Automated High-Accuracy Hybrid Measurement for Distributed Embedded Systems

Marcus Thoss

Distributed Operating Systems (DOPSY) Group, Department of Computer Science,
Fachhochschule Wiesbaden - University of Applied Sciences, Wiesbaden, Germany
thoss@informatik.fh-wiesbaden.de

Abstract *— This paper presents the design and realisation of a hybrid, high-accuracy, automated measurement framework.*

A hybrid measurement system is introduced that has been successfully used to carry out high-accuracy measurements in distributed automation environments. The application of the hybrid measurement system is not limited to such environments, though. It is characterised by monitoring distributed events with low interference and by the usage of global timestamps based on the high-resolution clock of a logic analyser. Furthermore, it exhibits an extensively automated experiment control and data analysis framework. Measurement results gained with the approach are presented as well.

1 Introduction

The measurement of distributed, and possibly real-time and embedded systems provides a challenge for researchers and developers. For such systems, contradictory requirements like high accuracy, a global time base and limited in-system timing facilities make the venture of performance analysis a difficult task.

For this study, a crucial requirement, which also led to the inception of the approach described here, is that an accurate measurement of single events must be possible; cumulative data consisting of averaged values is not sufficient. While this requirement can be alleviated for many applications, it becomes indispensable whenever timestamp information is to be mapped accurately onto single events, especially when the events of interest expose singular characteristics like spikes or glitches.

Specifically, end-to-end delays of individual message transfers, which form an important aspect of distributed scenarios, cannot be measured using local clocks only. Lacking a reference time whose scope extends beyond the clock of a single system, only round-trip time values can be obtained with such clocks. End-to-end measurements, on the other hand, are necessary for the introspection of protocol stack delays. They are indispensable if one-way calls must be regarded. For this, a time source external to all participating nodes can be used, and the events of interest can be mapped onto this clock. In this paper, such a clock will be referred to as "global clock", providing a "global time", although the scope of this time base is strictly limited with regard to two aspects: for the approach presented, the global clock must be physically connected to the nodes being measured and the validity of the clock mapping is limited to the duration of a consecutive series of measurements in an experiment.

Because there exist many notions of and approaches towards establishing a global clock, it must be pointed out that the concept described here does not claim to provide a global time solution for distributed systems. Solutions offering fault tolerance and dynamic, run-time oriented characteristics like the establishment of a global time base using time synchronisation protocols are already well-described[1]; they form a field of active research.

The measurement system presented here can be classified as a hybrid measurement system [2] consisting of hardware and software measurement components. Hybrid measurement systems with better scalability and even higher accuracy have been presented in other works, for example in [3], [4]. In contrast to those often highly customised environments, the system used in this study was designed to be implemented with standard test equipment hardware and low-latency instrumentation code. For the latter reason, it was also chosen to place instrumentation calls in-line within the application code instead of introducing a (possibly library-based) abstraction layer. An approach for monitoring components when direct access to the application source code is not available is presented in [5].

The solution described here was first designed and implemented to completely avoid the usage of local clocks. In a later re-application to a different series of experiments, it has been successfully modified to integrate locally generated timestamps with the initial approach. Since the latter variant is regarded as an extension of the former, with certain benefits for a subset of existing problems, both are presented in this paper in the order mentioned above.

Another important aspect of distributed measurement scenarios occurring in experiments is the space of configuration parameters for the experiment. Ideally, an experiment control is provided that supports iteration over all parameter values of interest over the course of the experiment. For complex scenarios, manageability must be achieved not only at the data collection level, but also with respect to a (partially) automated experiment conduction. The approach presented here therefore supports the conduct of automated experiments. With all aspects combined, the result is an automated high-accuracy hybrid measurement system.

2 A Hybrid Approach

For distributed systems, events initially occur at a local scope but their effects are often promoted throughout adjacent nodes or even over the whole system. To observe and quantitatively analyse such events of interest, they must be mapped to a common time base, which will be referred to as "global clock" in the following. Although, generally speaking, there is no such entity as a global clock, a device within a distributed system providing a unique clock can be nominated a global clock for the purpose of defining a time base that all measurements in the distributed system can be referred to.

