2007 International IEEE Symposium on Precision Clock Synchronization (ISPCS)
for Measurement, Control and Communication
Vienna, Austria, October 1-3, 2007.

Precise Time Synchronization in Semiconductor Manufacturing

Vinod Anandarajah, Naveen Kalappa, Rahul Sangole, Sulaiman Hussaini, Ya-Shian Li[+], Julien Baboud[+], James Moyne
Engineering Research Center for Reconfigurable Manufacturing Systems,
University of Michigan, Ann Arbor, MI 48109-2125.
vinoda@umich.edu, nkalappa@umich.edu, rsangole@umich.edu, hussaini@umich.edu, moyne@umich.edu
[+]Semiconductor Electronics Division, National Institute of Standards and Technology,
Gaithersburg, MD 20899-8120.
ya-shian.li@nist.gov, julien.baboud@nist.gov

Abstract

In today's semiconductor fabrication facilities ("fabs"), coordination of time-based information throughout the factory and enterprise has become necessary to support fab-wide diagnostics, control, and information management. This has driven the need to have time synchronization at all levels of the enterprise. Time synchronization protocols such as Network Time Protocol (NTP) and Precision Time Protocol (PTP) have been defined for performing synchronization over distributed systems. Lack of time synchronization among the various subsystems is seen as a factor of poor data quality in Equipment Data Acquisition (EDA) and Advanced Process Control (APC) analysis. The focus of our study is to investigate the extent and precision of time synchronization that can be practically applied with the available protocols at various levels of the semiconductor factory environment to meet next generation manufacturing requirements. To this end, we describe the objectives, details, and implementation of the simulator that aims to model a semiconductor factory network. This will provide a practical perspective to study the accuracy achievable and potential network factors contributing to accuracy degradation of factory-wide time synchronization.

1. Introduction

The backbone of all manufacturing processes is automation. It has resulted in a significant productivity increase and also enhanced the product quality to a great extent. However, as the product becomes more complex it has driven the need for more refinement in the automation processes. In semiconductor manufacturing, growing complexity and shrinking device sizes will require more precise manufacturing processes where data quality will play a critical role for process control and diagnostics.

The manufacturing industry has moved to distributed architectures allowing greater reconfiguration capabilities. The network plays an important role in distributed architectures. Considering the ubiquity and cost benefit obtained from Ethernet, it has become the network of choice for the manufacturing floor [1]. To facilitate a uniform data exchange mechanism on the network, eXtensible Markup Language (XML), the language used in many web services communications, is being used as the standardized data format. XML is the language specified in many semiconductor manufacturing standards including the Equipment Data Acquisition (EDA) communication standard [6,7,8,9]. By adopting Ethernet and the web services paradigm, the semiconductor industry can leverage mainstream information technology to rapidly ramp-up new data acquisition interfaces and other distributed software automation tools. However, XML also has the potential to contribute to quality of service degradation due to the amount of processing involved. One key issue in data acquisition quality of service is the ability to reconstruct the sequence of events to determine cause-effect. As data are being exchanged, collected and analyzed at higher speeds for better control of the processes, it has become important to have a precise time synchronization mechanism for merging the data obtained from different sources.

In the semiconductor manufacturing industry, EDA is seen as an important data source for fault detection and classification (FDC) which would be able to detect faults and determine the causes in real-time to improve product yield. The lack of accurate time synchronization and time-stamping from the various data sources hampers data quality obtained through this interface. The end-to-end performance studies identify the node as the primary cause for delay and delay variability in Ethernet-based network communication systems [2,3]. Application of time synchronization protocols such as Precision Time Protocol (PTP – based on IEEE 1588) in generic network communication to address the delay issues has also been studied [4].

EDA data are obtained from different semiconductor manufacturing equipment systems and sub-systems. To accurately correlate the data obtained with the occurrence of events from varied sources it is essential to have time synchronization among data sources [5].

