FabTime Cycle Time Management Newsletter

Volume 10, No. 5 June 2009

Information

Mission: To discuss issues relating to proactive wafer fab cycle time management

Publisher: FabTime Inc. FabTime sells cycle time management software for wafer fab managers. New features in this month include a new compliance chart to show BeginRun transactions that are not matched with a Dispatch transaction and support for lot-level planned x-factors on Forecast Outs charts.

Editor: Jennifer Robinson

Contributors: Philippe Vialletelle (STMicroelectronics)

Table of Contents

Welcome

Welcome to Volume 10, Number 5 of the FabTime Cycle Time Management Newsletter! We hope that you're all (in the northern hemisphere, anyway) having a great summer. In this issue, we have two brief community announcements, and a response from a subscriber to two previously introduced discussion topics (dispatch precision and tool state reporting). Our FabTime software user tip of the month is about using the alert functionality in FabTime to send alerts to other people from your team.

In our main article this month, we discuss problems that can stem from broken assumptions. Whenever you implement a series of steps, whether this is in software code, a spreadsheet, or an operational process in a fab, you make assumptions along the way. Often these assumptions seem so obvious that you don't even document them, let alone plan for them to be broken. But of course sometimes they do break. When that happens, the root cause is often difficult to identify. We decided to open up a dialog with our newsletter subscribers on this issue of broken underlying assumptions. We welcome your feedback.

Thanks for reading!—Jennifer

FabTime

Tel: (408) 549-9932
Fax: (408) 549-9941
www.FabTime.com
Sales@FabTime.com

Community News/Announcements

Next Fab Owners Association Meeting

The next Fab Owners Association meeting will be held in conjunction with Semicon West, at Moscone Center in San Francisco on July 16th. FabTime's Jennifer Robinson will be attending the associate member sessions of the meeting.

Call for Papers for AEC/APC Asia 2009

The AEC/APC Asia conference will be held November 9, 2009 in Tokyo, Japan. The abstract deadline was recently extended to June 30th. Here are some additional details:

"International SEMATECH Manufacturing Initiative (ISMI) and International Symposium on Semiconductor Manufacturing (ISSM) invite you to participate in the AEC/APC Symposium Asia Call for Papers.

The AEC/APC Symposium, part of a series of such meetings held three times a year in North America, Europe, and Asia, brings together IC manufacturers and suppliers in an effort to accelerate the move toward more efficient and more intelligent manufacturing through data-driven and automated decision making.

AREAS OF INTEREST

1. Equipment and Process Fault Detection, Classification and Prediction
2. Data Management and Process Control
3. Metrology, Sensors, and Analytics
4. Fab-wide FDC and APC, Yield and Multi-fab Approaches
5. Standard and Roadmaps
6. Future Needs and Proposals

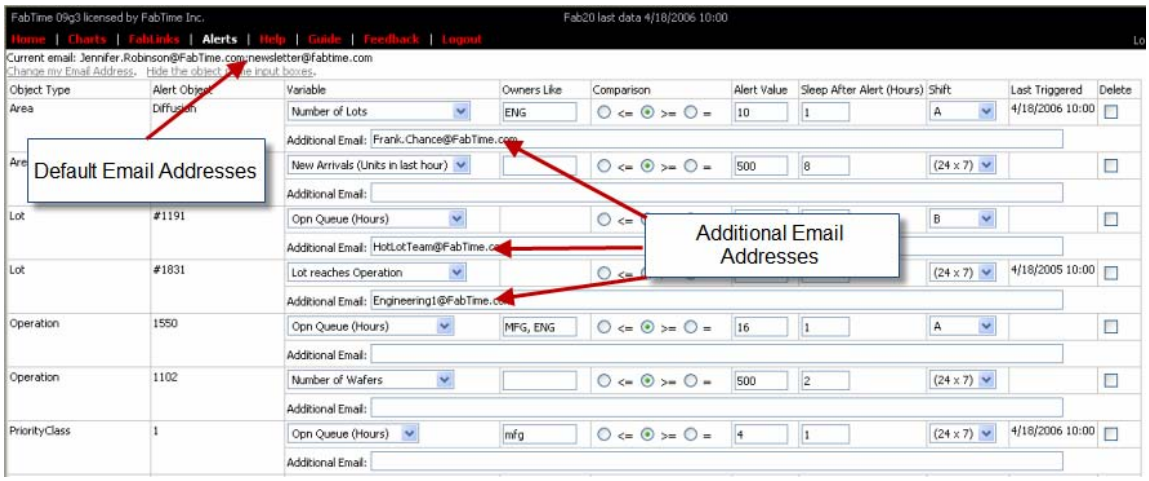
Complete information can be found on the AEC/APC Symposium-Asia website at http://www.semiconportal.com/AECAPC