Since observing the inner mechanisms of distributed nodes directly from the global clock site is seldom practical, the occurrence of an event must often be determined locally. This, on the other hand, means that the information about the exact time when the event happened is lost for the observer at the global clock. To overcome this discrepancy and provide a feasible, practical solution, a hybrid approach was chosen for the system described here. The primary observation of occurrences of events remain separated from the identification of the timestamps attributed to these events. Instead, the presented approach bridges the gap between the node generating the event and the

global clock source. This is achieved by providing a solution for mapping the event to the global timestamp representing its occurrence. Specifically, the approach uses explicit instrumentation of the application code to be measured and a wired connection of the distributed nodes to the global clock device.

2.1 The Global Clock

As a precondition, the global clock device to be used here must provide an accurate, high resolution time base and trigger input ports for event signals generated by the measured nodes. These inputs are used to trigger timestamp recordings in the device providing the clock. Such functionality is found in standard multi-channel hardware logic analysers. Obviously, the events recorded in the global clock device are ordered in accordance with their occurrence provided that the latencies introduced by event generation mechanisms in the measured nodes and the propagation delays over the connections to the recording device are equal, respectively. This can be accomplished using low-overhead I/O port access and symmetrical configuration.

Recording of single events in this way contrasts the measurement of the cumulated time of a series of looped events which is used in many studies due to a lack of appropriate hardware-based measurement systems. Such cumulated values, averaging the latency values of all events in the loop, could not provide an appropriate level of detail for a discussion like it is required by the assumptions made here.

2.2 Event Recording and Mapping

Besides the ordering of recorded events within a global time line, an assignment of the samples to application-level events is necessary, as depicted in Figure 1.

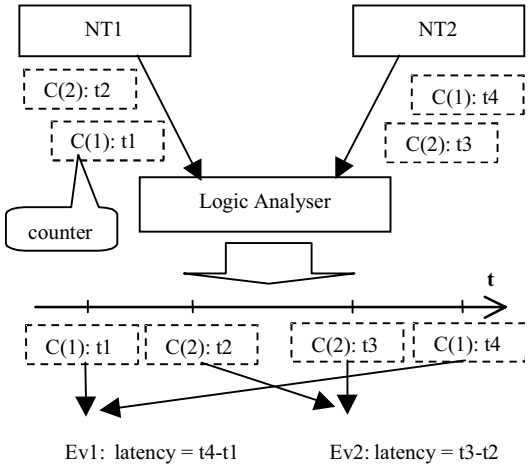

Figure 1: Counter-based Assignment of Events

Here, as an example for the mapping of correlated pairs of events, a transmission of packets from NT1 to NT2 over a network is assumed. The transmission and reception of packets form the set of events to be measured. This requires identifying pairs of transmit and receive event timestamps that belong to the same message, respectively. For this, additional information is provided as a n-Bit tag value (C in Figure 1) that is provided by the instrumentation code with every event signalled over the lines con-

nected to the timestamp recording device. This value, implemented as a rolling counter of subsequent messages, is embedded within every message transmitted. Thus, the counter value is available for being recorded together with the event on the receiver side as well. Assuming an only partially disordered transmission sequence, matching events can be identified easily during the subsequent processing of the sample data.

In Figure 1, two transmissions, which are assigned the counter values C(1) and C(2), generate events on NT1 and NT2, respectively. First, the recording of timestamp values t1..t4 by the logic analyser permits an ordering of the events with respect to global time t. In the next step, the counter tags are analysed, which identify the application level events Ev1 and Ev2 with latency values t4-t1 and t3-t2 attributed to them.

The maximum acceptable displacement is 2^n packets, which results from the n-Bit tag width available.

3 A Modified Approach Adding Local Clocks

Although the approach described so far had been successfully applied in a study [6], a similar experiment that was devised in a later study proved to be infeasible without modification. While the same node and clock hardware was used, the number of operating nodes and the amount of context data to be recorded with every timestamp had to be significantly greater. Therefore, a modified approach was devised that could make use of the local timers running in the processors while retaining the important global clock characteristics of the approach described previously. In the following, the latter is referred to as the "original approach".

