*Proceedings of the 2011 Winter Simulation Conference*
*S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu, eds.*

# CHALLENGES AND SOLUTION APPROACHES FOR THE ONLINE SIMULATION OF SEMICONDUCTOR WAFER FABS

Daniel Noack
Marcin Mosinski
Oliver Rose

Institute of Applied Computer Science
Dresden University of Technology
01062 Dresden, GERMANY

Peter Lendermann
Boon Ping Gan

D-SIMLAB Technologies Pte Ltd
8 Jurong Town Hall Rd, JTC Summit #30-04
Singapore 609434, SINGAPORE

Wolfgang Scholl

Infineon Technologies Dresden
Königsbrücker Strasse 180
01099 Dresden, GERMANY

## ABSTRACT

To make use of short-term simulation on an operational level, three aspects are essential. First, the simulation model needs to have a high level of detail to represent a small part of the wafer fab with sufficient precision. Second, the simulation model needs to be initialized very well with the current fab state. And third, the simulation results need to be available very fast, almost in real time. Unfortunately these conditions contradict each other. In particular, it takes a large amount of time to initialize a high precision full fab simulation model because of the huge amount of data. In this paper, we present the prototype of a fab driven simulation approach to overcome these time consuming limitations. We will show how it is possible to start a short-term simulation from the current fab state immediately, i.e., without further delay.

## 1 INTRODUCTION

During recent years, huge effort has been spent to realize online simulation. For semiconductor manufacturing, Bagchi et al. (2008) and Scholl et al. (2010) showed that short-term simulation is feasible. They point out that methods of discrete event simulation are useful to solve problems on an operational level. Thus the importance of online simulation increased considerably.

With online simulation we are able to predict future lot arrivals and future work in progress (WIP) for a short time horizon. Therefore it is very useful for daily decision support on the shop floor. Several applications, based on results from short-term simulation, help to increase the fab performance:

- WIP balancing
- Bottleneck detection
- Preventive maintenance planning
- Dynamic dispatch rule selection/ parameterization

978-1-4577-2109-0/11/$26.00 ©2011 IEEE

### 1.1 Problem Description

The characteristics of such an online simulation system compared to an offline simulation include (Scholl et al. 2010):

- A short-term simulation run will cover a small time period (e.g. one week).
- The simulation model needs to have real time capability, i.e. to provide results before they are obsolete.
- The simulation model needs to have a high level of details.
- The model initialization is important because the warm-up period is in the range of interest and not the steady state.

Such properties have a huge influence on each other, especially on the real time capability. As seen in Table 1, a very short simulation period is in agreement (+) with the real time capability. Hence the simulation execution time becomes very short. The high level of detail and the detailed model initialization is in contradiction (-) with the real time requirement. A high level of modeling details increases the simulation execution time. The importance of a detailed model initialization makes it necessary to extract a huge amount of fab data at the beginning of the simulation.

Table 1: The effect of online simulation characteristics on the real time capability

| | | |
|---|---|---|
| Real time capability | + | Short simulation period |
| | - | High level of modeling details |
| | - | Importance of model initialization |

These characteristics of an online simulation violate the requirements of a real time system. The time for data input and model initialization is very long, while the runtime of the simulation run is extremely short. Other colleagues working on online simulation also mention that the time for data input for online simulation takes much longer than the actual simulation run. Furthermore the major trend in semiconductor manufacturing is the increasing wafer fab size with increasing equipment and lot numbers. This results in an increase of the data volume to initialize a simulation model. In Noack et al. (2010) we show that the data extraction, and model creation time is longer than the simulation execution time.

### 1.2 Related Approaches

Besides the fab driven simulation approach, there are other methods to reduce simulation model creation time. A first approach is to differentiate in the frequency of fab data extraction. It is important to extract flow information, like lot moves, every time when a short-term simulation model is supposed to run. It is not useful to extract master data like route or equipment information with the same frequency. Therefore the update period can be much less. A second approach applies for model creation. Multithreading approaches make it possible to read simulation model information in parallel. At the moment most simulation engines read the model information like equipment, route, and lot information sequentially one by one. Both methods are applicable in our case as well. The disadvantage of these methods is that they minimize the data extraction and model initialization time, but they will not speed up the model initialization time up to instantaneous availability.