FabTime welcomes the opportunity to publish community announcements. Send them to newsletter@FabTime.com.

FabTime User Tip of the Month

Send Alerts to Other People

If you've been using FabTime for a while, you've probably experimented at one point or another with setting alerts for yourself (short text emails that you receive when some condition that you specify is met). But did you know that you can also send alerts to other people? This is a newer option in FabTime, and one that we'd like to make sure people understand. Whenever you create an alert (using the Alert interface), that alert will be sent to you at whatever email address (or addresses) you have set up in FabTime. You can see those in the upper left-hand corner of the alert page. Beneath each alert that you create, you'll also find an "Additional Email:" field. This is where you enter other people's email addresses, if needed. Separate multiple email addresses with a semicolon. Just enter the email addresses, and click the Save button in the lower right-hand corner. You can also edit this "Additional Email" field for existing alerts. An example is shown at the top of the next page.



For example, you might:

- Create an alert to notify you whenever any Priority 1 lots wait too long, and copy the hot lot team on that alert; or
- Create an alert for when a particular tool is in a "standby-WIP-waiting" state, and copy the lead operator responsible for that tool;
- etc.

Just be forewarned: your user ID will be included as part of the alert text that's sent out. This means that the recipient will know that the alert came from you. Setting something up that sends hundreds of emails to your boss, well, we'll leave that decision up to you.

If you have any questions about this feature (or any other software-related issues), just use the Feedback form in the software.

Subscriber Discussion Forum

Dispatch Precision

Philippe Vialletelle, "Fab Expert" from STMicroelectronics responded to our recent discussion of Dispatch Precision, saying:

"I see Dispatch precision as a dream and this, for at least 2 reasons.

The first one is that the objective targeted through this indicator is often very fuzzy: what do you wish to measure and thus to improve? Is it the "discipline" of your operations? Is it the performance of your global system (i.e. lot transportation + storage + location + "scheduling" + tool loading)? Is it your dispatch rules?

The second one is that as long as you speak of dispatching, you try to balance two conflicting objectives. Indeed, the lot

you put on top of the list is very often there for reasons linked to customer service. The operator usually calls the list for one tool at a given point in time - when he wants to load the tool. The problem is that it may often be a better candidate on another tool just a few seconds later. Skilled, experienced operators often review what they have "on hand" before making any decision. They don't act in "real time", but rather balance the WIP over the available capacity, relying on information that may not even be in the system.

Last, but not least, in a "non-automated fab", constantly moving lots just before processing them because the dispatch decision was changed is not feasible.

So, you can't reach a 100% dispatch precision. The solution we found at STMicroelectronics was to "freeze" a variable planning horizon by tool type and to schedule the work for each tool periodically. Each tool was then allocated a list of lots. The operator's task was to bring the lots to the shelves in front of each tool. The compliance computation was then straight forward. If the lot processed is the first of the "schedule", it is compliant (inversions between same species / set-up / mask tolerated), otherwise not. A "hand-carried" lot is always compliant. Summing the total quantity of "compliant" wafers versus the total "moves" done gave the dispatch compliance. Last, but not least, the dispatch compliance was then introduced as a performance criteria in the computation of manufacturing incentives. Finally, the system allowed us to improve the scheduling engine: operator was given the possibility to justify each non-compliant decision. Today, scheduling compliance is constantly above 95%, the only exception being low WIP conditions (lots still not available at location)."

