verum

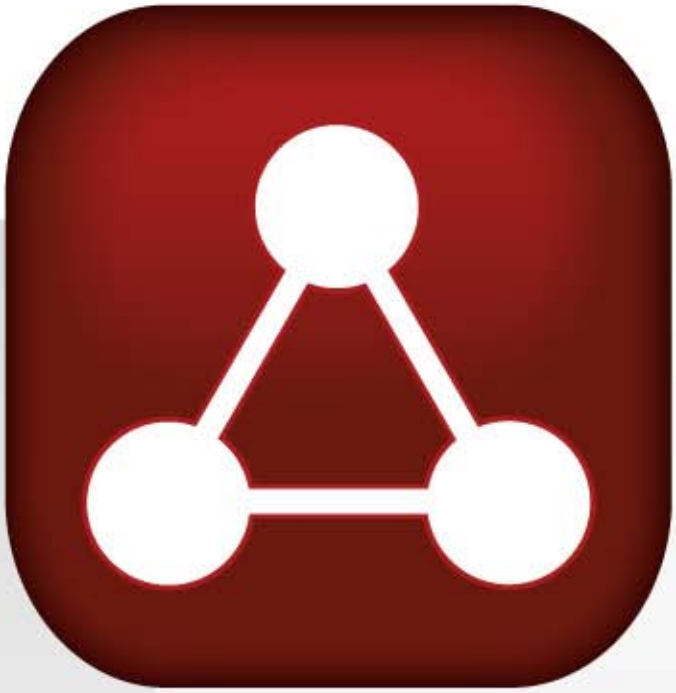
Tools for building mathematically verified software

# Developing device drivers using ASD:Suite

## NXP pilots Verum's ASD for $I^2C$ device driver software

*Authors: Freek van Wetten and Arjen Klomp*

*Version: 1.0, 12/10/2009*




The simple way to build complex software systems

Laan van Diepenvoorde 32 - 5582 LA - Waalre - The Netherlands - Tel: +31 402359090 - Fax: +31 402359099 - info@verum.com

## Executive summary

In this Analytical Software Design (ASD)[1] project, NXP's IP and Architecture group successfully used Verum's ASD:Suite to model the behaviour of an $I^2C$ device driver. The ASD:Suite had been demonstrated in other case studies to save time and cost for software teams developing and maintaining large and complex systems. But this was the first time ASD had been applied to driver software.

NXP is a leading semiconductor company founded by Philips. Working with Verum, NXP undertook a project to evaluate the benefits of using the ASD:Suite for upgrading its device driver software.

*Verum's ASD:Suite helps embedded software developers do their job more quickly and efficiently. The tool makes it easy to build a model, validate it, produce correct code and integrate it into the rest of the environment.*

The case study results can be summarized as follows:

- Verum's ASD:Suite fully captured and documented the $I^2C$ device driver's dynamic behaviour, and in particular clarified the hardware/software interfaces
- NXP established that with the ASD:Suite it is possible to produce higher quality software in less time, while lowering future maintenance costs
- The ASD:Suite generated defect free code that delivers performance at least equivalent to the existing product and at a size that is acceptable for embedded devices

The project showed that Verum's ASD:Suite generates verifiably correct software that requires far fewer test cases to thoroughly test than using a conventional approach, and which is practical for device drivers. The ASD:Suite helps embedded software developers do their job more quickly and efficiently. The tool makes it easy to build a model, validate it, produce correct code and integrate it into the rest of the environment.

Following the success of this project, NXP is looking at other opportunities to gain the benefits of using the ASD:Suite; for example, in its smart card device business. NXP expects to achieve a further gain in productivity and quality when it includes hardware models in the ASD flow.

"ASD provides a framework that helps you model and design more thoroughly. It forces you to think about the requirements and cover all the exception cases."
**Ruud Derwig**, Technology Manager at NXP

[1] An introduction to ASD, http://www.verum.com/resources/papers.html

# 1 Introduction

NXP's IP and Architecture group is responsible for developing and maintaining reusable intellectual property (IP) that can be applied across multiple NXP business lines.

The group was developing code for an $I^2C$ device driver, for next generation systems. It was interested in the possibility of using Verum's ASD:Suite to upgrade the product, without the risk of inadvertently introducing new errors into the code. The ASD methodology generates code that is guaranteed to be defect free.

To learn more about the capabilities of the ASD:Suite and its underlying technology, the group carried out a joint pilot project with Verum in which ASD was applied to model the behaviour of the device driver.