Related work in this area is done by Aydt et al. (2008) in studies on symbiotic simulation. A simulation of a wet bench work center is updated concurrently. The model update is done by a workcenter snapshot, whereby another simulation model emulates the behavior of a real wafer fab. This approach is very interesting because it is already quite close to our work. The major difference between this and our approach lies in the update mechanism. While Aydt et al. use a full snapshot to update their model we improve the update performance by using events, thereby significantly reducing the amount of update data needed. This results in improved scalability for our event based approach, which even allows updating of a full fab model without violating real time requirements.

### 1.3 Motivation

Our motivation to work on fab driven simulation is that the application of discrete event simulation is highly useful for operational decisions. The obstacle of a long initialization time can be solved by using innovative methods. Therefore we developed the fab driven simulation model. The reduction of model initialization time offers huge benefits for the models application including:

- Forecast results from simulation are not obsolete before they are available
- Immediate response to the user
- Additional application areas for a very short time horizon (e.g., lot arrival prediction for lot scheduling)
- Higher initialization quality through additional short-term data (e.g., position of lots in transport system)

## 2 CONCEPT

In this section we describe the basic principle of fab driven simulation. We explain how fab driven simulation will change the way to start a short-term simulation runs in the future.

### 2.1 Process Flow for Online Simulation

The modeling process of a regular online simulation run, without the fab driven simulation approach, is depicted in Figure 1 on the left side. The user needs to extract the data (1) and initialize the model (2) before starting a simulation run (3). This is necessary to obtain the latest fab state. For short-term simulation we extract the fab data automatically from distributed data bases. Several very time consuming data transformation, error checking, and error correction steps have to be executed. Furthermore the model initialization can take more time than the simulation run itself, depending on the particular scenario. All model files will be loaded from the hard disk into the memory step by step.

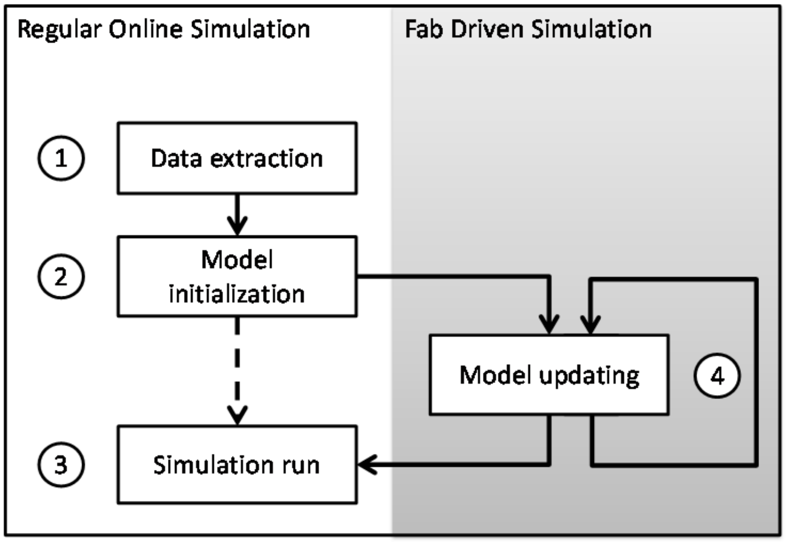


Figure 1: Process flow with fab driven simulation

With the fab driven simulation approach the model needs to be initialized (1 & 2) as well. But this initialization can be done beforehand, for example overnight. After this point in time, the model will be updated continuously (4), see Figure 1 on the right side. When the user wants to start a simulation (3), the updated model is already available. The user is able to run a simulation immediately and gets the results fast. In future, it is planned to duplicate the process instance of the fab driven simulation process before starting a simulation run. Thus the model update process will not terminate when the user starts to simulate. This approach provides a separate continuously ongoing update process at all times even during simulation.

## 2.2 Updating Events

In the previous sections, we talked about updating the simulation model, but we did not mention yet what exactly needs to be updated. For Table 2 we analyzed the most common event types in a real wafer fab. For each event type we highlight the time intervals at which typical events occur. As you can see, the lot movement event type occurs about every 2.4 seconds while a new route is created only every 12.8 days. Thus, these times are an indication which event types have an impact on model update horizon and on model accuracy. For proof of concept in this paper we decided to update the lot moves because they are very relevant for the simulation results and because they occur very often.

