HYDROGEN SYSTEM

Timebox 4 – 03-05-2011

This report covers Timebox 4 in the realization phase of the Hydrogen subproject, which is a part of the overall Energy Hub project.

The project is a mandatory part of the 4th semester at the Electronic Design Engineer education at AU-IBT.

The Project has been supervised by Klaus Kolle and Morten Jakobsen both teachers at the Electronic Design Engineering program.

Lasse Lykkegaard

Dennis Thomsen

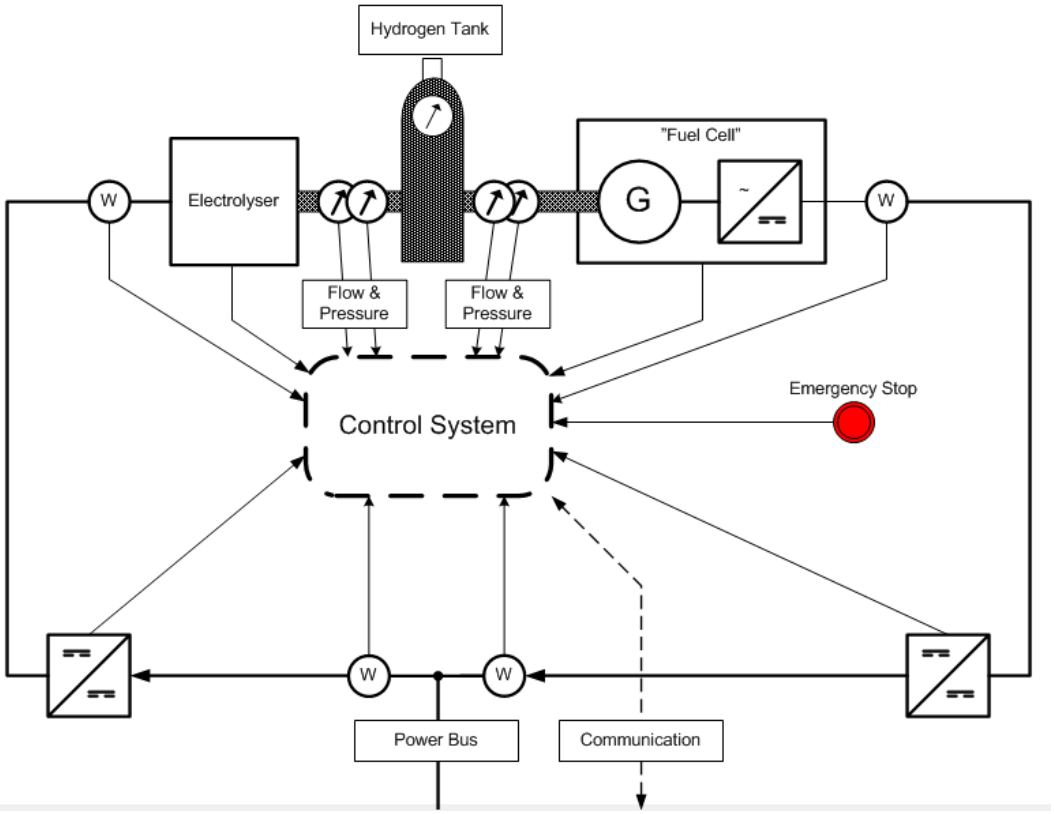
Knud Baastrup

CONTENT

System Overview

Lasse Lykkegaard

Below figure show the System Overview created during the PRO3 Launch project



Change log

This section describes some project adaptations done in PRO4 compared to PRO3.

The altering specifications and the work in the realization phase have revealed some none coherent specifications which have been altered.

The functions getSensorData() does not need the ProductionID argument supplied, when working in the data model...sensors are no longer connected to a Production sytem. Instead it has been supplied with a &data, where the data is written. This makes us able to track errors.
getSensorData(int SensorId, int &data)

The interface between the user-space application and the Data model was previously separated with two interfaces, one for the sensor class and one for the production system class. These two have been merged together into a h2SystemInterface class.

The State Machine diagram of the Fuel Cell and Electrolyzer should be altered to include a super state as emergency shutdown. At least the emergency shutdown should be made a blocking condition. Super state is preferred.