Most modern desktop and workstation microprocessors offer a high-resolution timestamp counter running synchronously with the processor clock. Embedded systems are almost always equipped with timer components. The existence of such clocks permits the recording of processor-local timestamps in the instrumentation code.

To regain the ability to compare timestamps recorded for distinct nodes, the local timestamps must be mapped to a common time base. Here, the concept of establishing the mapping to a global clock is inherited from the original approach. At the beginning and at the end of the experiment, the central timestamp recording device records a single timestamp on behalf of each of the participating nodes. For this, the mechanisms of the original approach are used. Thus two pairs of reference timestamps are obtained for every host, with one timestamp denoting the same point in time for the local and the global time base, respectively.

After the experiment has been conducted to its end, a set of local timestamps resides on each host. Before the measurement results can be evaluated, a final step is now introduced that must be performed on all local timestamps: To determine the global time equivalent of a locally recorded timestamp, the time space of the local PC timer must be transformed into that of the logic analyser. Equation (1) shows the transformation using indices *LA* and *PC* for the global and local clock time spaces, respectively, with symbols *T* denoting the reference timestamps and *t* denoting the timestamp to be transformed and the result, respectively.

$$t_{LAn} = \left(t_{PCn} - T_{PC1}\right) \cdot \frac{T_{LA2} - T_{LA1}}{T_{PC2} - T_{PC1}} + T_{LA1} \qquad (1)$$

Thus, the measurement results are processed, resulting in sets of locally recorded timestamps with attributed context data and a global timestamp mapping. Linear drifts of local clocks are eliminated by the mapping. The neglect of non-linear drifts is considered an acceptable simplification for the limited time covered. A recent study addressing issues of drifts among local clocks is described in [7].

An implementation of this modified approach relies on the global timestamp recording, which is now only used for the reference points, and a small amount of instrumentation code to record the local timestamp and additional context data. If the amount of timestamps to be kept is known to be sufficiently limited, a linear memory buffer can be used to hold the measurement data. The buffer is flushed to a storage medium at the end of the recording session. Access to the timestamp counter register of the processor is commonly available using utility functions or macros in most operating system environments, which further simplifies the implementation.

4 An Example Experiment

For an experiment, which is shown in Figure 2, using the first approach described, the global clock is part of a standard multi-channel hardware logic analyser with a resolution of 10 ns and a buffer of 32k events. Because the time span covered by measuring in free run mode is only about 320 µs at the highest resolution, which is too short for a statistically relevant amount of data to accumulate, a method for measuring timestamps for single events with high accuracy was developed for this study.

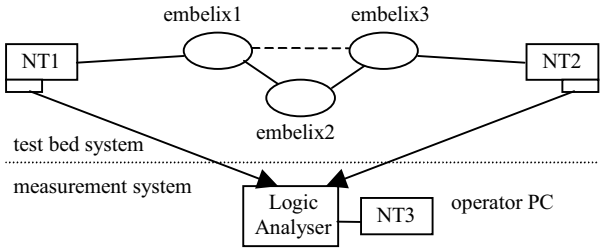

Figure 2: Test Bed Set-up

The logic analyser has an internal state machine that can be set-up to fire transitions on selectable patterns occurring on its measurement pods. During one state, a finite number of events can be recorded. Exploiting these capabilities, a state machine that records single events with high-precision timestamps for every toggle of one of three input ("trigger") lines was implemented. The flow of the state machine is optimised such that only two sample event values are recorded for every external event. Thus, a maximum of 16k events can be recorded with the internal time base of the logic analyser still running at the highest possible frequency of 100 MHz, equalling a resolution of 10 ns. The data recorded includes the state of all inputs connected, such that additional tag information can be passed in the bit pattern that is present on the additional lines during the toggle of a trigger line. It fully satisfies the criterion mentioned in the introduction, measurement of single events.

The connection between distributed nodes and the global clock source can be any kind of digital I/O port that is wired to the measurement pod of the logic analyser. For the implementation test-bed used to realise the measurement approach for a concrete

experiment in a study[6], Siemens automation PC hardware was used. For the recording of events, the logic analyser was connected to the parallel printer port of each of the PCs running the experiment.