Performance and benefits of EDA implemented using SOAP/XML messages over a HTTP connection have been studied [10]. The criteria for using the SOAP/XML platform are based on its interoperability in applications due to mainstream adoption and broad commercial support and availability. Further investigation into the design and implementation of XML-based messaging is needed with particular focus on semiconductor manufacturing equipment due to the rapid growth of technology taking place in the field. With the development of the simulator, data acquisition performance and time synchronization aspects could be studied more closely before deployment of specific technologies. A study of the EDA factory data throughput utilizing the Microsoft.NET platform [11] indicates current technology is sufficient for supporting industry requirements. With data throughput capability comes the need for data quality. It is imperative to characterize the performance of XML-based data acquisition in a factory network and to assess its impact on time synchronization and time-stamping accuracy.

This paper presents the results to-date of a project focused on exploring the impact, capabilities, and limitations of utilizing time synchronization in aspects of semiconductor manufacturing. Specifically, a simulation approach is being utilized to study the impact of time synchronization on EDA networked systems throughout the semiconductor "fab." The developed EDA simulator helps determine the extent and precision of time synchronization required for meeting relevant performance criteria. The simulator is capable of generating data traffic similar to the patterns seen in the semiconductor manufacturing environment. In addition, there is a capability for introducing random traffic and noise patterns. EDA data obtained from different sources will eventually be time-stamped at different levels of precision (hardware and software time-stamping) and time synchronization protocols among the different sources will be applied. Under different conditions of network and equipment loading, correlation between data obtained from varied sources and their respective occurrence of events is carried out. This will help to determine the extent of time synchronization accuracy that is required to ensure proper data quality under different conditions. Based on simulation results, effective recommendations will be provided for implementing fab-wide network time synchronization, data acquisition, and data time-stamping. A cost-benefit analysis will be performed for the application of NTP (Network Time Protocol) and PTP (Precision Time Protocol – based on IEEE 1588) at various levels of the semiconductor factory floor.

Section 2 gives a brief overview of the EDA standard. In Section 3, the design details, implementation and present project status of the semiconductor factory network simulator is discussed. The conclusions and future work are mentioned in Section 4.

2. Components of EDA

EDA specifies a collection of SEMI standards for the semiconductor industry to facilitate communication between data collection applications (EDA client) and factory equipment (EDA server).
The SEMI standards used to describe EDA are

- E120 – Common Equipment Model
- E125 – Equipment Self Description
- E132 – Client Authentication and Authorization
- E134 – Specification for Data Collection Management

The SEMI E134 [9] standard helps to create a more manageable and flexible high speed data collection environment. The process and operational data are organized into logical, named units through Data Collection Plans (DCPs) that can be individually activated or deactivated.

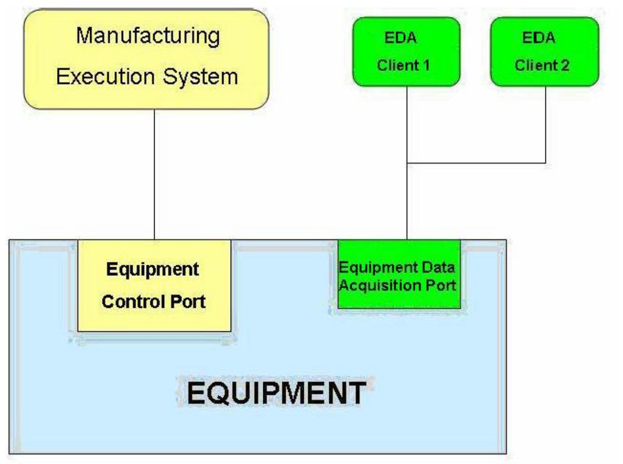

Manufacturing Execution System
EDA Client 1
EDA Client 2
Equipment Control Port
Equipment Data Acquisition Port
EQUIPMENT

Figure 1. EDA port on the equipment.

As shown in Figure 1, EDA provides multiple client access to data gathering capabilities. Through EDA, data concerning process information, equipment utilization details, sensor feedback and actuator states are gathered which eventually facilitates process and product improvement, equipment utilization and equipment maintenance. All equipment control is achieved through a different port as indicated in Figure 1. The EDA interface uses SOAP/XML messages over a HTTP or

HTTPS connection for communication between the client and equipment at high frequency. E134 defines the behavior associated with the execution of data collection plans in the form of finite state machines.

