last data point closer than 3*IQR from median, and outlier boxes for points outside of 3*IQR from median)

One of the major advantages of moving to SAS8 and ODS has been the ability to hyperlink reports to other reports to get a more detailed view of the data (figure 2). The multi-lot box plots have the parameter plotted on the y-axis and lot number (ordered by test date) across the x-axis. Clicking on the cross (marker for the mean for that lot) will take the user to the single lot phonebook for that parameter (where x-axis is now wafer within the lot). Such drill down capabilities are extremely

useful for investigating problem lots and troubleshooting large amounts of data very quickly.

Many times, the primary chart of interest is the box plot (lot level trends ordered by test date); clicking once for the individual parm, and clicking once again for the large box plot (and then two more clicks to get back to the parm selection sheet) becomes a real time drain when reviewing upwards of 100 parameters. An additional (very long) HTML page with nothing but the box plots, one after the other, is also provided for easier mass review of all charts.

Phonebooks have the additional capability of plotting split number on the x-axis, in order to compare the effect of a special set of processes on the parameter of interest. Normally this feature is reserved for single lot reports only.

Box Only

Output and storage of the phonebook reports is the most resource intensive portion of the system. In particular, the I/O required for outputting the cumulative probability plot can take more time than the processing and output of all 3 other phonebook plots combined (some of this is reduced by graphing the plot at wafer level, rather than site level). Nonetheless, many times we need additional box plots with new features; rather than reoutputting the entire phonebook where only the box plot would change, we created the option to create the box plot only as a report type (figure 3).

Key additional features of this report are the ability to specify a “colorby” parameter. A categorical variable (i.e., equipment used at a critical process step) can be extracted from a SQL statement in the rules file (as noted in an earlier section), merged into the main SAS data set and specified as “colorby”. This is extremely useful when we have a particularly sensitive set of equipment that affects multiple parameters, and we want to quickly monitor the effect it is having.

Pipeline

Pipeline reports are a very high level snapshot of the data being analyzed. Output is a box plot in an HTML frame, where each box is one parameter (figure 4). Y-axis scale ranges from ~200% to -200% (100% is the upper spec limit for that parm and -100% is the lower spec limit) - ideally you would like to see the data for each box within the 100% to -100% range, meaning that the data is all within spec limits.

Pipeline reports are also hyperlinked; clicking on the cross (mean for that parameter) will drill down to the phonebook for that parameter, where x-axis will be lot or wafer, depending on whether you started in multi-lot or single lot pipeline report.

Since a complete set of parametric reports at the end of the process may have 70-100 parameters, one pipeline report would be very crowded if all parameters were shoved into one image. The spec file contains a field for pipeline categorization, such as “NMOS”, “PMOS”, “CT/VIA”, “METAL SHEETS” and so on. The initial link to the pipeline reports is a cover sheet with links to the many different categories.

Cpk Reports

Cpk is defined as the minimum of (mean-LSL)/3σ and (USL-mean)/3σ, where LSL and USL are the upper and lower spec limits for a parameter. Volume semiconductor manufacturing depends on the ability to produce millions of chips the same way

every time, and Cpk is one of the key metrics used to judge the stability of the fab. Cpk reports have a detailed text output and a simple GIF of Cpk trend (figure 5). Data can be specified to be grouped by lot, week, month, qtr, etc... Text report details mean, standard deviation, Cpk, skew and total fail percentage (TFL%) for each parm and each group (lot, month, etc.). GIF output is a stacked bar chart trending Cpk by group. The stacked bar consists of the number of parms for a particular group with Cpk less than 1, between 1 and 1.5 and greater than 1.5. (Parameters with Cpk greater than 1.5 are considered well centered in between spec limits and with a reasonably tight distribution – parameters with Cpk less than 1 need help...)

A more recent version of the cpk report has been released, that contains Cp, in addition all the standard statistics above, but in HTML format (rather than text), and summarized by lot, week and month as defaults, all in one report.

Yieldscrap

A subset of parameters tested are critical to device performance and reliability; wafers are scrapped if they do not meet specified criteria for these parms. These scrap parameters are specified in a technology definition document laying out the upper/lower spec limits and the number of sites per wafer that can fail before scrapping the wafer. Typically transistor, isolation and sheet resistances are scrapped if 3 sites per wafer fail (out of 9). Vias, contacts are scrapped on 1-2 sites and GOI is scrapped with only 1 fail. The spec file specifies if a parm is to be scrapped on n sites, or the value is set to 0 if the parm is not a scrap parameter. Additional complications are included by retesting wafers that are to be scrapped to ensure that the failures are not test related problems; results of both initial and final probes must be considered to get accurate reports of wafers scrapped.

Yieldscrap reports are all done in HTML frames and table (figure 7,8,9). Sections of the report include summaries of initial probe, final probe and the combination of the two, as well as a disposition screen. For multilot yield reports, GIFs trending lot yield by lot, week and month are also output. For single lot reports, details of failures on scrapped wafers are also output for quick analysis and debug – failed sites are highlighted in red.

PCD Reports (General Text)

These are the oldest reports, probably now an evolution of an upgrade of an old version from some VAX/VMS system in the early 90's... They are generated exclusively for single lot reports and are a simple text output of mean, standard deviation, Cpk, skew, TFL%, as well as a simple ASCII distribution bar. Parmes are all grouped into categories, as in the pipeline reports. Reports are output by lot, by wafer, by site and (if applicable) by split. PCD reports are useful for copying simple, text only summaries to some other program, or for quickly comparing differences between the different levels (say if site 2 & 3 had significantly different distributions from the other sites for some key parms).