The experiment configuration reflects the packet transmission scenario mentioned in the above description of the approach, consisting of the packet source and sink hosts NT1 and NT2 running Windows 2000 and a routing network of three Linux hosts embelix1-3. The instrumentation code that is inserted in the monitored applications consists of calls into a custom low-latency parallel port kernel driver for Windows 2000 and port I/O instructions directly accessing the parallel port registers on the Linux machines, respectively. By determining the maximum frequency of a looped event output, an average latency of 17 µs for a single instrumentation call of the Windows version was determined with a mean probable error below 0.1 µs.

Regarding the tags attributed to events, even a small value of 5 bit proved to be a reasonable trade-off: No artefacts like inexplicably high latency values showing a violation of this limit could be identified in any of the measurements performed during the experiment.

Specifically in experiments using the modified approach also described above, the performance counter of an Intel Pentium III processor was used as local clock device.

5 Comparison of the Approaches

The two approaches taken both have advantages which play out differently, depending on the measurement context. One important assumption made when using local counters is the neglect of non-linear drifts of the counters against each other and the against the global time base. This assumption may hold sufficiently when

- identical hardware is used,
- conditions affecting the clock rate like voltage and temperature changes can be avoided, and
- the processors themselves run at a constant clock speed.

The last point is to be considered for systems capable of altering the processor clock speed, or, generally, the clock speed of the timers used. Embedded processors and especially modern "mobile" variants of desktop processors can be run with lowered clock rates to reduce power consumption. For many systems, the instrumentation code cannot determine such effects. But even if there exists a feedback mechanism, its introduction into the measurement system would increase its complexity too much to warrant the benefits. For such systems, the use of a single, global clock is certainly preferable.

On the other hand, three advantages of the approach using local clocks can be identified:

1. Less overhead for every timestamp taken.

The effort required to generate an external signal for the central timestamp recording device is noticeably greater than reading a local timer register. In the experiments described here, the parallel port register access is achieved through a device driver call, incurring a constant delay and certainly adding to the jitter. Even if the code can be simplified by granting the application direct write access to the port registers, I/O bus access, port gate switching delays and edge sampling effects in the connection to the logic analyser increase the absolute delay time and its jitter.

2. Limited number of hosts that can participate in the experiment.

The state machine mechanism devised for collecting single high-accuracy timestamps over long periods of time is limited by the maximum number of states and the number of distinguishable predicates allowed by the logic analyser. For the device used in the experiments, only two nodes could be managed. Using the local clock approach, the number may be much higher, because collisions of the signals to be detected by the logic analyser can only occur during set-up time, which can be taken into account when planning the experiment.

3. Context data collection.

Depending on the nature of the experiment, more than a small rolling counter value might have to be recorded for every timestamp. This is still possible using a global clock only, because context data can be recorded locally, marked with the counter value transmitted to the logic analyser. After the measurement phase, these can be correlated with the timestamps, resulting in a complete set of timestamps with context attributes. Of course, this correlation has to be implemented first. But the existence of complex context data already hints at the complexity of the instrumentation point space. Only a small set of distinguishable characteristics is available with the timestamps recorded (the source node and the integer tag). Therefore, mapping of the timestamps onto the context data can soon become infeasible.

6 Automating the Experiment

6.1 Logic Analyser Control

The operation of the logic analyser in the experiment described above is controlled by a PC host using a proprietary control software package. The interfacing functions are also available as a shared library (16-bit Windows DLL). To facilitate experiment control and measurement data processing, an OLE Automation Server was wrapped around the DLL. It provides both access to the 16-bit library from 32-bit Windows applications and an interface that can be easily integrated with OLE Automation-capable applications like the Microsoft Office product line with their Visual Basic for Applications (VBA) programming facilities.

6.2 Experiment Control

To illustrate the integration of control facilities with an application to be measured, this section further details the experiment as depicted in Figure 2: The hosts NT1 and NT2 run a Windows DCOM client and server application, respectively, exchanging datagrams of configurable size. To fully automate the operation of the experiment, the client application to be measured was built to listen on a datagram socket for control messages, like shown in Figure 3. With a parameter message, a measurement is configured including the cycle time, the payload packet size and the reservation type. A manual set-up must be carried out for the background load level since the load generator has not been wrapped with a control interface; the same applies to the selection of the DCOM marshaling mode used. Two pairs of client/server executables were built for this purpose, each implementing one method. These applications have to be started manually according to the intended marshaling method. As this description shows, some components in a sufficiently complex and possibly heterogeneous set-up might

not be selectable for automated parametrisation. At the planning stage of an experiment it should therefore be attempted to minimise the number of these components requiring manual operator intervention.