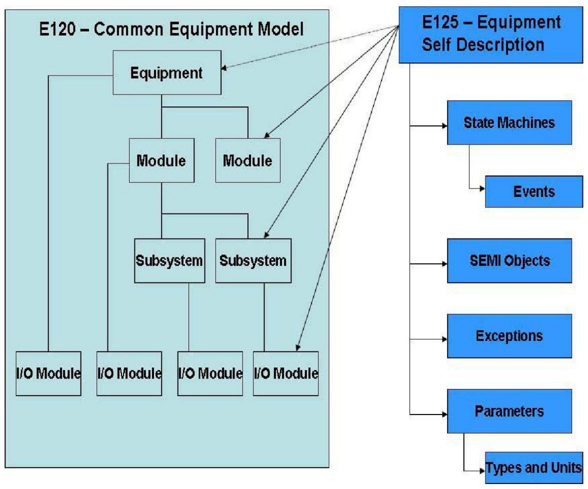


Figure 2. Mapping of E120 and E125.

Modeling of data from equipment requires knowledge of data from the different modules and subsystems. E120 [6] provides a general object model that represents an external view of equipment. It represents a logical hierarchy as represented in Figure 2, which includes the I/O device, subsystem, module, and equipment. E125 [7] allows clients to request descriptions of parameters (data, units, and types), events, exceptions, state machines, and physical configuration. All the available information is mapped into E120 Common Equipment Model hierarchy. With E125, the client is able to decide what data to monitor with regard to a particular context.

The E132 [8] specification defines security related features for EDA messaging which includes client authentication and authorization.

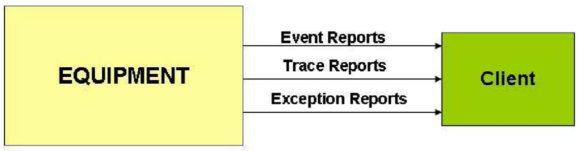


Figure 3. EDA reporting.

E134 is used to define and activate Data Collection Plans (DCPs). DCPs are defined for event, trace, and exception requests as shown in Figure 3. The DCPs are configured by the user (EDA client) based on the application need and downloaded to the equipment. On activation of a DCP from the client, the equipment sends data in the form of data collection reports (DCRs). Context is communicated via events and exceptions. Event reports give information on state changes in the equipment which provide knowledge of the working condition of the equipment. Exception reports monitor error conditions on the equipment. Trace contains the real-time series data for monitoring the equipment. The trace data may have to be buffered on the equipment before being sent to the client at periodic intervals. Figure 4 shows the format of a DCR which includes the event, exception and trace information. It also includes the data collection plan ID, and the start and end time of the report.

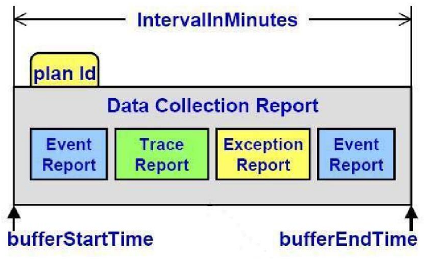


Figure 4. Data collection report (DCR) format.

Figure 5 shows a data exchange scenario between the EDA port on the equipment and a client. It is based on the standards discussed above, and the sequence has to be maintained for equipment servers and clients implementing the EDA data collection mechanism.

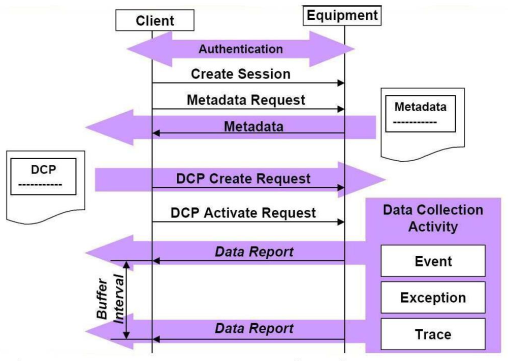


Figure 5. EDA communication sequence.