DEPLOYMENT PLAN

Knud Baastrup

Table 1 shows the deployment plans that were initially defined during the strategy planning of timebox 1. The plan has now been updated with more details on the later product versions that in the initial plan just included a heading for the expected deliverables.

Table 1 Deployment Plan

Product Version	Functionality	Deployment Date
1	Start/Stop Electrolyzer and implement Database: The Electrolyzer can be started and stopped from a software application running in user-space on a LPC2478 Development kit. The power bus is simulated by using the XPR 60-100 power supply. Database design is settled and implemented on lene-lasse.dk Participatory session with low fidelity prototype completed. The product version consists of the following artifacts: • Electrolyzer (final version) • Converter/relay-switch (prototype 1) • LPC2478 Development kit (final version) • Driver for converter/relay-switch (final version) • H2 application (version 1) • Database design (final version) • Low fidelity prototype • Report documenting the design	15-03-2011
2	Measure efficiency of the Electrolyzer and implement dynamic web: It is possible to measure the efficiency of the electrolyzer by measuring the energy consumed versus energy produced by the electrolyzer. This requires the ability to measure voltage, current, flow and pressure. The measured data can be read-out from the H2 application sw. Dynamic web site can show measured data (data are simulated using scripts). Video presentation to cover the participatory session with low fidelity prototype. The product version consists of the following artifacts: • Current sensor (prototype 1) • Voltage sensor (prototype 1) • Flow sensor (prototype 1) • Driver for A/D converter (final version) • H2 application (version 2) • Dynamic web site (version 1) • Video presentation (final version) • Report documenting the design	29-03-2011

3	Gain CAN knowledge and finalize PHP Websites: • Establish CAN communication between 2 LPC2478 boards • Login support for PHP website (with java script validation) • Support for Mail notification for sensor data • Improve SQL query for power calculation and improve grap presentation	12-04-2011
4	Final testing of sensor hardware and get started with FPGA support • FPGA Module design including 8 channel ADC • Final testing of sensor HW • Preparation tasks for EMC test • Update of sensor data to MySQL/web server • Preparation of usability test • Read-out of efficiency data via web • Continuously update of sensor data and average filter	03-05-2011
5	CAN support, EMC verification, usability test and finalize and FPGA support:	17-05-2011

STRATEGY PLANNING

Knud Baastrup

We have recognized that it could be beneficial to split the existing requirements into a number of more detailed requirements that together fulfill the required functionality. The increased number of requirements will allow a more detailed planning and as well allow part of an overall requirement to be deployed and signed off by the Customer.

We have also recognized that we are dealing with a good number of more and less unwritten requirements that have emerged during the realization due to the courses WEB2, EMC1, IDE1 and EMB FPGA that has become part of the project. Each course has demanded certain technologies that must be present in the system, but these demands have so far not been defined in terms of requirements.

Table 2 show the complete list of requirements. The original requirements from Launch (shaded rows) have been split into more requirements where feasible and the new requirements emerged during realization have been added to the end of the table. Detailed requirements can be identified by their usage of the 3rd level in the ID column.

The "Timebox" column, added during the strategy planning for timebox 1, has been updated to allow some detailed requirements to be deployed earlier than the overall requirement. The acronym "NP" for "Not Planned" has been added to visualize that some requirements have been down prioritized to the extent that they can no longer be committed for the PRO4 project. This prioritization will be hand-shacked with the customer representative during the timebox 4 deployment.

The currently "Not Planned" (NP) requirements include (among others) the following major bullets:

- All the requirements related to the "Fuel Cell" are Not Planned. The "Fuel Cell" implementation is based on the exact same HW components as the electrolyzer, which includes reuse of the same voltage, current and flow sensors, reuse of the HW for gain and level adjustment and reuse of the relay circuit HW. The HW components can be fully tested using the electrolyzer and we believe it is the best approach to complete these test before we build the HW for the "Fuel Cell", even though we can no longer commit this for the PRO4 delivery data.

  NOTE:
  There will still be SW support for the "fuell-cell", so once "fuell-cell" HW is introduced; the SW task will mainly include some test activities.

- All requirements related to a configuration options for a delay between start and stop of electrolyzer/"fuel cell" and stop and start of electrolyzer/"fuel cell" is Not Planned. These requirements were defined internally (not from customer) during launch and is not required by the CAN protocol.
- Requirements related to measurement of converter efficiency are Not Planned. These requirements were anyway optional and will only make sense if a converter is in use (currently we use the option to run the electrolyzer direct from the powerbus.