FabTime Response: It's great to have your insight and actual fab experience, particularly in regards to your dispatch precision metric. We like it – it's very simple and intuitive. A lot is either compliant or it's not. Of course there are always going to be people wanting to talk about degrees of compliance... But it's good to hear what works for ST in this area.

Tool State Reporting

Philippe also responded to a recent discussion on tool state reporting, saying: "The natural tendency of equipment engineering is to use EMCS for both execution AND reporting. This often leads to very heavy and complicated state models that do not fit any of their needs. From my experience, when you have more than 10 to 15 states (including sub-states!), execution becomes far too complicated. And you lose agility! Solutions exist and are at hand if you accept separating execution from reporting. Just think of the tons of data sent by automation. At one of our 200mm manufacturing sites, we developed a real-time reporting tool, mixing MES, EMCS and automation data that allowed us to detail tool status down to nearly 100 different sub-states. This tool is today the key driver for OEE improvement. Of course, it can't track states retroactively as "no operator". But, as it is real time, the state "idle with wip" is alarmed to manufacturing management who is acting accordingly. And anyway: if you have no operator, it's hard to ask an operator to log that there is no operator ;-)"

FabTime Response: We do agree with you that trying to manually log too many states is impractical. In regards to "idle with WIP" what we do in FabTime is use the WIP transactions to virtually log the tool to "standby WIP waiting". I completely agree with you that this has to be figured out automatically, and not rely on the operator to log it.

FabTime welcomes the opportunity to publish subscriber discussion questions and responses. Send your questions or comments to Jennifer.Robinson@FabTime.com.

Problems that Stem from Broken Assumptions

Introduction

FabTime's Frank Chance is the primary architect of our software (and co-founder of the company). Frank recently made some observations about what can happen when a process relies upon a hidden assumption that is later broken. Frank's thoughts on this were initially based on software development. However, we believe that this issue of broken assumptions is relevant also to operational processes in wafer fabs. We've decided to open this issue up to discussion.

Frank has been troubleshooting a variety of software-related issues with our customers. He observed that the root cause is frequently an assumption that we made when writing the code. This assumption was perfectly fine at the time. However, when the assumption no longer holds, problems arise. In many cases, these "broken assumption" problems are a) hard to troubleshoot, and b) perfectly obvious in hindsight.