3. Factory Network Simulator

The Engineering Research Center at the University of Michigan is working with NIST and has developed a first version of a network simulator for the semiconductor factory. This simulator will be used to determine the best practices for networked time synchronization in semiconductor manufacturing. The simulator will determine the extent and precision of time synchronization required for meeting data collection performance criteria. The simulator includes a network noise generator that can be used to generate traffic and noise patterns commonly seen on the semiconductor factory floor. Studies show that the predominant bottleneck is encoding and decoding of XML messages in XML based messaging systems [10,11]. Time synchronization protocols such as Network Time Protocol (NTP) and Precision Time Protocol (PTP - based on IEEE 1588) will be applied at different levels of the factory simulator and the benefits obtained will be studied. Additionally, the impact on data time-stamping accuracy based on time-stamping at the hardware, kernel, and application levels will be examined. Furthermore, the impact of using different operating systems will be examined. The simulator can also be expanded simulate advanced process control capabilities in predictive analysis. A comparison of alternative methods for time synchronization, data acquisition and data analysis will provide recommendations for deployment of EDA, IEEE 1588 and NTP in a semiconductor factory environment.

3.1. Design of simulator

The various elements of the simulator are illustrated in Figure 6; these include the simulation controller, time server, EDA servers and clients, traffic and noise generators, and I/O devices.

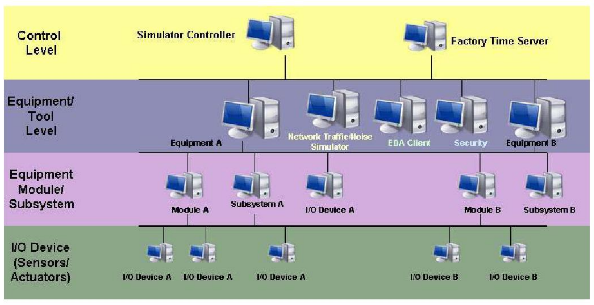


Figure 6. Simulation architecture.

The simulation controller is the centralized point of control; this single point of control allows the data collection simulation to run without additional network impact. The simulation controller GUI is depicted in Figure 7.

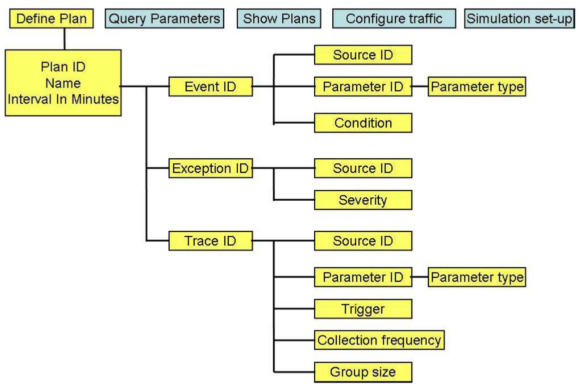


Figure 7. Simulation controller GUI.

From here, the user can configure the entire network. The user can query the EDA servers for possible parameters (based on E125 and E120) that could be obtained from the equipment. Using these parameters, the user can configure the EDA client by defining the DCP. The DCP is sent to the EDA server which then collects the data and compiles the DCRs. These DCRs are then sent back to the EDA client with the requested data at the specified time intervals. The simulator controller also allows the user to configure noise and traffic patterns. The user can further specify the noise size and frequency.

3.2. Implementation of simulator and current status

The implementation of the simulator is nearly complete. The noise generator and EDA client can be configured from the simulation controller GUI. The EDA server package is capable of receiving and handling DCPs and sending back DCRs to the EDA client. The EDA server consists of three main modules: the `DcpHandler` class, the `EventGenerator` class, and the `DcrHandler` class. The `DcpHandler` class accesses elements and attributes of the DCP that was downloaded to the EDA server from the EDA client. The `EventGenerator` class generates random events at specified time intervals which essentially is simulating the equipment data sources such as a sensor. Eventually, the simulator will be able to take data from physical hardware as well as from this simulated `EventGenerator`. The `DcrHandler` class forms the DCRs in compliance with the E134 schema. The interaction of these modules within the EDA server with the EDA client is seen below in Figure 8. The only remaining component that needs to be developed is the capability of the EDA server to include equipment models of different equipments and indicating available parameters prior to the configuration of the DCPs.