The "Signed" column has been added to track the requirements that have been deployed and signed by the customer/customer representative.

Table 2 The complete list of H2 requirements

ID	General Requirements	Timebox	Signed
G1.1	The H2 subsystem shall be able to produce hydrogen from electrical power using electrolysis and should as well store it in a hydrogen tank.	1	
G1.1.1	The H2 subsystem shall be able to produce hydrogen from electrical power using electrolysis.	1	
G1.1.2	The H2 subsystem should be able to store produced hydrogen in a tank.	NP	
G1.2	The H2 subsystem shall be able to produce energy as electrical power using hydrogen fuel cell or any other device that can produce electrical power from hydrogen.	4	
G1.3	The H2 subsystem shall be physical connected to a power-bus with a nominal voltage of 48 VDC. The voltage should be within 48 ±2 VDC with a maximum superimposed AC voltage of 2 V_{pp} in accordance with the Bus Voltage Specification. The power-bus will both deliver the power to be stored as hydrogen and consume the power produced using hydrogen fuel cell.	5	
G1.4	The electrolyzer and fuel cell shall not consume or deliver more than 3KW or 62.5A.	5	
G1.4.1	The electrolyzer shall not consume more than 3KW or 62.5A.	1	
G1.4.2	The fuel cell shall not deliver more than 3KW or 62.5A.	NP	
G1.5	The H2 subsystem shall monitor input/output power-bus voltage and input/output power-bus current and notify the Hub and other subscribed observers on changes in voltage or current level.	5	
G1.5.1	The H2 subsystem shall monitor input power-bus voltage and allow the voltage to be read-out at any given time when the H2 subsystem is up running.	2	
G1.5.2	The H2 subsystem shall monitor output power-bus voltage and allow the voltage to be read-out at any given time when the H2 subsystem is up running.	NP	
G1.5.3	The H2 subsystem shall monitor input power-bus current and allow the current to be read-out at any given time when the H2 subsystem is up running.	2	
G1.5.4	The H2 subsystem shall monitor output power-bus current and allow the current to be read-out at any given time when the H2 subsystem is up running.	NP	
G1.5.5	The H2 subsystem shall notify the Hub on changes in voltage and current level	5	
G1.5.6	The H2 subsystem should notify the Web Server on changes in voltage and current level	4	
G1.6	The H2 subsystem shall be able to deliver current efficiency data to other systems on request.	4	
G1.6.1	The H2 subsystem shall be able to deliver latest efficiency data for electrolyzer on request.	4	
G1.6.2	The H2 subsystem shall be able to deliver latest efficiency data for fuel cell on request.	NP	
G1.7	The H2 subsystem should be able to deliver efficiency data measured over certain duration of time that at least should cover up to 24 hours.	4	
G1.7.1	The H2 subsystem should be able to deliver electrolyzer efficiency data measured over certain duration of time that at least should cover up to 24 hours.	4	
G1.7.2	The H2 subsystem should be able to deliver fuel cell efficiency data measured over certain duration of time that at least should cover up to 24 hours.	NP	