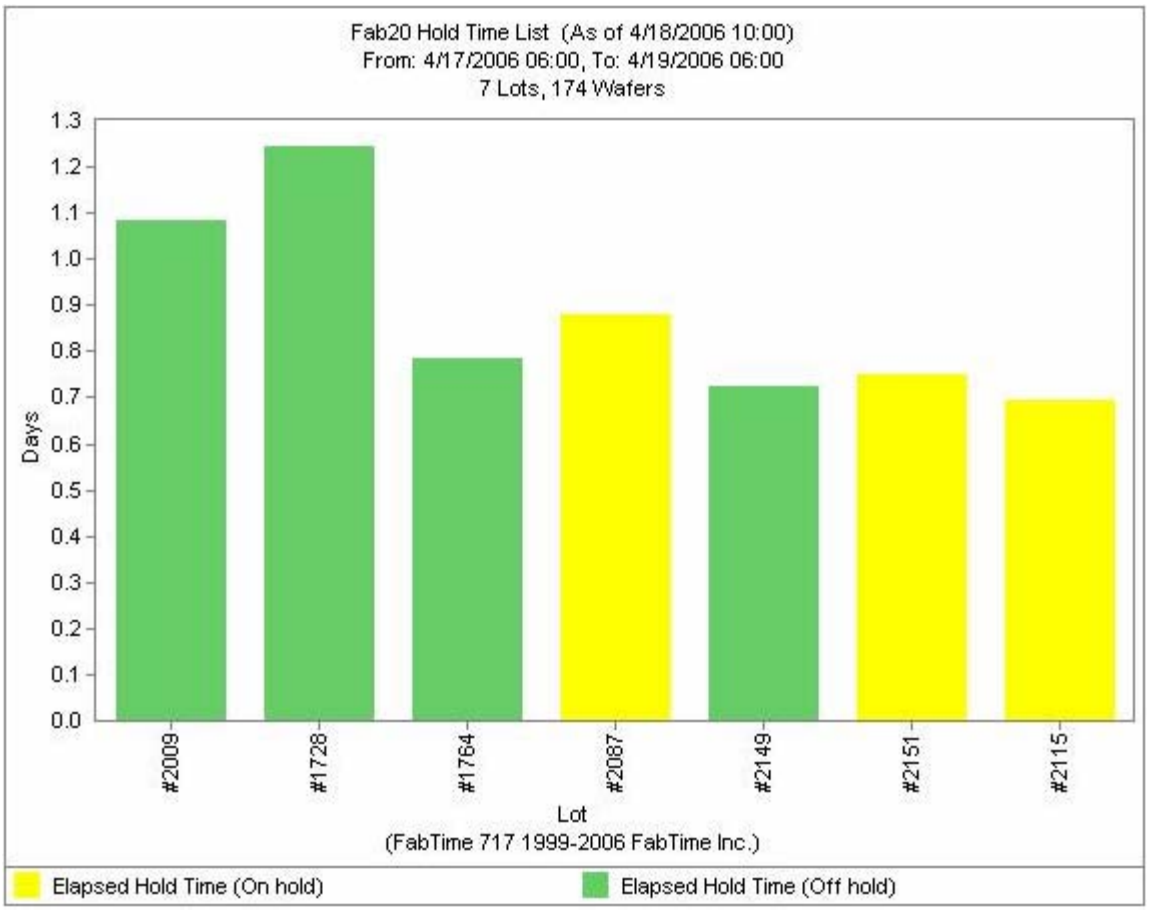
Here are several FabTime software development examples:

1) Recently Frank was validating WIP data for a probe site, and puzzling over a case where a lot wasn't properly reopened when it had post-close transaction activity in the MES. Lots can be reopened for a variety of reasons, a common one is the reversal/deletion of an incorrect merge transaction (e.g. a lot is recorded in the MES as being merged back into its parent, but in reality it was not merged and so the merge transaction is reversed in the MES). The root cause of the problem in this case was an assumption we made when writing the reopen code that reopened lots would have non-zero wafer quantities. From a fab point of view (all of our customer sites at the time the code was written were fabs), this assumption was quite reasonable – if a transaction comes through for a lot that has already been closed, and the wafer quantity is zero, then there is no reason to reopen the lot. But… if it's a probe site, some manufacturing execution systems will set the wafer quantity to zero. Only the die quantity is non-zero. And if the die quantity is non-zero, the lot should be reopened. Lesson learned – and code changed.

2) FabTime's hold time charts (example on the next page) list the number of distinct times that lots went on hold within each reporting period. Frank was investigating a customer report that this count was too low, and found that the code was incorrect because it assumed that the same lot would not have multiple holds of exactly the same duration (measured in minutes) within the reporting period. It might seem unlikely, but of course this has happened, and so we have changed the code to remove this assumption.

3) Out-of-sequence transactions are particularly good at exposing hidden assumptions. Last year Frank was troubleshooting a problem with lots not being properly closed in FabTime when they had been closed in the MES. He discovered that the MES was generating a close transaction that was timestamped prior to the last active transaction for the lot. In FabTime, we had made the assumption that a lot would be closed only after (time-wise) all of its transactions had been recorded. But this is not always the case, we learned.