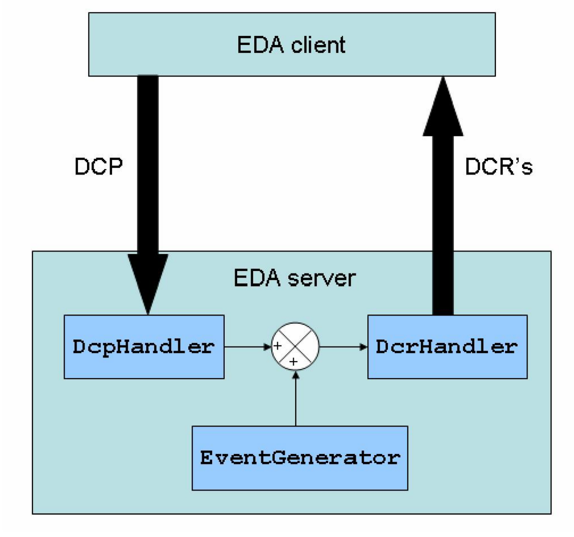


Figure 8. EDA server/client interaction.

3.3. Noise generator validation

The noise generator has been validated. Experiments were conducted using a simplified data exchange of a fixed data size and time interval in between messages. Initially, a 10 Mbps Ethernet hub was used because a low amount of noise was required to introduce time delays into the system. Having validated the noise generator, future tests will be performed over a 100 Mbps Ethernet hub as well as a switch. The experimental setup is shown below in Figure 9 in which the data exchange takes place between the example server and example client and the noise is directed from the noise server to the noise client. The network traffic of TCP packets is monitored using Ethereal, a network protocol analyzer.

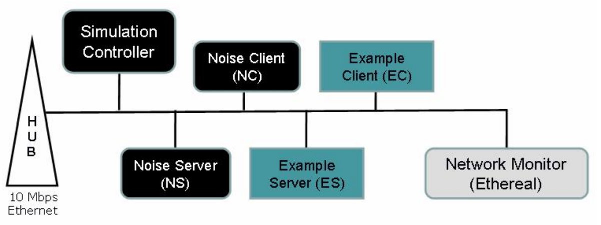


Figure 9. Noise generator validation setup.

The time delays resulting from the noise generator at a low noise message size of 10 KB and a high noise message size of 1 MB is shown below in Figure 10. The tests were designed such that noise messages were sent in a continuous loop, the limiting factor being the time taken by the application to form the messages and transmit them. There is provision for the user to configure the time interval as well as the noise size from the simulation controller. The tests validated that the noise generator successfully introduces delays and delay variability into the network. This will be used to generate noise typically found on semiconductor factory floors.

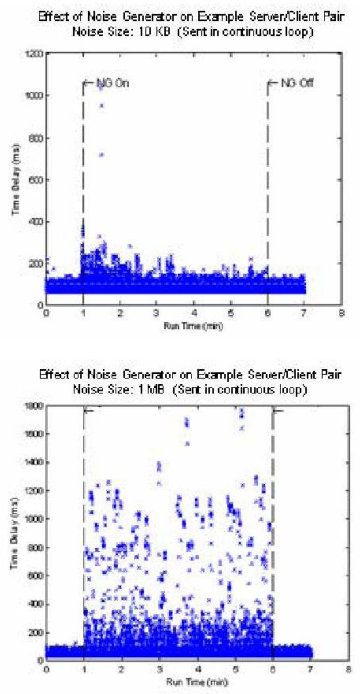


Figure 10. Noise generator successfully introduces time delays and time delay variability into the network