Link to Download Data to Interactive Tool

Static reporting and correlations have their place in engineering, but typically only serve as the first line of defense to highlight a problem, not to give you the level of detail and compilation needed to drive an issue back to the root cause.

A link to a Windows clients – such as Spotfire, JMP or Excel – would be a logical step for further ad-hoc analysis. In the absence of a true SAS Client/Server environment (such as SAS/Intrnet) capable of providing data on the fly to those

clients, we decided that SAS/Bar would generate an XML file containing enough elements to recreate the data environment and deliver it locally on demand. SAS/Bar static web output now contains a button capable of activating a Javascript which presents the users with choices allowing a selective download into a local application called Spotfire (see more details below). Because the current SAS8.2 XML engine is limited, we wrote our own SAS datasteps to create this XML file (figure 6). (With the expected delivery of a powerful XML engine in SAS 9.x, we will most likely convert the request file and the specs file to XML format as well.)

A software package called Spotfire is our (latest) implementation of a customized commercial solution for interactive engineering data analysis. Though short on statistical prowess, the strengths of the tool are the ability to filter and display any data any way conceivable very quickly and easily. The Windows client can log onto a server to access a suite of “workflows” to extract data from scratch, add data to an existing data set (from any database or data source), run a commonality analysis. A custom workflow for accepting a Javascript launch from an HTML button on the phonebook was developed to aid with importing of data from the static SAS reports to the dynamic workflow environment, bypassing the manual entry of lot numbers and parameters of interest and further increasing engineering efficiency.

Putting It All Together – An Analysis Example

One possible train of analysis might go like this: I check the previous month to date reports to look at the yield trends (by lot or by week). I notice a slight rise in the number of wafers scrapped recently for PMOS I-drives. I next pull up the Cpk's for the same time frame and notice a drop in Cpk and TFL% for both NMOS and PMOS I-drives; this month's median is somewhat less than the previous month. From there I decide to zero in on the previous week to date reports. I pull up the pipelines for transistor parameters and note that several other transistor parms are also giving consistently lower (or “colder” I-drives); clicking on PMOS I-drive, I see the trend by lot of this parameter in the box plot in the phonebook. The last 4 lots tested show significantly lower values than the other lots; clicking on one of the low lots, I see that all wafers are low, not just most or a few. Since all parameters seem to be consistently drifting “cold”, I check the reports with the gate measurements for these lots. All are showing wider than normal dimensions, which is causing the trend in colder transistors. Next I check the current trends for gate test to see what is currently going on in that part of the process. Current material seems to be on target, but I notice a grouping of 8 lots processed earlier that have wide CDs (meaning I don't have to shut the gate process down). Additional lots with wide CDs are flagged as cold, and the raw data behind the report is pulled into Spotfire with 1 click on the report. Flagging a sample of lots as “Hot” and “Cold”, I run a good/bad commonality analysis, looking for a process step that has multiple equipment options where all the “Bad” lots ran on one tool and all the “Good” lots ran on the other tools. I come up with a hit at gate poly etch, and contact the equipment and process engineers to go through details on the excursion with them.

SAS Inside – Nuts & Bolts of the Code and Necessary Files

SAS/BAR is composed of a series of complex SAS macros and uses the following SAS Products and components:

- SAS/Access to Oracle, both with SQL Pass-Thru when performance is an issue, and LIBNAME statement with the Oracle engine
- SAS/QC, in particular Proc Shewhart for the phonebooks and pipeline charts, Proc Capability for Capability Indices calculations and Proc Reliability for Probability Plots in the phonebooks.
- SAS/Graph, in particular Greplay and Annotate to assemble various graphs in one page and Gmap to produce Wafer Maps.
- ODS to generate WEB enabled output, and display traffic lights in Proc Report (PCD).

One of the main requirements in the initial system design was to allow **batch unattended execution**, to spare engineers the precious time necessary for data extraction and report generation. The system delivers that, and while it is possible to run it real time, the majority of its use is via scripts triggered through a scheduler. The downside of this design is that interactivity is limited, and therefore the flexibility associated with drop-down lists for on the fly choices or exception handling does not exist. However, the system covers what it was intended to, and the ROI has been tremendous.

The interface to the user is a *Request File*, which captures the detailed elements of the user request, such as:

- The destination of the output.
- Which population to extract (test programs, time frame, single lots or multiple lots, etc.)
- Which specs file to use
- The reports to generate for that extract and their options.

The request file is the key file in setting up a report to run, and uses a series of keywords followed by an =, to take advantage of SAS named input. The structure of this file was designed such that someone with virtually no SAS experience could modify and customize a request file to meet their needs, and submit with a very simple 2-3 line SAS file. Please see Figure 10 for an example of the request file.

“Line=1” is designated for job setup – location of the specs and rules file, output destination, date range and/or population of lots to be included in the analysis. Dates can be explicit (01/01/03-06/01/03), or relative (5_weeks_ago, 0_months_ago); lot population can be a single lot, a list of lots or an integer N to select the last N lots tested.

“Line=2” specifies the test area and program; if multiple test areas and program are specified, the first entry defines the population of lots to be analyzed, and data from the second program will be pulled for those lots.

“Line=3” are the specific report types; phonebook, cpk_trend, yieldscrap, pipeline, pcd, etc... Multiple options include: baseline wafers only (Y/N), tukey screening applied (Y/N), use rules file (Y/N), break (month/week/lot/wafer/split/etc...), custom output subdirectory, output type (HTML, GIF, PDF). In addition, custom lists for programs, test areas and a specific lot list can be defined in “line=0” entries.

Although the request file comprises the bulk of the information needed to run the report, SAS still needs a **SAS** file to execute.

