System validation, 2IW26

Jan Friso Groote (J.F.Groote@tue.nl, MF 7.070), Julien Schmaltz (J.Schmaltz@tue.nl, MF 7.071b).
`http://www.win.tue.nl/~jfg/educ/2IW26/lente2014/overzicht.html`

The purpose of this course is to learn how to specify behaviour of systems and to experience the design of a system where you can prove that the behaviour is correct. So, you will learn how to formally specify requirements and to prove (or disprove) them on the behaviour. With a practical assignment you will experience how to apply the techniques.

The lectures are mainly dedicated to learn the foundations of the specification language mCRL2 and to use it as a manual specification and verification tool. There will be 2x2 hours of lectures per week, on Wednesdays 3/4 and Fridays 1/2.

In parallel to the lectures there will be an assignment, which must be finished before the examination period at the end of the semester. The goal of the assignment is to apply the techniques and tools to the design of a small distributed and/or embedded system. The purpose is to design this system such that it is proven to comply with all the requirements which must have been formulated in advance.

The marks for the exam and the assignment contribute equally to the final score. The final mark is the average rounded to a whole number in the ordinary way (7.5 is rounded up to an 8, 7.49 is rounded down). The mark of the resit of the exam will consist for 50% out of the result of the assignment and for the other half of the result of the exam. The result of the assignment is only valid for one year. If the course is redone in a subsequent year, the assignment must be redone.

Literature

The course material consists of

- J.F. Groote and M.R. Mousavi. Modelling and analysis of communicating systems. Lecture notes. 2013. Chapters 1, 2 (not 2.3.2, 2.3.3, 2.4.4), 3, 4, 5 (not 5.6), 6, 7, 9 (not 9.7 and 9.8.3), 10, 11.
- See `www.mcrl2.org` for the tools, manual pages etc.

The lecture notes by Groote and Mousavi can be obtained at the Student Shop. The document can also be downloaded from the website (`www.win.tue.nl/~jfg/educ/2IW26/lente2014/overzicht.html`). There will be two versions, one that is equal to the reader, and one in which some errors have been removed. In the summer of 2014 a book with the same title will appear at MIT press. The content of the book and the reader are essentially the same. The exam will cover the indicated parts of the reader as well as everything said during the lectures.

Assignment

The assignment consists of designing a controller for a small distributed and/or embedded system. Below a suggestion for such a system can be found. It is a dynamic voltage and frequency scaling controller to be applied on complex energy aware multi-processor systems. But it is possible to design any embedded controller or distributed algorithm provided you obtain approval by the supervisor of your assignment. The assignment can be carried out in groups of one to four persons.

Carrying out the assignment consists of executing the following steps:

1. Identify in words global requirements for the whole system. Typical requirements are ‘a bridge will never open when the barriers are not closed’. These requirements are initially to be described in natural language.
2. Identify the interactions that are relevant for your system. Describe clearly but compactly the meaning of each interaction in words.

3. Translate the global requirements in terms of these interactions.
4. Describe a compact architecture of the structure of the system. It is required that the controller has at least three different parallel components.
5. Describe the behaviour of all controllers in the architecture using mCRL2.
6. Verify using the toolset that all requirements given in item 3 above are valid for the design in mCRL2.

The assignment must be documented in a technical report that covers all items above. This report must be a concise technical account of the system and must be written such that from it the requirements, action interface, architecture and behavioural design can be easily understood. It must also be clear how the requirements are verified, in such a way that this can easily be redone exactly without consulting any of the authors of the report. So, for instance the exact commands that are used must be listed, it must be obvious which version of the toolset was used for the verification and on which platform and operating system the verification was done.

The assignment is to design a dynamic voltage and frequency scaling (DVFS) controller for on chip use. A contemporary microprocessor consists of all kinds of components. We will restrict ourselves to processor cores and memory caches.

The processor cores and the caches can run at the highest speed when the clock frequency is maximal. If all components run at the highest speed, the processor will overheat and either be damaged, or a heat detector (outside the processor) will cut the power to the processor, which is also undesirable. It is necessary that the cumulative energy usage of all components on the chip is limited.

The DVFS controller can set the voltage v of each component to a few values, typically 0 volt, 1.2volt, 2.8 volt and 3.3volt, but this depends on each chip. The frequency f on which a component can run is limited by each voltage level. The lower the voltage, the lower the allowed frequency. If the voltage is too low and the frequency too high, the component will crash because the switching time will be too low. On the contrary, it is no problem if the voltage is high and the frequency is low. The formula for the heat dissipation is $P = c\,v^2 f$ where c is a constant.