3.4. Time-stamping performance study

Time-stamping performance is being evaluated and improved with Java running on Windows, the most common operating system used on the factory floor. Currently, the default time-stamping resolution of Java with the Windows operating system limits the data time-stamping accuracy. The EventGenerator module, which uses built-in Java schedulers, is accurate on an average but cannot achieve the 1 ms precision requirement for specified times less than or equal to 100 ms. For example, at a specified event generation frequency of every 5 ms, the Java scheduler generated three events every 15 ms instead of one event every 5 ms. Hence, on average the scheduler is very accurate; however its resolution is poor. Initial results can be seen below in Figure 11.

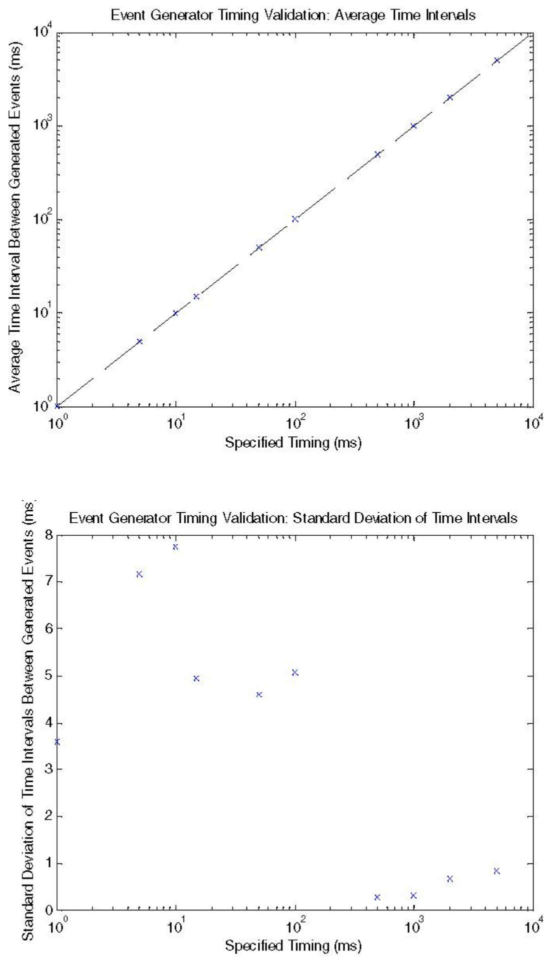


Figure 11. Java time schedulers are accurate on average but cannot achieve 1 ms precision requirement for specified timing under 100 ms

Furthermore, Java time-stamping has a resolution of approximately 10-15 ms using the Windows operating system. This was improved by using the Java Native Interface (JNI) to call upon C++ code to obtain time stamps; however, supplementing C++ compromises Java portability. Figure 12 depicts a comparison of time stamps from Java, C++, and integrated Java and C++ (JNI). The poor resolution of Java time stamps with Windows is evident. The range in between consecutive time stamps is (0,16) ms. On the contrary, C++ time stamps had a resolution of 0.059 ± 0.047 ms with a range of (0.050,6.979) ms. Likewise, the JNI time stamps also had a high resolution of 0.041 ± 0.058 ms with a range of (0.029,6.240) ms. Also, the higher time delays for JNI and C++ indicate jitter due to the Windows operating system. For the JNI and Java, jitter could also be due to the automated garbage collection, which may halt program execution at arbitrary times to attempt to reclaim memory that will never be accessed again by the application. The unpredictability in the pauses results in jitter in the time stamps [12].