ID	Electrolyzer Requirements	Time box	
E1.1	It shall at any time be possible to start H2 production on request from Hub when all of the following conditions are fulfilled: 1. Defined delay between stop and start has passed. 2. Electrolyzer has not been manually stopped. 3. No power production ongoing. 4. Other (the system must be in ready state) Hub shall be notified once H2 production has started.	5	
E1.1.1	It shall at any time be possible to start H2 production on request when electrolyzer has not been manually stopped and no power production is ongoing.	1	
E1.1.2	It should not be possible to start H2 production on request when the defined minimum delay between stop and start has not yet passed.	NP	
E1.1.3	It shall be possible to manage a request from Hub to start H2 production.	5	
E1.2	It shall at any time be possible to stop H2 production on request from Hub either immediately or when defined delay between start and stop has passed. Hub shall be notified once H2 production has stopped.	5	
E1.2.1	It shall at any time be possible to stop H2 production on request.	1	
E1.2.2	It should not be possible to stop H2 production on request when the defined minimum delay between start and stop has not yet passed.	NP	
E1.2.3	It shall be possible to manage a request from Hub to stop H2 production.	5	
E1.3	The H2 subsystem shall be able to stop H2 production and notify Hub if one of the following conditions takes place: 1. Sufficient power is no longer available for efficient H2 production. 2. Electrolyzer has been manually stopped due to malfunction	5	
E1.3.1	The H2 subsystem shall be able to stop H2 production if sufficient power is no longer available for efficient H2 production.	5	
E1.3.2	The H2 subsystem shall be able to stop H2 production if electrolyzer has been manually stopped due to malfunction.	5	
E1.3.3	The H2 subsystem shall be able to notify Hub on the stop of H2 production.	5	
E1.4	It shall at any time be possible to manually stop H2 production immediately and notify Hub in case of emergency or for maintenance/test purpose.	5	
E1.4.1	It shall at any time be possible to manually stop H2 production immediately in case of emergency or for maintenance/test purpose.	5	
E1.4.1	The H2 subsystem shall be able to notify Hub on the stop of H2 production due to emergency or maintenance/test purpose.	5	
E1.5	The H2 subsystem shall be able to produce hydrogen when sufficient power in terms of 1500W is available for the electrolyzer.	1	
E1.6	The H2 subsystem must be able to convert the power-bus voltage into the required input voltage for the given electrolyzer. Alternatively the electrolyzer should use the voltage given by the power bus despite an expected lower efficiency.	1	

E1.7	Delay between start and stop of H2 production as well as delay between stop and start of H2 production should be configurable via local configuration interface/web interface to H2 subsystem.	**NP**	
E1.8	The H2 subsystem should monitor the input voltage and input current to the electrolyzer to allow calculation of the converter efficiency for the converter required between power-bus and electrolyzer.	**NP**	
E1.9	The H2 subsystem shall monitor the hydrogen flow and hydrogen pressure delivered by the electrolyzer to allow calculation of the electrolyzer efficiency.	2	
E1.9.1	The H2 subsystem shall monitor the hydrogen flow delivered by the electrolyzer to allow calculation of the electrolyzer efficiency.	2	
E1.9.2	The H2 subsystem should monitor the hydrogen pressure delivered by the electrolyzer to allow calculation of the electrolyzer efficiency.	**NP**	
ID	**Fuel Cell Requirements**	**Time box**	
F1.1	It shall at any time be possible to start power production on request from Hub when all of the following conditions are fulfilled: 1. Defined delay between stop and start has passed. 2. Fuel cell has not been manually stopped. 3. No hydrogen production ongoing. Hub shall be notified once power production has started.	**NP**	
F1.2	It shall at any time be possible to stop power production on request from Hub either immediately or when defined delay between start and stop has passed. Hub shall be notified once power production has stopped.	**NP**	
F1.3	The H2 subsystem shall be able to stop power production and notify Hub if one of the following conditions takes place: 1. Sufficient hydrogen is no longer available for efficient H2 production. 2. Fuel cell has been manually stopped.	**NP**	
F1.4	It shall at any time be possible to manually stop power production immediately and notify Hub in case of emergency or for maintenance/test purpose.	**NP**	
F1.5	The H2 subsystem shall be able to produce power when sufficient hydrogen can be delivered from hydrogen storage.	**NP**	
F1.6	The H2 subsystem shall be able to convert the output voltage from the given Fuel cell into the required input voltage for the power-bus.	**NP**	
F1.7	Delay between start and stop of power production as well as delay between stop and start of power production should be configurable via local configuration interface/web interface to H2 subsystem.	**NP**	
F1.8	The H2 subsystem shall monitor the input flow and input pressure to the Fuel cell to allow calculation of the Fuel cell efficiency	**NP**	
F1.9	The H2 subsystem should monitor the output current and output voltage delivered by the Fuel cell to allow calculation of the efficiency for the converter required between Fuel-cell and power-bus.	**NP**	
F1.10	The H2 subsystem shall be able to deliver a Green-score on request from Hub.	**NP**	