The main objectives of the case study were to:

- Verify the benefits of ASD. The most important benefits of interest to NXP were:
    - Delivering a higher quality product with the same or reduced amount of development time and effort
    - Reducing costs of future maintenance
    - Achieving at least equivalent performance to the existing product
- Determine if ASD modelling helps in verifying the correctness of device driver software
- Assess whether Verum's ASD:Suite is a useful and practical tool for developing device drivers

This paper presents an overview of how the ASD:Suite was applied to the development of a Linux version of the $I^2C$ driver software, hosted on NXP's Energizer II chip, and draws conclusions based upon these objectives.

## 2 Current situation

As a leading supplier of semiconductors, NXP provides chips that are integrated into a wide variety of products. Customers include major manufacturers such as Philips, Sony and Samsung, which sell their products worldwide.

$I^2C$ (Inter-Integrated Circuit) devices are used in many systems on chips (SoCs) for controlling peripheral devices in other chips. These are typically employed in products such as digital TVs, set-top boxes, automotive radios and multimedia equipment, as well as in more general purpose controllers for vending machines etc.

Quality is a major concern, especially for the automotive market, where end user products are installed in high-end, expensive vehicles.

*There were issues that caused problems in development and testing, largely stemming from an incomplete definition of the hardware and software interfaces.*

The $I^2C$ driver software had been ported and updated many times, both to keep pace with the evolving hardware platform and to cater for the requirements of new and upgraded target operating systems.

The existing device driver suffered from timing and concurrency issues that caused problems in development and testing, largely stemming from an incomplete definition of the hardware and software interfaces. In addition, the costs of maintaining the software were high.

NXP was keen to find out whether Verum's ASD technology could help to deliver a higher quality product without any extra development effort. NXP defines quality in terms of:

- performance
- stability
- robustness
- correct functionality
- code size

# 3 Method

The ASD:Suite had been demonstrated in other projects to shorten timescales and reduce costs when used in the development and maintenance of large and complex systems. This was, however, the first time it had been applied to device driver software.

The project covered two key activities:

- Modelling of the device driver software
- Automatic generation of C code

NXP provided the domain knowledge for the project. Verum provided assistance and training in the use of the ASD:Suite.

*The dynamic behaviour of the original driver was largely unspecified and unclear in the original design. The application of ASD clarified the interfaces...*

## 3.1 Modelling the hardware-software interfaces

The dynamic behaviour of the original driver was largely unspecified and unclear in the original design.

The application of ASD clarified the interfaces, resulting in a better understanding of the behaviour. For example, the software interface, which NXP calls the hardware API (HWAPI), was assumed to be stateless. However, ASD modelling revealed HWAPI state behaviour that had not previously been documented.

## 3.2 Improving the design

The design of the original $I^2C$ driver software relied heavily on interrupt service routines. Using ASD it was possible to generate asynchronous code with kernel threads that could be proven to have no deadlocks or race conditions.

## 3.3 Major reduction in testing effort

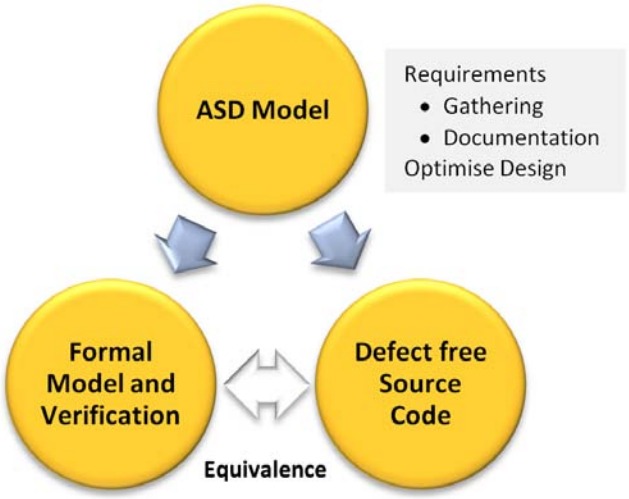
Because the ASD:Suite produces verifiably correct code, the number of test cases NXP needed to run to gain confidence in the final product was considerably less than it would have needed using a conventional approach to software development. The ASD:Suite ModelChecker revealed that there were more than 700.000 unique execution scenarios for the device driver. So without ASD, NXP would have required over 700.000 test cases to thoroughly test the software.