Table 2: The mean time between events $\bar{t}$ is computed by $\bar{t} = t/n$ , whereby $n$ is the number of events and $t$ is the observed time period.

| Event type | Mean time between events |
|---|---|
| Lot moves | 2.4 sec |
| Hold | 11.8 min |
| Split | 8.4 min |
| Unscheduled down | 6.6 min |
| Scheduled down | 1.3 min |
| New product | 19.7 hour |
| New route | 12.8 day |

Besides those events mentioned in Table 2, many other elements, which change over time, exist. It depends on the level of details of the simulation model whether those elements need to be updated. Events like dispatch rule changes are hard to quantify. Other changes (e.g., lot release changes) are done in batches and not one by one. Their characteristics have a significant effect on the event update for:

- Process time changes
- Dispatch rule changes
- Dedication changes
- Lot release changes
- Reticle information like reticle position, number of available reticles
- Equipment added or removed
- Sampling rate changes

Using the fab driven simulation approach, many more events could be included into the simulation model. When model initialization and simulation runtime is long, such events and the related results are already obsolete before the simulation results are available. But when simulation results are available in real time, those events considerably improve the quality of online simulation for very short time horizons. Such events include:

- Detailed equipment events like start/finish of the process, pump/vent, heating/cooling
- Chamber events for cluster tools like 1st wafer finished, 2nd wafer finished
- Current lot position in the transport system

## 3 DESIGN DECISIONS

From a conceptual point of view, multiple options exist to implement fab driven simulation. Thus, several decisions are required to find the best solution to update the model. In this section we will discuss multiple approaches and their advantages and disadvantages.

### 3.1 Update Trigger Approaches

This topic is concerned with the start of the update process. It is possible to trigger the update by

- An event that will be available in the fab (e.g., a lot move in), or
- A predefined time interval.

We decided to trigger the update regularly, within a predefined time interval of a few seconds. The advantage is a more predictable update cycle. It is easy to implement because the internal clock triggers the update and not an external event. Furthermore, we avoid the problem that multiple events trigger several updates at the same time.

### 3.2 Update Periods

In this section, we discuss the time between two updates. It is related to the length of the time period and its contained events. Furthermore, the real time capability of the data access is an issue. Two update types can be distinguished:

- Short update period, synchronous time stamp, real time data access.
- Long update period, asynchronous time stamp, no real time capability of the data access.

For the first type, the update period is very small, about a few seconds only. In this case the timestamp is almost synchronous, which means the timestamp of the latest update is similar to the timestamps of its contained events. The data access requirement is to immediately deliver all fab events that occur at this moment in time. The advantage is that the simulation model is always up to date.

For the second type, the update period is very large, about several minutes, or even one or more hours. The timestamp of the last update and the timestamp of its contained events are not similar. It is necessary to distinguish these timestamps to schedule all simulation events at the right time. The advantage of this approach lies in a delayed data access which does not need to satisfy real time requirements. Furthermore the system and network utilization is lower due to reduced update frequency compared to the first approach. The major disadvantage is the need to manage both update time stamps and event time stamps. Another disadvantage is the gap between the fab state and the model state. This gap comes from a higher delay for the model update.

For this study, we decided to use the first approach because the model is up to date without much delay. The key to success is to achieve real time capability for the data access. To achieve this, we use Real Time Dispatcher (Applied Materials 2011), the dispatching system used at the subject wafer fab.

### 3.3 Simulation Time Synchronization Approaches

There are two options to synchronize the simulation time with real time:

- An external timestamp from the latest fab update.
- The internal simulation clock time with the setting of 1:1, whereby one second of simulated time takes one second in reality.

The drawback of the second approach is the unsynchronized simulation and fab time and their deviation after a while. Typical reasons are minimal differences in clock accuracy, mismatch during leap year, and changes due to daylight saving time. Furthermore this approach will give a wrong impression to the user, in case the fab event updates fail. While the simulation clock advances, the user thinks the model is still up to date even when the model state is no longer synchronized and therefore outdated. Our conclusion was to use the first approach. The external timestamp from the last update is used as simulation time. The simulation time will be increased in a loop until the new update timestamp is reached. The underlying assumption is that the timestamp of the next update is larger than the timestamp from the last one. If this condition is not true, the simulation time will not be changed.