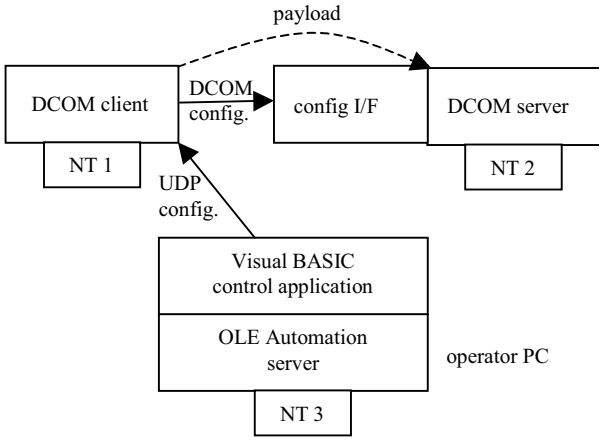

Figure 3: Experiment Control

The execution of the measurement is triggered by sending a start message to the datagram socket of the DCOM client. Before this, the logic analyser is set to running mode by issuing a call on its OLE Automation Server. When all invocations for this measurement have completed, a *finish* message is sent back by the client application to the experiment control module, and the logic analyser is stopped. Thus, the execution of the DCOM invocations to be measured and the gathering of sample data are operated synchronously.

A notable advantage of providing an OLE Automation interface is the ease of integration with commonly used analysis and report generation applications on Windows platforms. For this experiment, the Microsoft Excel spreadsheet application was chosen. It allows the programming of macro modules using the VBA dialect of the Visual Basic language, including access to COM socket programming objects. Using these facilities, a set of modules was created to fully automate the complete experiment control and data acquisition from the logic analyser.

Upon launch of the initial macro, the experimenter is presented a dialog box, prompting for parameter values that are to be used during the execution of the experiment run. Client and server applications are then initialised using the UDP and consecutively the DCOM configuration interfaces. Finally, the logic analyser is set up and the experiment is started. Upon completion, the measurement data is read from the logic analyser using the OLE Automation server interface. After the values thus gathered have been sorted and reduced by data preparation macros that are also part of the automated experiment control, the calculation of statistical data is performed on several newly created Excel sheets. In a last step, even report diagrams like the one shown in Figure 4 are generated automatically.

Thus, no manual step is left between the choice of measurement configuration data and the viewing of readily prepared analysis charts. The macros involved in the measurement and preparation process are easily customisable as far as end-to-end sample data is to be processed. For different measurement tasks, mainly modifications of the intermediate data processing macros and possibly other diagram generation macros could be necessary.

7 Measurement Example

The advantages of the approaches described here are best shown using an example measurement. The experiment set-up is shown in Figure 2. The task is to determine the end-to-end latency of packets transferred from NT1 to NT2 over the network consisting of embelix1-3. Specifically in this experiment, a maximum latency time is to be honoured, which is achieved through bandwidth guarantees using a resource reservation protocol (RSVP). Details of this experiment are described in [6]. Multiple dimensions of tuneable configuration parameters like transmission cycle time, background load, packet size and reservation strategy required a repeated conduction of the measurement run with parameters changed each time. Figure 3 shows the end-to-end latencies of a measurement consisting of samples of communication events for a fixed packet size of 256 byte and a cycle time of 20 ms with RSVP reservations.

From this diagram, the advantages of the single-event measurement system are most obvious. Single events that violate the boundary could not be measured using cumulated values. In addition, it is possible to estimate the behaviour of the delays over time and discover artefacts like the one shown in Figure 3. Thus, configuration parameter sets of interest can be identified at a glance and can be selected for further investigation.