If the request file was /tmp/request1234.txt, then the SAS file required to execute the report would be:

```
filename request '/tmp/request1234.txt';
%request;
```

Various system variables (like database paths), and options like mprint can also be specified in the small SAS file.

In addition, the advanced user can insert their own SAS code in a *Rules File*, which is %Included during the extraction process (figure 12). Last, but not least, a *Specs File* must be supplied, with information on the variables to analyze: variable name, description, lower spec limit, target, upper spec limit, pcd category, number of sites per wafer scrap criteria, Cpk parameter (0/1), error code screening (0/1), pipeline category (see Figure 11).

In addition to the reports, the system saves the extracted data in the form of views, which can be retrieved at a later time for further analysis without complex coding.

The whole system of SAS macros provide wonderful capabilities for pulling data and crunching it into the forms we require; however, if it were required of each engineer to actually log onto the UNIX SAS box, create a request file for the data they needed and execute it with SAS then a) nobody would do it (except for a very determined few) and b) those that did would have to wait up to an hour for their reports to finish. The whole system was designed to allow a few simple perl scripts to generate the request and SAS files needed for all the key reports, and automatically generate varieties of them at given intervals. The perl programs are in turn controlled by configuration files that give the parametric test programs that need the data analyzed, along with a few key fields for report paths (which are portions of the device and technology designators). Reports are, in our facility, generated for last week to date, this month to date, last month to date and last 50 lots (for multi-lot reports), as well as single lot reports for each lot tested. Each of these variations has a perl program which reads the config file (with the list of test programs to generate reports for), and for each entry constructs the line=1, line=2 and line=3 entries (line=3 entries are standard options for standard reports, so everything gets generated the same way every time). In addition, the small 2 line SAS file is generate, and system commands are given to execute the SAS file. A master crontab file controls execution of these perl programs; last week to date and this month to date reports (for every test program) are regenerated on a daily basis, while last month to date and last 50 lot reports are generated twice a week.

Parametric Summary Database

The latest major addition to the system is the schema design, construction and loading of a parametric data summary database. Basically, instead of custom reports having no choice but to pull raw, site level data we can now pull key statistics for each parm at wafer level (data reduction of 9 to 1, minimum) or lot level (data reduction of 9*24, or 216 to 1). Stats calculated are standard: N, min, max, mean, median, standard deviation, Q1, Q3, P5, P95, Cp, Cpk, skew, % above upper spec, % below lower spec. In addition, these calculations are done for both unscreened (raw) data and with Tukey screening limits applied.

General Observations in Analysis System Design

Use programming contractors! Neither IT nor end customers (product engineering) can keep it all up alone... Let the contractor handle main technical package, and then let product engineering (hands on customer) handle implementation and details.

Begin with the end in mind. Think about what the end solution should look like, and go from there. Know (anticipate) your customer's requirements up front.

Don't try to do everything with SAS; it has its place, so do other elements. Use its strengths, circumvent its weaknesses.

Get plenty of horsepower – disk space and CPU cycles are cheap! Delays to analysis are not.

Conclusion

This is far from a complete and comprehensive analysis system (which is essentially impossible or impractical given the complexity of semiconductor processing and the near infinite number of things that **could** go wrong). However, much of the need for a basic first pass check on the process and simple diagnosis of problems is met with this reporting system. The flexibility of the data extraction to be able to coalesce and merge different facets of data is extremely useful and saves a lot of engineering time from trying to do complex joins manually.

The multiple levels of reporting (single lot, multi lot, last week to date, last month to date, last 50 lots, etc...) enables quick snapshots of as much or as little data as the engineer requires. Pregeneration of the reports using perl and a network of scheduled cron jobs allows reports to be setup and then generated automatically with a minimum of maintenance. The fact that they are already pregenerated is a huge time savings; most users have no in-depth knowledge of SAS and need none. Even engineers who run custom reports can do so without anything more than just the basic knowledge of how to edit a file, submit a SAS job and read a log file. All this combines into a system that allows engineers to spend a maximum amount of their time looking at the data and solving problems, not trying to extract, manipulate, screen, crunch and visualize the data.

Acknowledgements

Special thanks to Pamela Wang, Foong-Ying Rousey, David Aldrich, Suresh Subramaniam and Rod Threadgill for their help in the development of this reporting system.

Contact Information

Tom Winter: t-winter@ti.com, (858) 675-0253
Willy Waks:

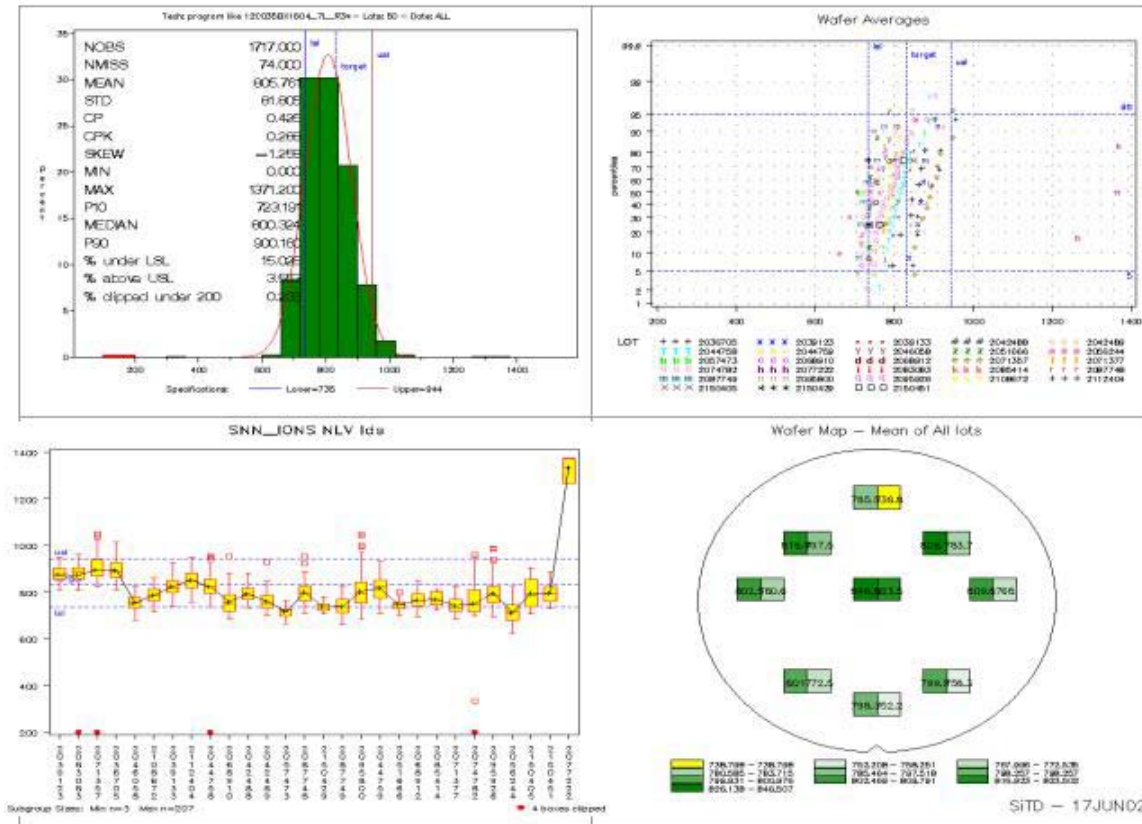


Figure 1 - Phonebook HTML graphic

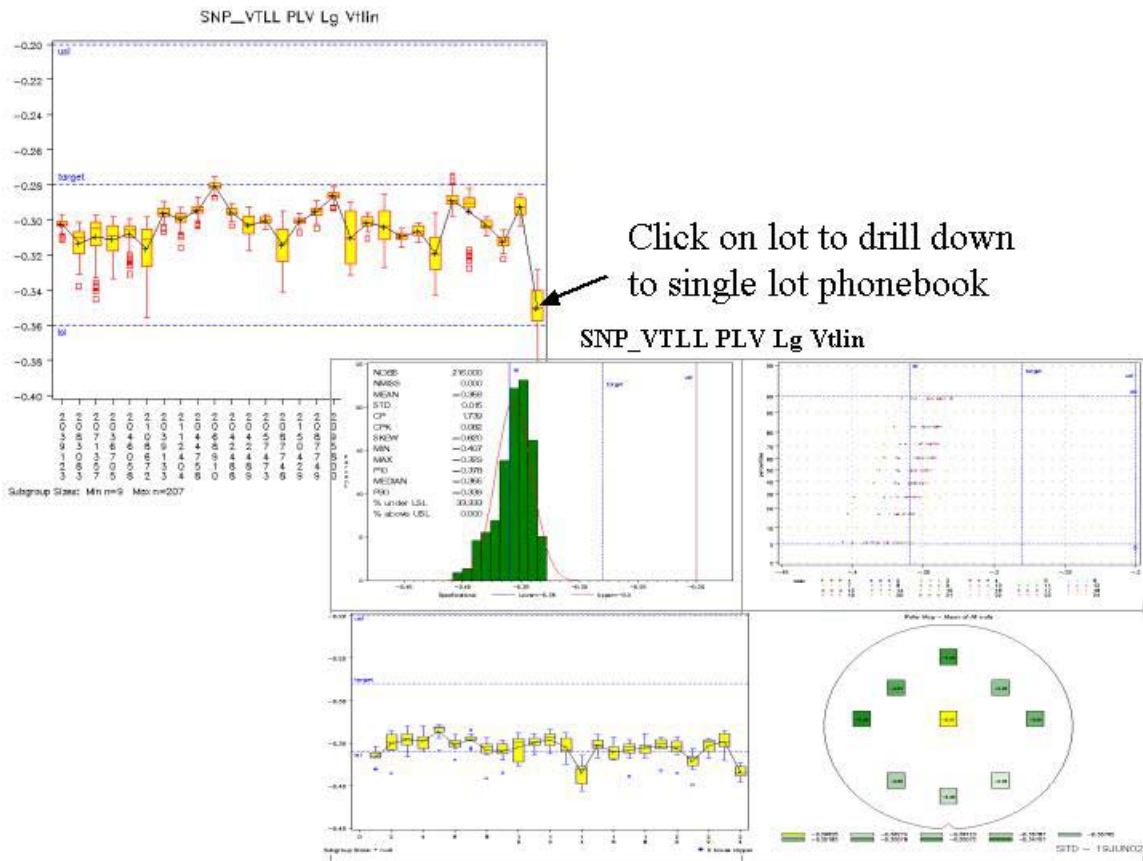


Figure 2 - Multi lot to single lot phonebook drill down

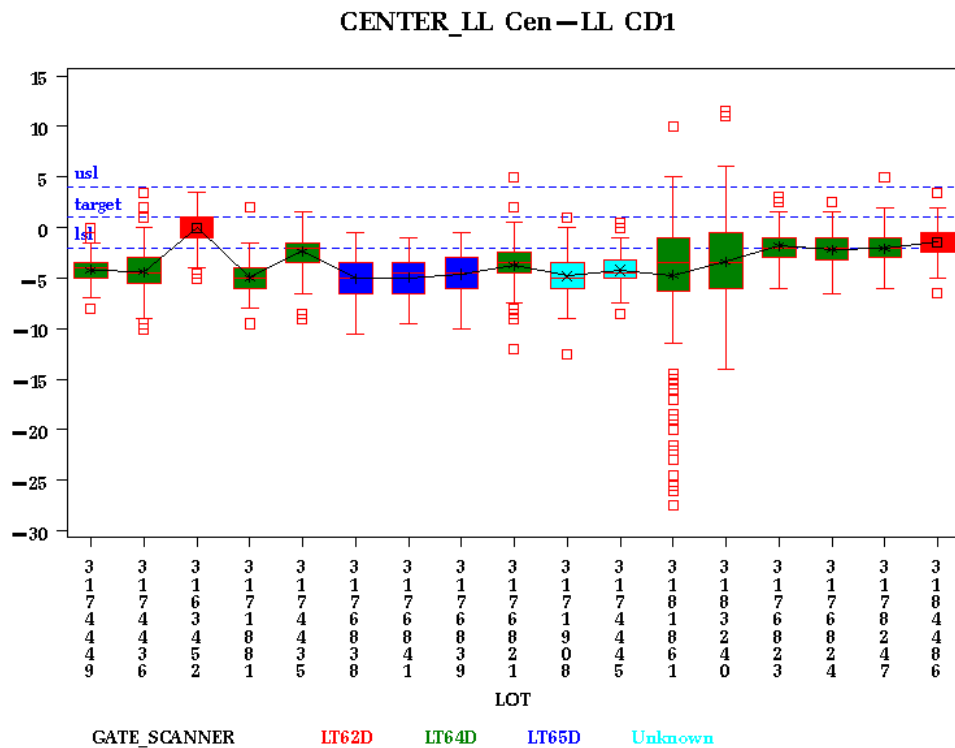


Figure 3 - Report box_only, colored by gate scanner equipment

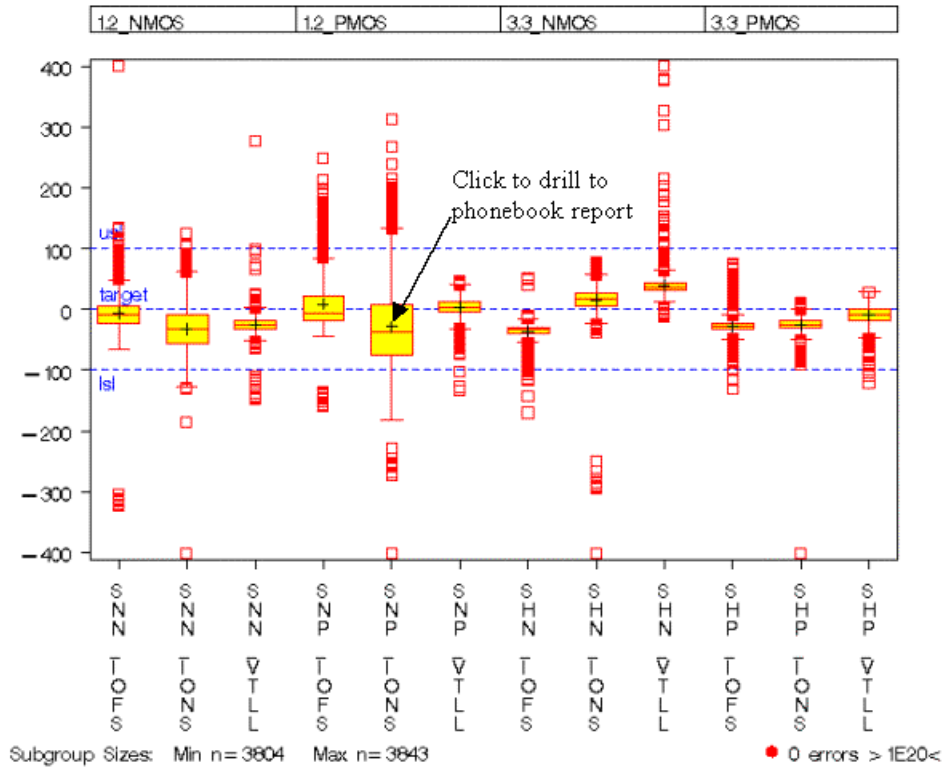


Figure 4 - Transistor pipeline graphic

Tech: program like 1233C035AX5576_6L* - Lots: 50 - Date: ALL

CPK SUMMARY REPORT - by month - Baseline only

Testdate ranges from 03/01/2002 to 06/14/2002

**** Report generated by SiTD Testing on 16JUN02 18:25 ****

Order by CPK												
SAS_VAR	MODULE_NAME:DUT:TEST	Month	MA	MB	TARGET	MEAN	STNDV	LSPEC	USPEC	CPK	SKEW	TFL*
U2K_NOM	Via2 VdP	JUN02 8385	8275	1.5	0.7106	0.057	0.5	3	1.24	0.98	1.31	
U1K_NOM	Via1 Sq. VdP	JUN02 8382	8341	1.5	0.7171	0.055	0.5	3	1.31	0.84	0.49	
U3K_NOM	Via3 VdP	JUN02 8385	8377	1.5	0.7518	0.061	0.5	3	1.37	0.67	0.10	
U1C_NOM	Via1 Sq. Ch	JUN02 8387	8328	1.5	0.7655	0.063	0.5	4.5	1.41	1.09	0.70	
RS_NGT_N	PPOLY RSH	JUN02 8383	8310	10	6.3744	0.323	5	15	1.42	0.82	0.87	
U3C_NOM	Via3 Ch	JUN02 8388	8340	1.5	0.9291	0.101	0.5	4.5	1.42	1.09	0.57	
RS_NTANK	NUELL Res	JUN02 8388	8385	540	498.73	8.471	460	520	1.52	-1.26	0.04	