### 3.4 Handling of Simulation Events in Fab Driven Mode

For fab driven simulation two different sources for a single event exist, the simulation event and the fab event. To keep the model up to date, the fab event definitely needs to be included into the simulation model. The question is what happens with events from the simulation model. The following approaches exist to handle simulation events:

- Remove simulation events and block resulting state changes.
- Allow simulation events changes and correct state changes later.

The “Remove simulation events and block resulting state changes” strategy removes the simulation event when the related fab event occurs first. If the simulation event is triggered first, the effect of this event will be blocked. The advantage is that simulation will always be synchronous to the wafer fab. The disadvantage is that many events are blocked over time.

The second strategy allows regular simulation event execution and corrects state changes later. When the fab event occurs first, the simulation event will be removed like in the first strategy. In several cases, the simulation event occurs, before the fab event is available. In this case, the simulation event will be executed, in contrast to the first approach which does not allow simulation event execution. If a fab event occurs after the related simulation event, the real fab event corrects the state change caused by the simulation event.

Our decision is to implement the first approach which does not allow state changes caused by simulation events. Therefore when a simulation event occurs, we block the effects of this event. There are two major reasons for this decision. First, the model is always synchronized with the wafer fab. Second, it is not necessary to implement a rollback strategy for simulation events.

## 4 IMPLEMENTATION

The implementation of fab driven simulation consists of several modules, as shown in Figure 2. The following modules and their triggers are described in the following:

- Event export module: This module exports the fab events into the update file. It also exports the current time stamp of the fab. The export module is triggered after a predefined time interval. This is done by the systems of the wafer fab.
- Event import module: This module imports the incoming fab events from the update file into the simulation. It furthermore imports the latest update timestamp of the fab. The file import is triggered by the simulation itself. Due to the fact that the import and export are not synchronized, the frequency of the data import must be higher than the export to avoid overwriting the last event list put into the update file.
- Lot reallocation module: Based on incoming events, this module will include the lot movement events into the simulation model. This module will be executed after the file import. Following main tasks are necessary to update the lot movements:
  - Check if the lot already exists.
  - Remove the lot from its current equipment or queue, if the lot exists.
  - Create a new lot if the lot does not exist yet.
  - Insert the lot into the queue, equipment or sink.
- Time synchronization module: It controls the time of the simulation. The purpose is to avoid deviation between the simulation time and the time of the wafer fab. For this implementation the timestamp given by the import module will be used. The time synchronization will be executed after the lot reallocation, when all other model state changes are done.

  - The simulation clock will be increased iteratively if the simulation time is smaller than the latest update timestamp.
  - The simulation time is stopped when the latest update timestamp is reached.
- Mode switching module: This module is necessary to terminate the fab driven mode and switch to simulation mode. The user decides when to start the simulation. Therefore simulation events will be enabled while fab driven updates will be disabled. Following changes occur:
  - Disable update file import.
  - Disable lot reallocation.
  - Disable time synchronization.
  - Enable lot generation at the source.
  - Enable lot movements from the queue if equipment is available for processing.
  - Enable lot movement when process time at equipment is over.
  - Increase the simulation time, internally.

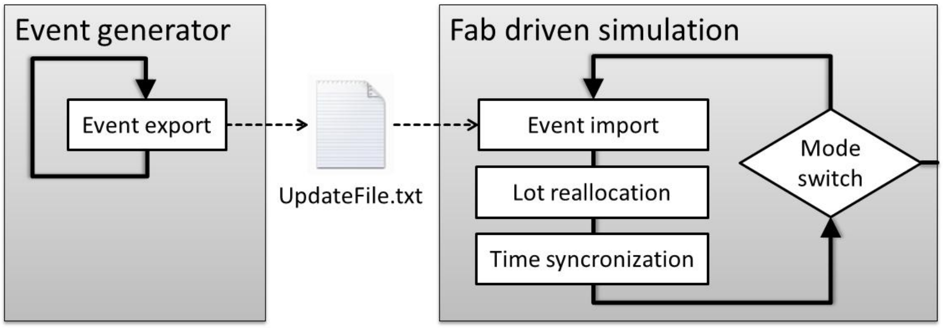
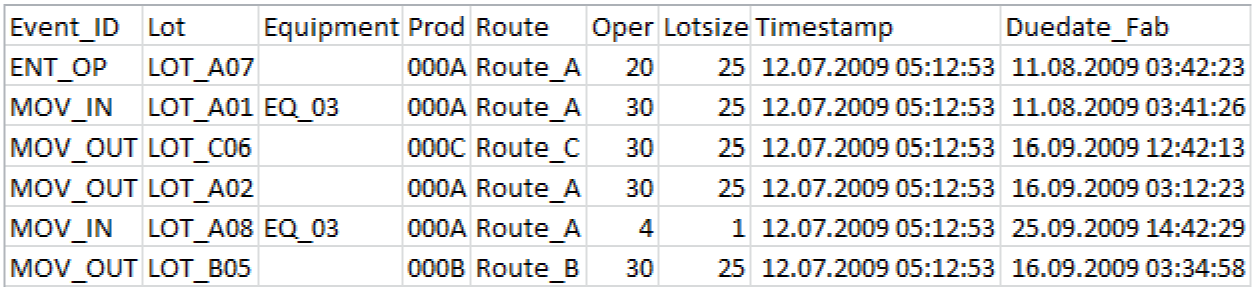