Who would have thought that there could be valid post-close lot transactions where the wafer quantity is zero? Or that lots could be closed with a timestamp prior to the last processing activity? OK, what we've really learned over the past 10 years is that all sorts of unexpected data quirks arise in fabs. The trick is to make the software code robust enough to handle them. But there are always going to be assumptions made about what data flows



into FabTime from the MES, what should be included and what should not, and so on. And when these assumptions are no longer correct, troubleshooting is required.

More generally, of course, all processes, including business processes and operational processes, rely on assumptions. Sometimes those assumptions are so obvious that they are not even documented. Checklists and instruction sets for how to do things certainly help. We have written them for all sorts of FabTime activities, from sending out the newsletter to setting up a new server. Checklists cut down on some types of errors. But they seem more suited to execution-type activities (install a patch, respond to an equipment failure, etc.). We generally don't have a checklist when doing something new, when creating something from scratch, and that is where these hidden assumptions keep cropping up.

A possible example for fabs is the construction of a spreadsheet capacity model. When you build a model like this from the ground up, you have to make assumptions about how to treat rework, how process flow information will be stored and mapped to products, whether you're going to include scrap at every step, etc. When you have a spreadsheet model that's been in use for five years, and the person who initially built it is gone from the company, how likely is it that one of these assumptions, possibly one that isn't documented anywhere, is no longer true?

Here are a couple of potential operational examples.

- Because the average batch size in diffusion is nearly full, the fab sets a policy of waiting to run until the batch is full. This works well for a while. However, when the volume for the fab goes down, the assumption that running full batches is

optimal is no longer true. However, the policy remains in place.

- The fab sets cycle time goals based on the assumption of having no single path operations. Each tool group has at least two tools in it. The problem is, in one of the toolgroups, the second tool is in another part of the fab, and it turns out that the operators don't use that one. An undocumented "soft dedication" scheme is in place. The fab ends up with what is functionally two one-of-a-kind tools, and thus has higher cycle time than expected.

Then there are the "everybody knows" assumptions. "Everybody knows that we never run device X on tool Y at operation Z, even though it's qualified there." Or "everybody knows that we hold WIP at tool B until tool C is free, and then start processing, so that there's no waiting at tool C". But does the new employee know? Does the new dispatch system know? It's been our experience that the closer you get to the floor, the more likely you are to hear about these types of assumptions. Getting them all incorporated into the dispatch system can be quite a challenge. We're sure that there are dozens of other potential examples in fabs, because they are such complex environments.

What it seems to come down to is the split between execution-type activities (things that should be repeatable and thus can be documented in a checklist), and creation-type activities (where you are building something from scratch and there is no checklist). It seems likely that over time we'll have a checklist of things to do and not-to-do in our software code, and we can review against this whenever we develop something new. But it doesn't seem like that really cuts to the heart of the problem, which is the hidden assumption that is so reasonable that you wouldn't even think to prepare for it to be broken (e.g. the assumption that lot transactions with zero wafers are always safe to ignore after a lot is closed).

All we know is that we do spend a lot of time chasing down and fixing these problems, and if at all possible, we'd like to avoid making the same basic mistake in an infinite variety of different ways! We thought that perhaps this same type of thought has occurred to people in fabs, and that you might have feedback on this issue.

Conclusion

Whenever you implement a series of steps, whether this is in software code, a spreadsheet, or an operational process in a fab, you make assumptions along the way. Often these assumptions seem so obvious that you don't even document them, let alone plan for them to be broken. But of course sometimes they do break. When that happens, the root cause is often difficult to identify (even if it is obvious in retrospect). We are hyper-careful about documenting processes and assumptions, but this still happens to us, more often than we would like. Given the complexity of the fab environment, we're sure that this happens in fabs, too. So we thought that we would open up a dialog with our newsletter subscribers on this issue of broken underlying assumptions. We welcome your feedback.