Tech: program like 1233C035AX5576_6L* - Lots: 50 - Date: ALL

CPK text

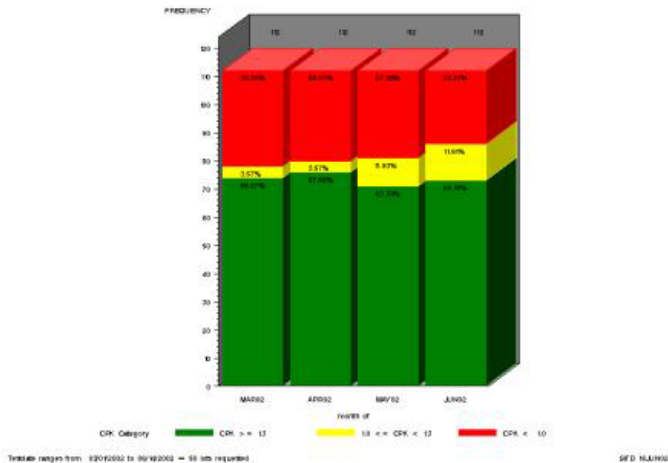


Figure 5 - Cpk text and gif reports

```

<?xml version="1.0" standalone="yes" ?>
=<Last_50>
  <JOBINFO>pid="2941" job_date="16JUL03:17:17:02" plan_consol="1118C021K"
    device="X1704" program_family="1118C021KX1704_1L"</JOBINFO>
  =<AREALIST>
    =<AREA id="NONE" test_area="PARAMETRIC" test_program="1118C021KX1704_1L_R00">
      <LOT lot="3097822" lot_seq="289552" lot_end_time="11APR2003:23:00:44" />
      <LOT lot="3127640" lot_seq="298815" lot_end_time="20MAY2003:05:22:53" />
      <LOT lot="3154747" lot_seq="303054" lot_end_time="04JUN2003:00:34:55" />
    </AREA>
  </AREALIST>
  =<REPORTLIST>
    <REPORT num="1" report="cpk_trend" break="month" rules="Y" baseline="Y" tukflag="Y"
      rep_subdir="cpk" />
    <REPORT num="2" report="cpcpktrend" break="" rules="Y" baseline="Y" tukflag="y"
      rep_subdir="cpcpk" />
  </REPORTLIST>
  =<SPECCLIST>
    <SPEC macname="cap" no="1" var="SNN_IIONS" desc="1.1V NMOS Ids" calc="N"
      cat="1.1NMOS" pipeline="XTOR" scrap="3" lsl="540" target="675" usl="817" reports="1,2" />
    <SPEC macname="cap" no="5" var="SNN_VTL" desc="1.1V NMOS Vtn" calc="N" cat="1.1NMOS"
      pipeline="XTOR" scrap="3" lsl="0.29" target="0.37" usl="0.45" reports="1,2" />
  </SPECCLIST>
</Last_50>