Figure 2: Simplified module execution flow chart for fab driven simulation

The update file is the link between fab operations and simulation run. It is specified in Table 3, including self-generated data examples. The column "Event_ID" defines the type of the lot movement event. The keyword "ENT_OP" indicates that a lot enters the queue of the current operation. The keyword "MOV_IN" indicates that a lot enters a tool. The "MOV_OUT" event tells the system that the lot leaves the tool. The column "Equipment" is required only for lots with a "MOV_IN" event. In the update file, the column "Lot" contains the lot name. It must be unique because a single lot cannot have more than one update event at the same time. Additional columns like "Product," "Route," or "Lotsize" are used to show that it is possible to update several other lot attributes as well.

Table 3: Update file example



| Event_ID | Lot | Equipment | Prod | Route | Oper | Lotsize | Timestamp | Duedate_Fab |
|---|---|---|---|---|---|---|---|---|
| ENT_OP | LOT_A07 | | 000A | Route_A | 20 | 25 | 12.07.2009 05:12:53 | 11.08.2009 03:42:23 |
| MOV_IN | LOT_A01 | EQ_03 | 000A | Route_A | 30 | 25 | 12.07.2009 05:12:53 | 11.08.2009 03:41:26 |
| MOV_OUT | LOT_C06 | | 000C | Route_C | 30 | 25 | 12.07.2009 05:12:53 | 16.09.2009 12:42:13 |
| MOV_OUT | LOT_A02 | | 000A | Route_A | 30 | 25 | 12.07.2009 05:12:53 | 16.09.2009 03:12:23 |
| MOV_IN | LOT_A08 | EQ_03 | 000A | Route_A | 4 | 1 | 12.07.2009 05:12:53 | 25.09.2009 14:42:29 |
| MOV_OUT | LOT_B05 | | 000B | Route_B | 30 | 25 | 12.07.2009 05:12:53 | 16.09.2009 03:34:58 |

To show how the procedure works, let us assume the current model state is defined as depicted in Figure 3. Three lots are in the queue. Two lots are in the first equipment. One lot is located in the second equipment.

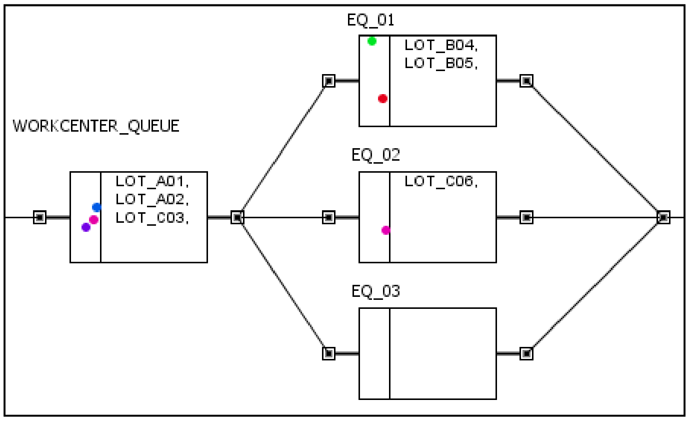


Figure 3: Model state before an update

Now a single update as depicted in Table 3 will be applied to the model in Figure 3. The lot “Lot_A07” from Table 3 will enter the queue. The lot “Lot_A01” move from the queue to equipment “EQ01.” The lot “Lot_A06” leaves the tool “EQ_01” and will be removed by the sink, and so on. When all events have been ended, the next model state as depicted in Figure 4 will be reached. This single event update transfers the old model state to the new model state. By doing such a single update in an iterative way, the lot moves in the simulation model are updated over a long time period.