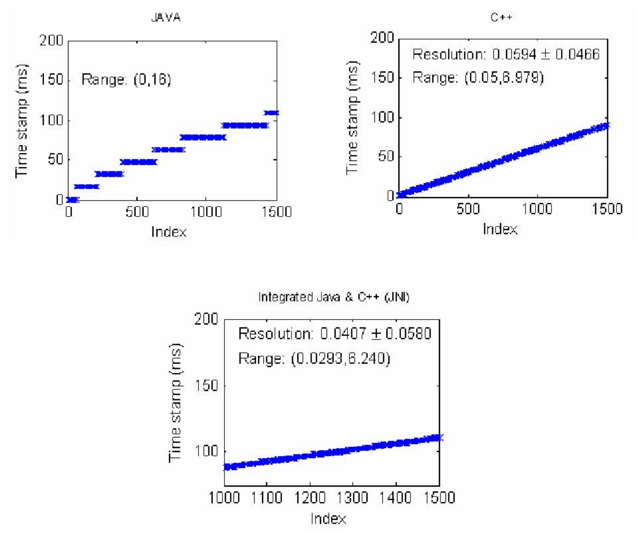
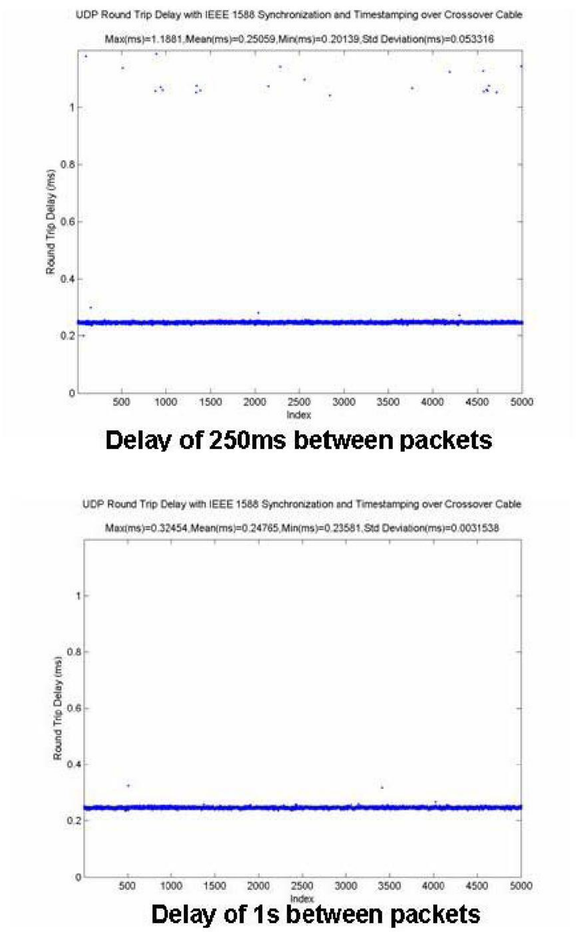


Figure 12. Poor resolution of normal Java time-stamping when compared to C++ and integrated Java and C++ using the JNI.

3.5. Initial testing using IEEE 1588 PCI cards

Initial data exchange tests were conducted using National Instruments 1588 PCI cards [4]. The cards did not allow time-stamping of data packets at the hardware level, only the PTP messages were time-stamped at the hardware level. Though there was nanosecond-level synchronization between the nodes, the time-stamping of data packets was done through the LabVIEW graphical user interface (GUI) which added additional delay and delay variability. Results of the tests are shown in Figure 13. It can be seen that as the interval between transmitted packets is increased, the standard deviation of the delay decreases. The decrease of the standard deviation indicates the impact of the application processing on the variability of network transmission times. To eliminate time-stamping variability there should be a capability of time-stamping at the hardware-level.



Delay between packets	Max (ms)	Mean (ms)	Min (ms)	Std Dev (ms)
250ms	1.188	0.251	0.201	0.053
1s	0.317	0.243	0.234	0.003

Figure 13. Initial tests with 1588 PCI cards indicate that application processing time hampers network times.

3.6. Effects of XML-based messaging systems

Studies indicate that the encoding and decoding of XML messages is the predominant limitation on data collection performance in XML-based messaging systems [10,11]. Initial tests conducted show that for an 800 KB sample DCP XML file, the entire EDA loop currently takes 185 ± 20 ms which is significant. The entire EDA loop consists of downloading the DCP from the EDA client to the EDA server, the generation of the DCR, and its transmission from the EDA server to the EDA client. Once the simulator is complete, a complete study on the effects of XML-based messaging systems on data collection performance will be conducted.