```

Figure 6 – Sample XML file for link to Spotfire

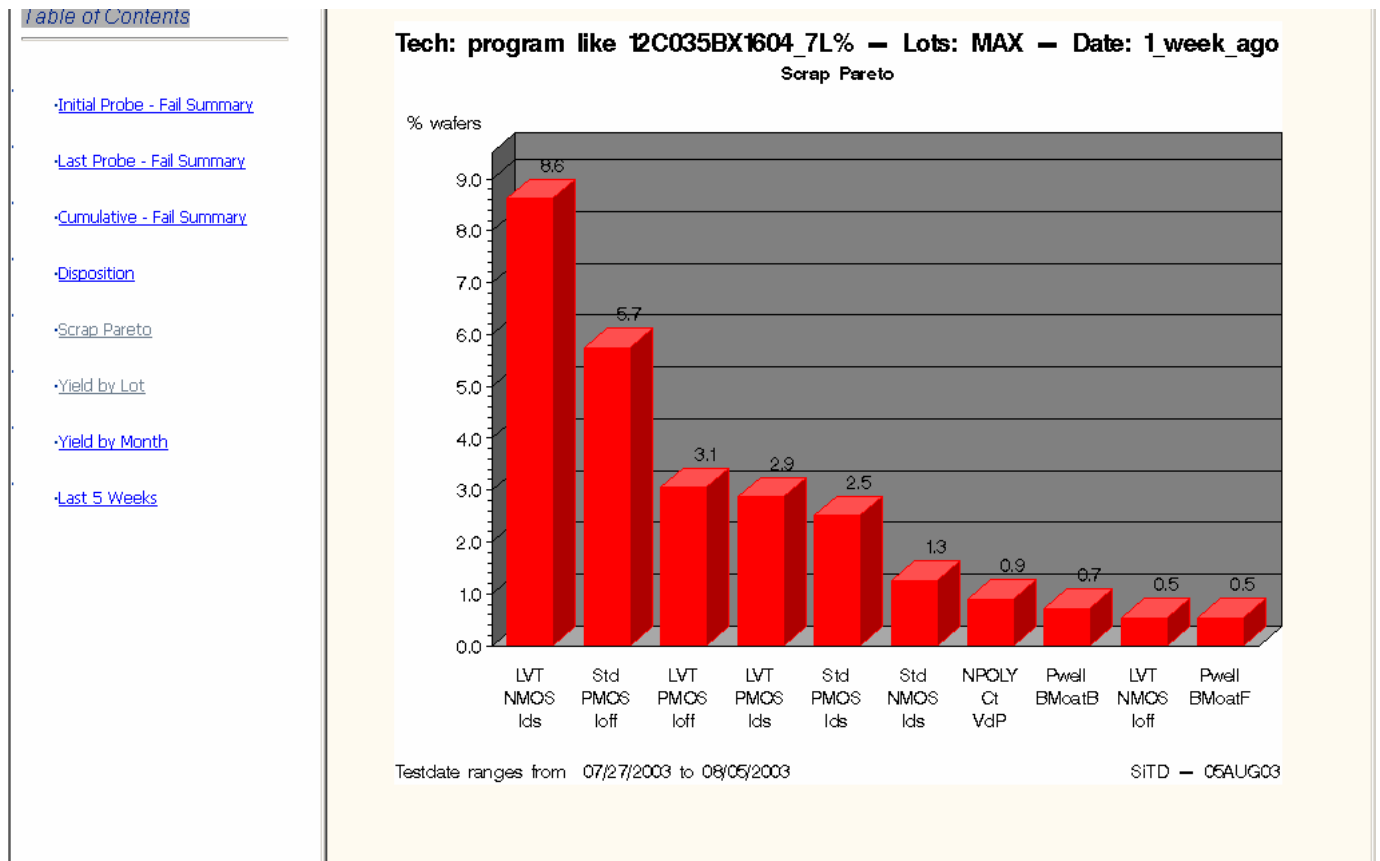


Figure 7 – Multilot yieldscrap pareto chart

[Table of Contents](#)

- [Initial Probe - Fail Summary](#)
- [Initial Probe - Fail Pareto](#)
- [Initial Probe - Detail Failures](#)
- [Last Probe - Fail Summary](#)
- [Last Probe - Fail Pareto](#)
- [Last Probe - Detail Failures](#)
- [Disposition](#)
- [Scrap Pareto](#)