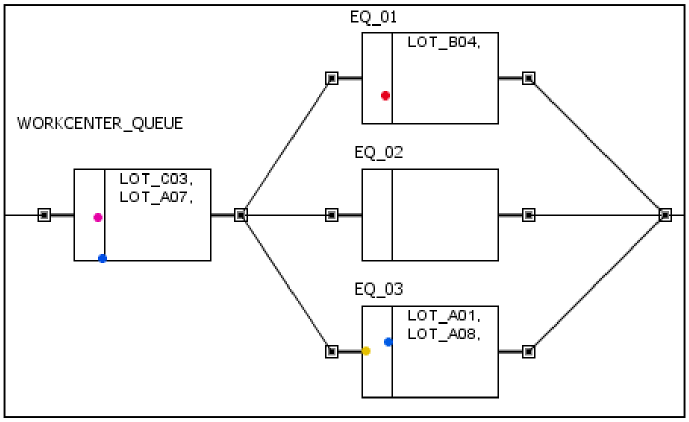


Figure 4: Model state after an update

We use the Real Time Dispatcher (RTD) version 7.2.4 to export the update file. The simulation model itself is created with AnyLogic 6.5.0 (XJ Technologies Company 2011). The relevant model extension modules are implemented in Java.

## 5 RESULTS

The objective of the results section is to show the feasibility of fab driven simulation. In the first part, relevant test cases will be presented. The second part contains results from a fab driven simulation run in a real wafer fab.

### 5.1 Basic Test Cases

Beside several unit tests where each module has been tested separately, numerous black box tests have been performed. The purpose of these black box tests is to check if the whole software solution is working correctly. All these test cases have been evaluated manually. The data source for the fab driven simulation is an artificial update file. Table 4 contains a small subset of the test cases that have been executed.

Table 4: Test cases for update function

| **Test case description** | **Expected results** | **Test passed** |
|---|---|---|
| A new lot enters a queue | The lot appears in the queue. | Yes |
| A queue lot enters a queue | The lot is in the queue. | Yes |
| An equipment lot enters a queue | The lot appears in the queue and disappears in the equipment. | Yes |
| A new lot enters an equipment | The lot appears in the equipment. | Yes |
| A queue lot enters an equipment | The lot appears in the equipment and disappears in the queue. | Yes |
| An equipment lot enters a different equipment | The lot appears in the target equipment and disappears in the previous equipment. | Yes |
| An equipment lot enters the same equipment | The lot is in the equipment. | Yes |
| A new lot moves out | This will be interpreted as data error because a lot which is not in the operation before cannot move out. It will be ignored. | Yes |
| A queue lot moves out | The lot appears in the sink and disappears in the queue. | Yes |
| An equipment lot moves out | The lot appears in the sink and disappears in the equipment. | Yes |
| Update the simulation model time when real time is later than simulation time | The simulation model time becomes the real time. | Yes |
| Update the simulation model time when real time is earlier than simulation time | The simulation model time do not change. | Yes |

We created many test cases. We consider the origin of the lot - whether it is an existing lot from the queue, or from equipment, or does this lot not exist in the model yet. We checked whether the update behavior is correct, even when the previous and the new location of the lot is the same.  Furthermore, several test cases have been executed for single lots, and for multiple lots that move simultaneously.

### 5.2 Test in a Real Wafer Fab

Other test cases were performed in a real wafer fab. Therefore one workcenter was selected to demonstrate that fab driven simulation is working under real world conditions. The criteria to select this workcenter were:

- The number of workcenter moves is high. So it is possible to show lot movements even for a live demonstration.
- The work center is isolated. This means the lots running on this work center cannot be processed by an alternative work center.
- The work center contains only a few tools to show all of them on one screen.

We have selected an isolated work center with 5 tools. The throughput of this small work center is huge with about 120 lots per day. This is one lot movement every 12 minutes whereby one move consists of “enter operation”, “move in”, and “move out”. The result of this test is depicted in Figure 5. The number of WIP lots increases and decreases over time. The runtime is about 90 minutes or 5400 seconds, whereby the time unit is in seconds since simulation start.

