**Journal**

**Exercise 2 HW/SW Co-design**

Date: **07/12-10**

Authors: **Brian Vestergaard Danielsen,**

**Teddy Holm Roskvist,**

**Anders Hvidgaard Poder**

# Introduction

In this journal contains our findings and conclusions for Exercise 2. The first part of the Journal will contain the answers to the questions posed in the assignment, and the appendix contains the detailed rationales and minutes of meetings that lead us to these answers.

# Assignment 2.1

During our first meeting we arranged to come up with our individual suggestions and then agree on a method.

The individual suggestions may be found in the appendix.

The conclusion was that we would use SysML as much as possible, and amend with custom timing diagrams if needed. We did however decide to not use the requirements diagram, but instead simply use a table for the non-functional requirements. We also decided to try and map our diagrams to the Y-chart, in order to maintain a good link to the HW/SW co-design methodology.

For details about the individual diagrams please refer to the appendix, or simply see them in the following parts of the journal.

# Assignment 2.2

We decided to do assignment 2.2 and 2.3 in plenum during our first design session, based on prepared input from the participants with respect to use cases. The final Use-case diagram came to be as follow:



It may be seen that we have included a HW/SW separation, and though this is not officially a part of the Use-case diagram, we would like to illustrate that we already here started to notice an “obvious” separation into HW and SW components.

The rational for this separation may be found in detail in the appendix, but will also be discussed in assignment 2.3, but it is a matter of how likely the functionality is to change, how math-intensive it is, and what its performance requirements is. As for performance we talked about the non-functional requirements, and decided on the following:

|  |  |  |
| --- | --- | --- |
| ReqID | Related UseCase(s) | Description |
|  |  |  |
|  |  |  |
|  |  |  |

Finally there are some design constraints dictated by the assignment.

|  |  |  |
| --- | --- | --- |
| ReqID | Related UseCase(s) | Description |
|  |  | VGA |
|  |  | S-Video |
|  |  | 2 microphones |
|  |  |  |

This is clearly not a complete requirement specification, just like the detailed Use-case descriptions have been left out. This is done on purpose to focus on the architectural design and not the requirements.

Furthermore the assignment calls for analyzing the functionality with diagrams, but this we would like to postpone to assignment 2.3, where the architecture and design will describe the functionality and design.

# Assignment 2.3

There are many ways to document a system like this, and many different diagrams one may choose, but to keep the journal manageable we have decided to use the block diagram type (basic and internal) to describe the composition and communication flow of some of the important components.

Firstly we look at the static structure of the overall system with a basic block diagram



As it may be seen no decision has been made as to what is implemented in HW or SW, except for the parts that is a physical unit, e.g. the physical microphone, which has been moved from actor to HW block.

Looking at the internal block diagram we start adding more detail, and yet we still do not have to decide on HW or SW implementation, but the added details may aid us in our decision.







Instead of doing the remaining diagrams we have decided to focus on the HW/SW separation. Naturally after the HW/SW separation many more diagrams should be drawn, like state diagrams for the Remote Control would be an obvious choice, and timing diagrams for the delay/filter and mixing in Audio Control.

## Separation of HW and SW

There is much information what must be considered when deciding on whether functionality should be mapped to HW or SW:

* Our current platform has no HW support, live with it.
* We have no HW development resources available.
* The given functionality cannot be implemented in SW and meet its deadline.
* The lower unit cost cannot outweigh the added cost and complexity of developing this in HW (neither ASIC or FPGA).
* We already have HW RTL components available for this functionality, so HW implementation is not a problem.

The only direct restriction that we are facing is whether a SW mapping is fast enough, which we may determine by analyzing the number of instructions required to process one sample and the speed at which the samples are arriving – that way we can determine the required size of a CPU or HW (not the same, as required instructions in SW is not the same as the required instructions in HW).

We have decided to do a simple table based approach and look at the different blocks and determine how suited they are for HW mapping, based on the following criteria:

* Performance requirements
  + If there is a high throughput performance requirement then HW is a good alternative to SW, and might also be the only possibility.
* Risk of change
  + Changing HW is much more complicated then changing SW, so if there is a high probability of change to the block then a HW implementation should be avoided.
* Data flow vs. Control logic
  + HW is not very well suited to implement control logic, but is much more suited for simple filtering and other basic block algorithms. A good way to determine this in practice is to look at the state diagram and the breakdown into Basic Blocks. If there are many states and Basic Blocks then it is not very suited for HW-mapping.

Insert table

# Conclusion