

Multi-Level Web-Based Reporting System for Semiconductor Parametric Data

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

Abstract

Early semiconductor process development is especially dependent on parametric data; 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 text reports & gifs for cpk. 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-developed 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. Many 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.

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.

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 device; "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 grab 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.

Report Details

Phonebooks

"Phonebook" reports are a snapshot combination of a histogram, cumulative probability plot, box plot and wafer map. 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 and displayed as a large GIF. 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.

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. 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.

Phonebooks have the additional capability of plotting split number on the x-axis. Normally this feature is reserved for single lot reports only.

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. 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 $(\text{mean}-\text{LSL})/3\sigma$ and $(\text{USL}-\text{mean})/3\sigma$, 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. 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...)

Yieldscrap

A subset of parameters tested are critical to device performance and reliability and wafers are scrapped if they do not meet certain criteria. 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. 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.

PCD Reports (General Text)

These are the oldest reports, probably now a 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. Parns 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 parns).

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 parns 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 further work is then initiated to find the root cause of the wider gate dimensions.

SAS Inside – Nuts & Bolts of the Code

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 on-line, 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 uses a series of keywords followed by an =, to take advantage of SAS named input.

In addition, the advanced user can insert their own SAS code in a *Rules File*, which is %Included during the extraction process. Last, but not least, a *Specs File* must be supplied, with information on the variables to analyze, in particular one or two-sided specs limits, target and other attributes.

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.

All the reports for different devices in the fab are auto-generated with preset options via a set of configuration files and perl programs that read them in and generate the necessary SAS files and executes them. The configuration files essentially specify key fields about the device and test programs to retrieve and report the data for. The spec file and rules file for those programs have already been setup by the engineer responsible.

The perl programs generate the appropriate code for the reports desired in temporary files and execute the SAS programs; perl is the language of choice for this task because of its ease of manipulation of text and files, suitability for programming in a UNIX environment and general knowledge among many engineers.

Future Improvements

One of the future improvements planned for the system are HTML "meta-tags" on phonebook reports that would allow one click access of the raw data behind the graph to be pulled into Spotfile (a general purpose interactive analysis tool we have licensed – very flexible, very user friendly).

Another planned improvement is a general purpose box chart, where data can be plotted by machine, by week, by any categorical variable; expanded flexibility in the rules file now allows user defined pull of data from any other source and merging with the original parametric dataset.

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 working 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 (972) 995-3996

Willy Waks: wwaks@swbell.net (972) 661-8619

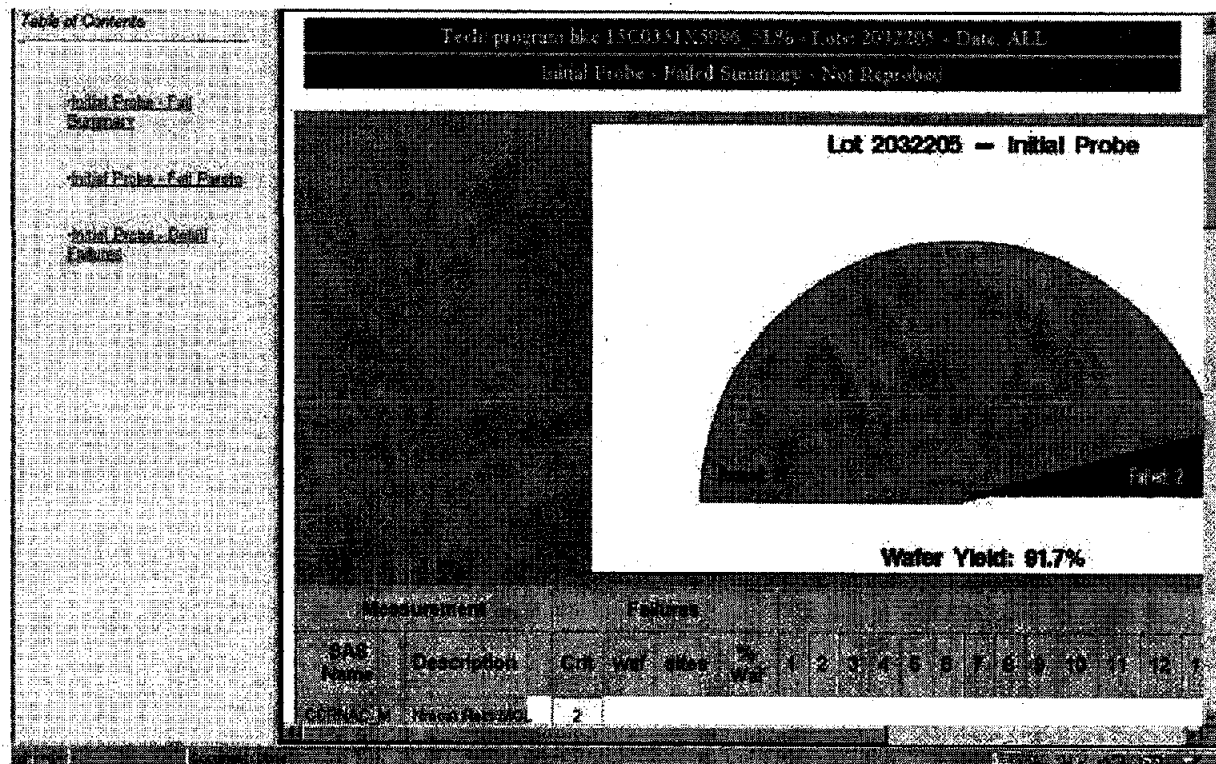


Figure 5 - Single lot yield scrap report

```

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_mge=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=pipeline 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 6 - 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 7 - Excerpt from sample spec file