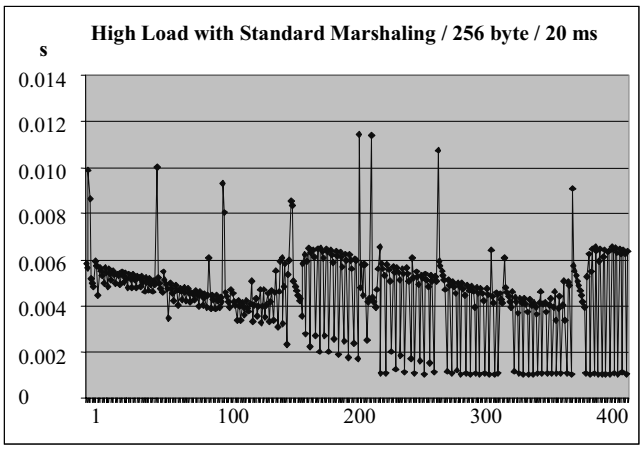

Figure 4: Example Measurement

For [6], a total of more than 400 measurements each consisting of 400 samples was taken. Taking into account that a row of test set-ups had to be evaluated before the final experiment environment could be measured, the total number of measurements was actually much higher. Apparently, experiments exposing multidimensional parameter spaces require automated control to become feasible, given the limited time, personnel and funding resources of most laboratories.

8 Summary

In this study, a hybrid, high-accuracy measurement system to carry out automated measurements was devised. It is characterised by monitoring distributed events with low interference and global timestamps based on the high-resolution clock of a logic analyser. Furthermore, it exhibits an extensively automated experiment control and data analysis framework.

The automated measurement system presented was applied successfully to run experiments with almost unattended operation over a series of parameter configurations. Support of single event recording within a distributed environment is indispensable for the accurate analysis of end-to-end latency samples. It further allows the discovery of singular phenomena which would never surface in a set of aggregated samples. Our measurement system exposes these features together with a low-interference instrumentation for signalling the events, recording them with high-resolution timestamps.

In particular, the benefits of performing measurements with minimal operator intervention are twofold: Time is saved for the operator and errors in the conduction of the measurements and analysis of the recorded data are prevented. Even the often tedious task of report generation is covered by the framework. Thus, judgements on the proceeding of the experiment by browsing initial analysis results is possible at an early stage, and the adjustment of parameters in the test bed system leads to immediate feedback at a high level of abstraction.

Acknowledgments

The study described here was done as part of a research cooperation of the DOPSY laboratory and Siemens AG Automation & Drives. The author would like to thank the DOPSY members and the Siemens staff involved, Georg Biehler in particular, for their support.

References

[1] H. Kopetz. *Real-Time Systems: Design Principles for Distributed Embedded Applications.* Kluwer Academic Publishers, Boston, Dordrecht, London, 1997

[2] D. Ferrari, G. Serazzi, and A. Zeigner. *Measurement and Tuning of Computer Systems.* Prentice Hall, Inc., Englewood Cliffs, 1983.

[3] R. Hofmann, R. Klar, B. Mohr, A. Quick, and M. Siegle. Distributed Performance Monitoring: Methods, Tools and Applications. *IEEE Transactions on Parallel and Distributed Systems*, June 1994: 585-598.

[4] F. Lange, R. Kroeger, and M. Gergeleit. JEWEL - Design and Implementation of a Distributed Measurement System. *IEEE Transactions on Parallel and Distributed Systems*, December 1992.

[5] G. Hunt and M. Scott. Intercepting and Instrumenting COM Applications. In *Proceedings of the 5th Conference on Object-Oriented Technologies and Systems (COOTS'99)*, pages 45-46, San Diego, CA, May 1999.

[6] M. Thoss, R. Kroeger, and G. Biehler. Improving DCOM Communication for Automation Environments. In *Proceedings of the 2003 Summer Computer Simulation Conference (SCSC'03)*, Montreal, July 2003.

[7] H. Kopetz, A. Ademaj, and A. Hanzlik. Integration of Internal and External Clock Synchronization by the Combination of Clock-State and Clock-Rate Correction in Fault-Tolerant Distributed Systems. *The 25th IEEE International Real-Time Systems Symposium,* Lisbon, Portugal, Dec. 2004.