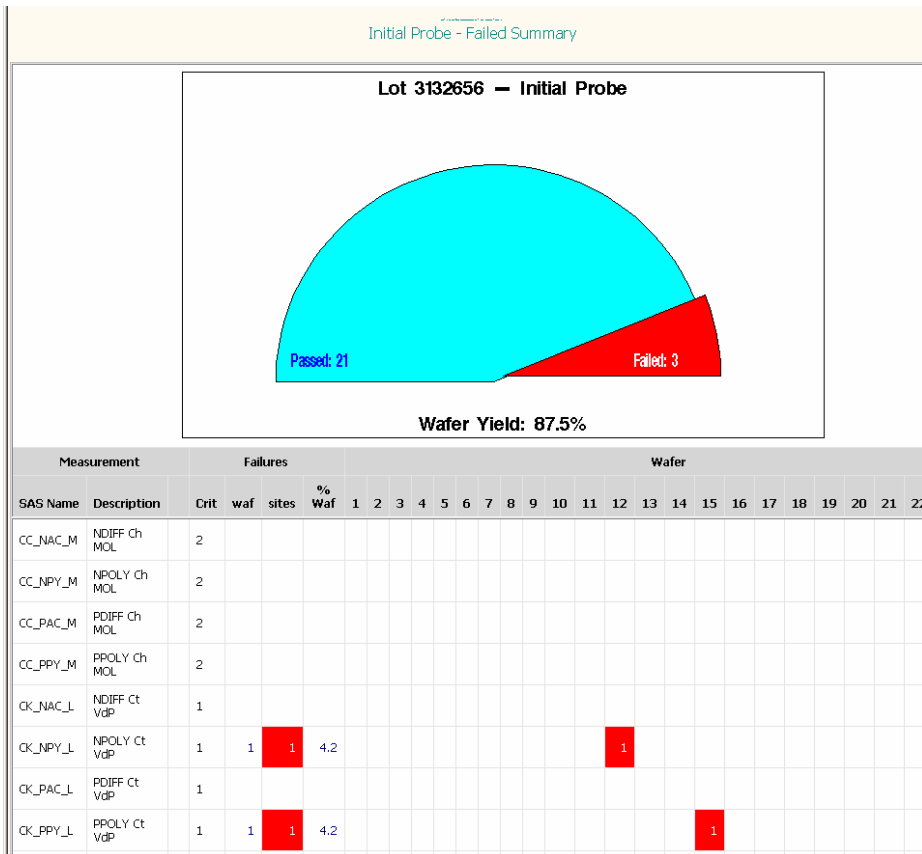


Figure 8 - Single lot yield scrap report – summary view

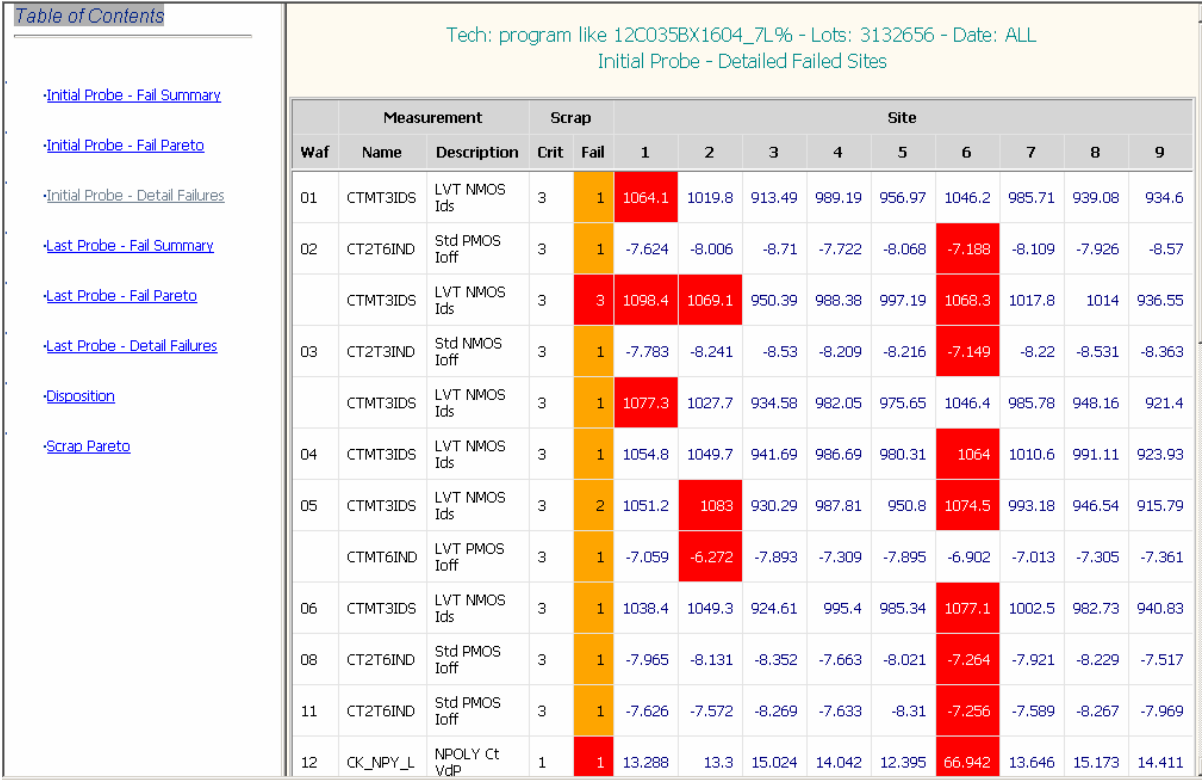


Figure 9 - Single lot yield scrap report – detail failures view

```

line=1
Rep_dir=/tw/kfab/C035/15C03515/X5986/PARAMETRIC/15C0351X5986_5L/single_lot_reports/2018470 /
rules_path=/tw/kfab/C035/15C03515/X5986/PARAMETRIC/15C0351X5986_5L/spec/rules.sas /
spec_rev=/tw/kfab/C035/15C03515/X5986/PARAMETRIC/15C0351X5986_5L/spec/specfile /
dte_rnge=ALL lots=2018470
line=2 area=PARAMETRIC tech=15C0351X5986_5L
line=3 report=cpk_trend break=lot /
rules=Y baseline=N tukflag=Y rep_subdir=cpk
line=3 report=phonebook dispmode=nodisp gdevice=html /
rules=Y baseline=N tukflag=N rep_subdir=phonebook
line=3 report=phonebook break=cat rules=Y tukflag=N baseline=N gdevice=HTML
line=3 report=phonebook dispmode=nodisp gdevice=html /
rules=Y baseline=N tukflag=Y rep_subdir=phonebook_screened
line=3 report=pipeline break=cat rules=Y tukflag=Y baseline=N /
gdevice=HTML rep_subdir=pipeline_screened
line=3 report=rep_tw break=lot rules=Y baseline=N rep_subdir=pcd
line=3 report=rep_tw break=waf rules=Y baseline=N rep_subdir=pcd
line=3 report=rep_tw break=site rules=Y baseline=N rep_subdir=pcd
line=3 report=yieldscrap rules=Y baseline=N gdevice=HTML

```

Figure 10 – Sample single lot request file

Name	Description	LSL	Target	Usl	Category	Scrap	Cpk	Tukey	Pipeline
SNN_IONS	NLV Ids	627,744	877	1.2	NMOS	3,1,1	NMOS		
SNN_IOFS	NLV Ioff	-10.1,-8.7	-7.7	1.2	NMOS	3,1,1	NMOS		
SNN_TOX	NLV Toxacc	20.5,22	23.5	1.2	NMOS	0,1,1	NMOS		
SNN_VTLL	NLV Lg Vtlin	0.14,0.22	0.3	1.2	NMOS	3,1,1	NMOS		
SNP_IONS	PLV Ids	244,297	360	1.2	PMOS	3,1,1	PMOS		
SNP_VTLL	PLV Lg Vtlin	-0.4,-0.32	-0.24	1.2	PMOS	3,1,1	PMOS		
SNP_IDL	PLV Idlin	55,63	72	1.2	PMOS	0,1,1	PMOS		
SNP_TOX	PLV Toxacc	19.5,21	22.5	1.2	PMOS	0,1,1	PMOS		

Figure 11 – Excerpt from sample spec file

```

/*****
/* Global Rules to Remove Extraneous Probe Sites */
*****/
if site in ('10','11','12','13','14','15','16','17','18') then delete;

BETA=SNN_IONS/SNP_IONS;
N2P=SNN_IONS + 2*SNP_IONS;

libname ORADB oracle user=smsuser orapw=smsuser path="@KFABsms";

proc sql noprint;
  select distinct LOT into :mylots separated by ','
  from one;
  create table gatestep as
  select distinct LOT, LOGPOINT, LOG_DATE, EQUIP_ID as gate_scanner, ITEM_ID
  from ORADB.lotequip_data where LOGPOINT='3290' and OPERATION='3500' and
  LOT in ("&mylots");
  create table gateetch as
  select distinct LOT, LOGPOINT, LOG_DATE, EQUIP_ID as gate_etcher, ITEM_ID
  from ORADB.lotequip_data where LOGPOINT='3434' and OPERATION='4500' and
  LOT in ("&mylots");
quit;

data one;
  merge one gatestep(in=in2) gateetch(in=in3);
  by lot;
  if not in2 then gate_scanner='Unknown'; /* or if gate_scanner=' ' */
  if not in3 then gate_etcher='Unknown';
run;

```

Figure 12 – Sample rules file