Most of the time processors are idling. In order to save energy components must be shut down completely, or run at a low voltage and frequency. If components are shut down they will loose their information (contents of the cache and contents of the processor registers). A component can be requested to prepare to shutdown. In such a case, caches will try to move their content to main memory, and processor cores will in cooperation with the operating system move running processes to other cores. This component will then let the DVFS controller know that it is ready to be shut down, after which the DVFS controller can cut the power. It can also decide that the power will not be cut, but that a component can continue to operate.

When components are moving from the shutdown mode to any voltage/frequency mode, the component will run at full power consumption for a short while. In order to allow such a component to start up, other components must first be brought to a lower power consumption state.

A processor with a cache needs it to operate. It makes no sense to shutdown the cache but leave the processor running. It can be useful to shutdown the processor, while the cache is still functional for instance when not all the information in the cache is saved to memory yet. If all cores of the processor are shut down, the operating system cannot function. On low energy processors, there is for this reason at least one simple energy efficient processor core, often without cache, that will be used when the processor in mainly idling. It is important to guarantee that always one processor is active in the system.

In order to design such a DVFS controller, the interactions that are needed to let the DVFS controller perform its tasks must be defined. There are many ways to do this, and it is part of the assignment to invent such a communication scheme. We are not concerned how the interactions are actually implemented on the chip. This can be done by dedicated signal lines, but as this often takes

too much space on the chip a special control bus is often used where dedicated sequential messages are communicated between the different components involved in governing the energy usage on the chip.

The description above will turn out to be rather vague if the behaviour of the system is modelled. In those cases you are allowed (even stimulated) to choose your own view on the system even if it is not in accordance with the description above as long as you can defend your choice. One of the reasons for this is that the time to accomplish this assignment is rather short. You should not be delayed by awaiting a verdict from the teachers of this course but resolve modelling and design choices yourself.

Tool set

See `www.mcrl2.org` and the webpage of the course.

Global time schedule

Below a global time schedule is indicated. There are exercises indicated, which upon request can be treated during the lectures. Students are supposed to make these exercises before the lectures, and compare their answers with those of the lecturer. The exercises and short indications of their answers can be found in the reader.

5/7-2-2014. Chapter 1. Chapter 2. Chapter 4. Transition systems, basic processes, process equivalences, conditional operator, time. Elementary reasoning with axioms. The toolset and its philosophy. Exercises 2.2.2, 2.2.3, 2.3.2, 2.3.8, 2.3.9, 2.3.10, 2.4.5, 2.4.6 4.2.2, 4.2.3, 4.3.1, 4.3.2, 4.4.1, 4.5.1, 4.5.2.

12/14-2-2014. Chapter 3. Chapter 4. Section 9.4. Appendix A. Abstract data types. Constructors. Built in data types, bool, quantifiers, pos, nat, int, real. list, set, bag, functions, structured type. Difference between $\approx$ and $=$. Predefined data types, induction. Exercise 3.1.2, 3.1.3, 3.1.4, 3.2.2, 3.2.3, 3.3.3, 3.4.1, 3.5.1, 3.5.2, 3.5.3.

18/20-2-2014. Section 4.6. Chapter 5. Section 9.6. Recursion. RSP. Proving recursive specifications equal. Parallel processes and hiding. Expansion law. Communication, multi-actions. Exercises. 4.6.1, 4.7.1, 4.7.2, 4.8.1, 5.1.1, 5.2.1, 5.4.1, 9.6.4, 9.6.5, 9.6.6, 9.6.9.

26/28-2-2014. Chapter 6. The modal μ-calculus with data. Exercise 6.1.2, 6.2.1, 6.3.1, 6.3.2, 6.4.1, 6.4.2, 6.5.1, 6.5.2, 6.5.3.

12/14-3-2014. Chapter 9. Sketch of the lambda calculus. Sum axioms. Sum elimination theorem. Precise proof system. Exercise 9.4.1, 9.4.2, 9.4.3, 9.5.2, 9.5.3, 9.5.4, 9.5.5.

19/21-3-2014. Chapter 10. Chapter 11. Linearisation of processes. CL-RSP, CL-RSP with invariants. Exercise 10.1.4, 10.2.9, 10.2.10, 10.2.11.

26/28-3-2014. Confluence and τ-priorisation. Exercise 11.1.3, 11.1.4, 11.2.6.

2/4-4-2014. Reserve.

11-4-2014. Exam (14:00-17:00). Closing date for registration: 30-3-2014.

18-4-2014. Last date to hand in the pre-final report for the assignment. The report must be handed in on paper.

28-4-2014. Last date to hand in the final report for the assignment. The report must be handed in on paper and must be accompanied with the corrected pre-final report.

24-6-2014. Exam (resit, 9:00-12:00). Closing date for registration 15-6-2014.