4. Conclusions and Future Work

The initial phase of the EDA simulator developed has been able to create scenarios typical of the Equipment Data Acquisition messages seen on the factory floor. XML message processing is hampering network performance times as can be seen from the initial results. Also from the noise generator analysis, it has been shown that the network communication delay and delay variability increases with increase in noise. The simulator has allowed the study of scenarios with XML message and different noise patterns that are common on the semiconductor manufacturing floor thus giving a deeper insight into the network delay problems. Additionally, noise in the network would degrade the time synchronization performance of software-based protocols such as NTP and software-only PTP.

Accurate time-stamping of data in a Windows environment currently requires C++, which can be integrated into Java through JNI. The use of C++ significantly improved the scheduling resolution. Additionally, initial experiments have shown the availability of precision time synchronization, such as PTP is not sufficient for ensuring data time-stamping performance. Time-stamping is limited at the application layer by the language used to develop the application and the operating system.

While the web services paradigm provides more rapid implementation and integration of distributed systems, the use of XML with the amount of data being transported can increase network jitter, which degrades time synchronization and time-stamping accuracy. The next phase of the simulation will research means to mitigate the jitter through data collection algorithms such as batching and buffering.

Future efforts will also include scaling the simulator to run multiple instances of equipment server objects simultaneously. Additionally, the capability of the EDA server of interpreting E120/E125 from different equipments and indicating available parameters prior to the configuration of the DCPs needs to be added to the simulator. Furthermore, SOAP transport will be implemented to provide a more realistic rendering of the EDA process and its impact on network performance, and subsequently time synchronization performance.

An expanded speed and jitter study of application programming and operating system environments will be conducted in order to understand their applicability with various time synchronization capabilities. Time-stamping will be compared at the hardware, kernel, and application levels. Likewise, time-stamping will be compared on different operating systems. Furthermore, real-time Java capabilities will be implemented and evaluated [12,13]. Additionally, NTP and PTP (based on IEEE 1588) will be applied to various levels of the factory simulator and a cost-benefit analysis will be performed. Recommendations for deployment of PTP and NTP in a semiconductor factory environment will be provided.

References

[1] J. R. Moyne and D. M. Tilbury. The Emergence of Industrial Control Networks for Manufacturing Control, Diagnostics, and Safety Data. In *Proc. IEEE*, 95(1): p29 – 47, 2007
[2] J. T. Parrot, J. R. Moyne and D. M. Tilbury. Experimental Determination of Network Quality of Service in Ethernet: UDP, OPC, and VPN. In *Proc of the ACC*, 2006
[3] J.D. Decotignie. Ethernet-Based Real-Time and Industrial Communications. In *Proc. IEEE*, 93(6): p.1102 - 1118, 2005
[4] N. Kalappa, J. Moyne, J. Parrott and Y. Li. Practical Aspects Impacting Time Synchronization Data Quality in Semiconductor Manufacturing. In *Proceedings of the IEEE 1588 Conference*, October 2006
[5] J. Moyne, J. Parrott, N. Kalappa and Y. Li. Practical Aspects Impacting Time Synchronization Data Quality in Semiconductor Manufacturing. In *Proceedings of the AEC/APC Symposium*, October 2006
[6] SEMI E120 Common Equipment Model Specification
[7] SEMI E125 Equipment Self Description Specification
[8] SEMI E132 Equipment Client Authentication and Authorization Specification
[9] SEMI E134 Data Collection Specification
[10] S. Wang, Y. Botros and J.W. Martin. Enabling Robustness and Flexibility of Equipment Data Collection through SEMI EDA Standards. In Proceedings of Advanced Semiconductor Manufacturing, May 2004
[11] Technical White Paper, Rapid Development of the SEMI Equipment Data Acquisition Specification Suite on the Microsoft.NET Platform
[12] http://java.sun.com/javase/technologies/realtime/index.jsp
[13] http://www.netbeans.org/kb/articles/java-rts.html