### 3.4 ASD methodology

At all stages of a project, ASD technology is applied to the various parts of the device driver software, and can also be applied to the hardware. In summary the approach was:

1. Using Verum's ASD:Suite, gather and document requirements, capturing them in the ASD model
2. Generate a formal model of system behaviour and verify the correctness of the design (before any code has been generated)
3. Generate the source code, guaranteed to be defect free and functionally equivalent to the formal model

*ASD generates defect free, high performance code that helps embedded software developers shorten project timescales, reduce the cost of maintenance and improve product quality.*




### 3.5 Key benefits

When a software developer has gained familiarity with using the ASD:Suite, it is possible to improve the efficiency of the development process by reducing dependence on testing, cutting development timescales by typically 30%. Maintenance becomes easier because changes are implemented to the model, not the code, which can then easily be regenerated.

Verum's ASD:Suite delivers high performance software with low resource requirements, which is suited to highly embedded systems. Most importantly, the quality of the software improves, because code defects are eliminated.

## 4 Results

NXP focused on measuring whether two of its key objectives had been met.[2]

Following completion of the project, NXP carried out:

1. Extensive stress testing, to check the stability and robustness of the product, and that the functionality was correct
2. Performance analysis, in terms of speed of operation and footprint

*NXP's overall impression of the ASD:Suite was favourable. The pilot project clearly demonstrated the ASD:Suite's applicability to device drivers and how it will enable NXP to achieve higher quality.*

The results of these investigations were positive, leading NXP's management to decide to promote ASD for new projects inside NXP.

NXP concluded that the size of the generated code is small enough although it would like to see the footprint reduced in future releases of the ASD:Suite.

The software performance, in terms of speed, was an improvement over the previous design. Although the ASD designed code was about 2% slower it significantly reduced the time spent in the interrupt service routine by a factor of 3, giving an overall responsiveness improvement. Additionally, NXP found the correct modelling of complex hardware interfaces is a key challenge which will require more attention in future device driver projects.

NXP's overall impression of the ASD:Suite was favourable. The pilot project clearly demonstrated the ASD:Suite's applicability to device drivers and how it will enable NXP to achieve higher quality. ASD modelling and the use of the C code generator will also improve the maintainability of NXP's products.

But for NXP the greatest benefit is the way in which the ASD methodology and the ASD:Suite support a specification process that forces software developers to think carefully about product requirements and ensure all execution scenarios are covered.

[2] Owing to the existing device driver software having more extensive features, and having been ported and updated many times, it was not possible for NXP to make a direct comparison of development costs using ASD against what these would have been without ASD.

## 5 Conclusions

NXP believes Verum's ASD:Suite provides major benefits by enabling its software developers to model system behaviour correctly, verify the model and automatically generate defect free code.

One of NXP's business lines produces chips for smart cards, which have strong security and certification requirements. Being able to prove, through the use of the ASD:Suite, that the embedded software is correct will help NXP gain certification.

*The NXP case study has clearly shown that ASD modelling helps in developing and verifying deeply embedded software, and that Verum's ASD:Suite is beneficial and practical for device drivers.*

The NXP project has clearly shown that ASD modelling helps in developing and verifying deeply embedded software, and that Verum's ASD:Suite is beneficial and practical for device drivers.

It was not possible in this NXP case to make a direct comparison of development effort with and without ASD, but other studies have shown that using Verum's ASD:Suite can reduce development time and costs by around 30%.

NXP believes that to achieve this productivity gain it will need to generate formal hardware models using ASD, so that the hardware-software interface can be checked and proven correct. Currently, most effort expended in the development of device driver software is in understanding the behaviour of the hardware and the protocol used for controlling it. Typically the hardware-software interface is not fully specified, and hardware datasheets do not always contain all the required information.

NXP's own investigations have demonstrated the quality of the product and the performance of the ASD:Suite generated code. Additional benefits include much easier and less costly maintenance.

Even where there are requirements with timing constraints that cannot be modelled using ASD, it is a major advantage for developers to be able to perform manual timing checks on code that has been guaranteed correct with respect to the functional specification. Further, the analysis of all possible execution scenarios by the ASD:Suite means developers can be confident there are no deadlocks or race conditions.

## 6 About the authors

Freek van Wetten is a Senior Systems Designer at TASS Software Professionals.

Arjen Klomp is Commercial Manager at Verum Consultants BV.

## Acknowledgements

We are grateful to NXP for allowing us to present this case and for their cooperation in applying ASD to the development of the Linux $I^2C$ driver for IP3204 on the NXP Energizer II platform chip.