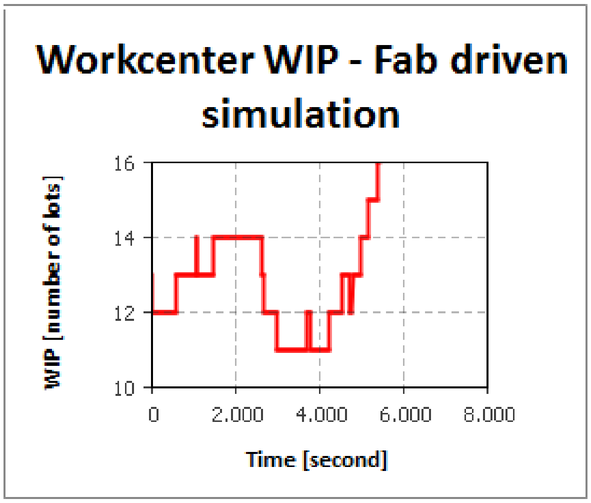


Figure 5: Workcenter WIP of fab driven simulation

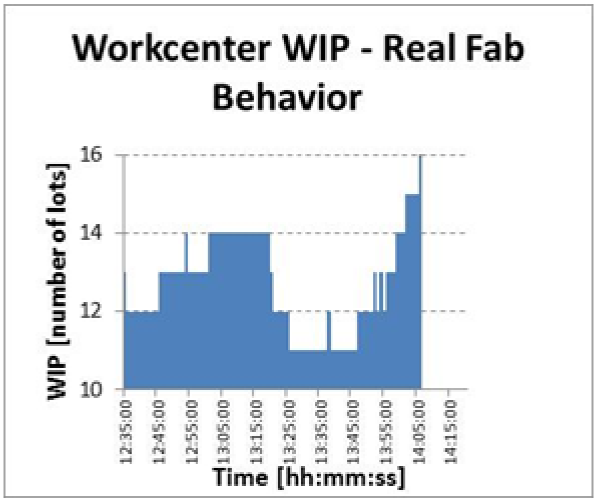


Figure 6: Workcenter WIP in reality

To show that the WIP of the fab driven simulation in Figure 5 is correct, the real WIP is depicted in Figure 6. This curve shows the number of lots for the same work center at the same time period. As you can see, the WIP over time is exactly the same. The simulation model update works under real world conditions within a long time period. It demonstrates that fab driven simulation is feasible.

## 6 CONCLUSIONS

It has been shown that fab driven simulation is feasible. A prototype has been implemented to update the lot moves for one workcenter. This prototype demonstrates that it is possible to continuously update a simulation model during runtime. By having an updated model, the user is able to run a short-term simulation forecast immediately. No time consuming data input and model initialization is required.

The "lessons learned" from this project are that the complexity of fab driven simulation is huge. The interaction of fab events and simulation elements increase the complexity of fab driven simulation significantly. The software engineering part is a highly important aspect to realize fab driven simulation for short-term simulation. Furthermore there is a limitation from software engineering perspective. The fab driven simulation feature is highly connected to the simulation tool. So it is not easy to transfer the capability of fab driven simulation to another simulation tool from a different vendor.

The next steps involve extending the fab driven simulation approach. First, it is required to increase the scale of the model to the whole wafer fab. Second, it is necessary to update more simulation event types besides lot moves. The third element is to clone the fab driven simulation process during runtime. At the moment exactly one simulation run can be done with the fab driven model. Our objective is to let the fab driven simulation model continuously run and create a copy of this running process. The copy will be used to create a short-term forecast, while the original process keeps on synchronizing with the real fab.

## REFERENCES

Applied Materials. 2011. APF RTD. Accessed September 23. http://appliedmaterials.com/services-software/library/apf-rtd-apf-reporter.

Aydt, H., S. J. Turner, W. Cai, M. Y. H. Low, P. Lendermann, and B. P. Gan. 2008. "Symbiotic Simulation Control in Semiconductor Manufacturing". In *Proceedings of the International Conference on Computational Science*, edited by G. van Albada, J. Dongarra, and P. Sloot, 26–35. Berlin / Heidelberg: Springer.