Closing Questions for FabTime Subscribers

Can you think of other examples where a broken assumption could cause (or has caused) an operational problem in a fab? Or in your day-to-day life? Do you think that there's a systematic way to avoid these types of problems? How do you handle this in your fab?

Subscriber List

Total number of subscribers: 2775 from 469 companies and universities.

Top 21 subscribing companies:

- Maxim Integrated Products, Inc (205)
- Intel Corporation (150)
- Chartered Semiconductor Mfg (86)
- Micron Technology, Inc. (83)
- X-FAB Inc. (73)
- Western Digital Corporation (69)
- Texas Instruments (63)
- Freescale Semiconductor (58)
- ON Semiconductor (58)
- Analog Devices (57)
- TECH Semiconductor Singapore (56)
- International Rectifier (55)
- NEC Electronics (53)
- IBM (48)
- STMicroelectronics (46)
- Infineon Technologies (44)
- NXP Semiconductors (42)
- Cypress Semiconductor (38)
- Seagate Technology (36)
- BAE Systems (30)
- National Semiconductor (30)

Top 3 subscribing universities:

- Virginia Tech (11)
- Arizona State University (8)
- Ben Gurion Univ. of the Negev (8)

New companies and universities this month:

- FH-Kärnten University
- Klune Industries
- Mutah University
- SPI Analysis
- United Test and Assembly Center Ltd.

Note: Inclusion in the subscriber profile for this newsletter indicates an interest, on the part of individual subscribers, in cycle time management. It does not imply any endorsement of FabTime or its products by any individual or his or her company.

There is no charge to subscribe and receive the current issue of the newsletter each month. Past issues of the newsletter are currently only available to customers of FabTime's web-based digital dashboard software or cycle time management course.

To subscribe to the newsletter, send email to newsletter@FabTime.com, or use the form at www.FabTime.com/newsletter.htm. To unsubscribe, send email to newsletter@FabTime.com with "Unsubscribe" in the subject. FabTime will not, under any circumstances, give your email address or other contact information to anyone outside of FabTime without your permission.

FabTime® Cycle Time Management Software

"Instead of spending time preparing reports, shift facilitators can get the data they need quickly from FabTime, and then spend their time making real improvements."
Mike Hillis
Cycle Time and Line Yield Improvement Manager
Spansion Fab 25

FabTime Subscription

One low monthly price includes

- Software installation and real-time connect to your MES
- End user and system administrator training
- Unlimited users via your Intranet.
- Software maintenance and regular upgrades (approx. 6 per year, via our no-downtime patch system)
- Add-on dispatching and planning module for a slightly higher monthly fee

Interested?

Contact FabTime for technical details and/or a web-based demonstration.

FabTime Inc.
Phone: +1 (408) 549-9932
Fax: +1 (408) 549-9941
Email: Sales@FabTime.com
Web: www.FabTime.com

Turn fab MES data into information and save time and money

- Are your supervisors swamped with daily reports, but lacking real-time information?
- Is it difficult to link equipment performance to cycle time?
- Does each new cycle time analysis require IT resources?

FabTime can help. FabTime saves your management team time daily by turning fab MES data into information, via a real-time web-based dashboard that includes lot dispatching. FabTime saves your IT staff time by breaking the cycle of custom-developed reports. With FabTime, the end user can filter for exactly what he or she needs, while staying in a comprehensive framework of pre-defined charts. Most importantly, FabTime can help your company to increase revenue by reducing cycle times up to 20%.

"I use FabTime every day, and so do the supervisors who report to me. The data that I need is right on my home page where I need it when I come in every morning."
Jim Wright
Production Manager
Headway Technologies

FabTime Benefits

- Cut cycle times by up to by 20%.
- Focus improvement efforts on the tools that inflate cycle time.
- Improve supervisor productivity – cut reporting time by 50%.
- Improve IT productivity – eliminate need for custom reports.