Bagchi, S., C.-H. Chen-Ritzo, S. T. Shikalgar, and M. Toner. 2008. "A Full-Factory Simulator as a Daily Decision-Support Tool for 300mm Wafer Fabrication Productivity." In *Proceedings of the 2008 Winter Simulation Conference*, edited by S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, and J. W. Fowler, 2021–2029. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Noack, D., R. Kohn, M. Mosinski, Z. Zhou, O. Rose, W. Scholl, P. Lendermann, and B. P. Gan. 2010. "Data Modelling for Online Simulation - Requirements and Architecture." In *Proceedings of the 2010 FAIM Conference*. edited by B. Pulikkal, M. Ramiah, and P. Ronald, 1037-1044.

Scholl, W., B. P. Gan, D. Noack, P. Preuss, M. L. Peh, P. Lendermann, and O. Rose. 2010. "Towards Realization of a High-Fidelity Simulation Model for Short-term Horizon Forecasting In Wafer Fabrication Facilities." In *Proceedings of the 2010 Winter Simulation Conference*, edited by B. Johansson, S. Jain, J. Montoya-Torres, J. Hugan, and E. Yücesan, 2563–2574. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

XJ Technologies Company. 2011. "Anylogic." Accessed September 23. http://www.xjtek.com/anylogic.

## AUTHOR BIOGRAPHIES

**DANIEL NOACK** is a Research Engineer at D-SIMLAB Technologies. His focus is on simulation and simulation-based optimization. He is also a PhD student at Dresden University of Technology. He is a member of the scientific staff of Professor Oliver Rose at the Chair of Modeling and Simulation. He received his M.S. degree in Computer Science from Dresden University of Technology. His email address is daniel@d-simlab.com.

**MARCIN MOSINSKI** is a PhD student at Dresden University of Technology. He is a member of the scientific staff of Professor Oliver Rose at the Chair of Modeling and Simulation. He received his M.S. degree in computer science from Dresden University of Technology. His research interests include forecasting of complex problems in manufacturing facilities and the statistical data analysis. His email address is marcin.mosinski@tu-dresden.de.

**OLIVER ROSE** holds the Chair for Modeling and Simulation at the Institute of Applied Computer Science of the Dresden University of Technology, Germany. He received an M.S. degree in Applied Mathematics and a Ph.D. degree in Computer Science from Würzburg University, Germany. His research focuses on the operational modeling, analysis and material flow control of complex manufacturing facilities, in particular, semiconductor factories. He is a member of IEEE, INFORMS Simulation Society, ASIM, and GI. His web address is www.simulation-dresden.com and his email address is oliver.rose@tu-dresden.de.

**PETER LENDERMANN** is the Co-Founder and CEO of D-SIMLAB Technologies, a Singapore-based company providing simulation-based decision support solutions and services to Aerospace, Semiconductor Manufacturing and other asset-intensive industries. Prior to this he worked at the Singapore Institute of Manufacturing Technology where he led the simulation-related research activities until spinning them off into D-SIMLAB Technologies. He has been engaged in the simulation community since the early 1990's. Peter holds a PhD in Applied High-Energy Physics from Humboldt- University in Berlin (Germany) and an MBA in International Economics and Management from SDA Bocconi in Milan (Italy). He is also an Adjunct Associate Professor at the Department of Industrial and Systems Engineering at the National University of Singapore. His email address is peter@d-simlab.com.

**BOON PING GAN** is the CTO of D-SIMLAB Technologies (Singapore). He has been involved in simulation technology application and development since 1995, with primary focus on developing parallel and distributed simulation technology for complex systems such as semiconductor manufacturing and port operations. He was also responsible for several operations improvement projects with wafer fab-fabrication clients which concluded with multi-million dollar savings. He is now responsible for managing the technology and product development at D-SIMLAB as well as execution of projects in the semiconductor and engineering domains. He holds a Master of Applied Science degree, specializing in Computer Engineering. His email address is boonping@d-simlab.com.

**WOLFGANG SCHOLL** works as a Senior Staff Expert for modeling and simulation for Infineon Technologies in Dresden (Germany). He studied Physics at the Technical University of Chemnitz (Germany) and graduated in solid-state physics in 1984. From 1984 to 1995 he worked as a process engineer for ZMD in Dresden. In 1996 he joined Infineon Technologies and initially worked on the field of capacity planning. Since 2003 he is responsible for fab simulation. His email address is wolfgang.scholl@infineon.com.