ID	WEB2 Requirements	Time box	
WEB1.1	The static H2 web site developed during WEB1 must be dynamic using PHP, SQL and Java scripts	3	
WEB1.1.1	The H2 web site should present sensor data stored in MySQL according to low fidelity prototype prepared during WEB1.	2	
WEB1.1.2	The H2 web site should include login and register functionality with Java script validation.	3	
WEB1.1.3	The H2 web site should include a notification service where registered users can be notified via email if a sensor data value has reached a user defined condition/threshold.	3	
WEB1.2	It should be possible to update the H2 database with sensor data from the H2 system.	4	
ID	**EMC1 Requirements**	**Time box**	
EMC1.1	The H2 system shall be tested for emission and comply to existing laws	5	
EMC1.2	The H2 system shall be tested for immunity and comply to existing laws	5	
EMC1.3	At least 2 EMC reviews must be performed by Per Lysgaard	2	
EMC1.3.1	The PCB board for relay-circuit must be reviewed by Per Lysgaard	1	
EMC1.3.2	The PCB boards for gain and level adjustments must be reviewed by Per Lysgaard	2	
ID	**IDE1 Requirements**	**Time box**	
IDE1.1	A Participatory session using low fidelity prototype shall be performed.	2	
IDE1.2	Video presentation for participatory session shall be prepared and presented for class.	3	
IDE1.3	Usability test of web interface	5	
ID	**EMB FPGA Requirements**	**Time box**	
EMB1.1	The H2 system shall include a FPGA that communicate with LPC2478 Development kit using the Wishbone interface.	5	
EMB1.1.1	The H2 system shall implement a Wishbone compliant A/D converter that can off-load the CPU	4	
EMB1.1.2	The Wishbone compliant A/D converter should support 8 channels	5	
EMB1.1.2	HW in terms of R2R ladder and comparators shall be used to support the A/D converter.	5	

TIME BOX 4 SPECIFICATION

Knud Baastrup

This time box covers the below requirements, which must be realized in order to meet the functionality specified for product version 4 in the deployment plan in Table 1.

ID	Requirements	Timebox	Signed
G1.2	The H2 subsystem shall be able to produce energy as electrical power using hydrogen fuel cell or any other device that can produce electrical power from hydrogen.	4	
G1.5.6	The H2 subsystem should notify the Web Server on changes in voltage and current level	4	
G1.6	The H2 subsystem shall be able to deliver current efficiency data to other systems on request.	4	
G1.6.1	The H2 subsystem shall be able to deliver latest efficiency data for electrolyzer on request.	4	
G1.7	The H2 subsystem should be able to deliver efficiency data measured over certain duration of time that at least should cover up to 24 hours.	4	
G1.7.1	The H2 subsystem should be able to deliver electrolyzer efficiency data measured over certain duration of time that at least should cover up to 24 hours.	4	
WEB1.2	It should be possible to update the H2 database with sensor data from the H2 system.	4	
EMB1.1.1	The H2 system shall implement a Wishbone compliant A/D converter that can off-load the CPU	4	

Development Plan

All

Timebox 4	Week 15							Week 16							Week 17						
Task	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S
Flow Sensor circuit (Final version)																	D	D			
Current & Voltage Sensor (Final version)		K		K	K																
Gain & Level (Final version)			K																		
Solve issue with ADC pin configuration on LPC2478																	D	D			
Identify suitable cabinet (EMC/Safety)																	L	L			
Draw cabinet with PCB and cables (EMC/safety)																	L	L			
Thread for continous update of sensordata in H2 App.											L	L							L	L	
Average filter to smoothen sensordata in H2 App.												L	L								
Preparation for usability test																				L	
Complete end-to-end integration test (with Electrolyzer)																		A	A		
Update webserver with sensordata via socket in H2 App.						K	K			K				K							
Support the CAN driver development community																					
CAN protocol support in H2 System Interface																					A
FPGA Module Design Presentation	K																				
Create Wishbone Slave compliant A/D Converter									K	K									K	K	K
Extend FPGA based A/D Converter to support 8 channels																K	K				
HW mock-up to support FPGA based A/D Converter																	D	D			

L: Lasse Lykkegaard K: Knud Baastrup : Easter vacation

D: Dennis Thomsen A: All

Verification Plan

Knud Baastrup

We have recognized that the Product Acceptance test, prepared during the Launch project, lacks support for early deployment and we believe it is better to develop a set of functionality tests that can be carried out along with the customer (or a customer representative) for each timebox deployment. The Product Acceptance test from Launch can still, if required, be executed as part of a final product deployment towards the end of the project.

Test cases to verify the timebox 4 provided functionality will be developed in parallel with the implementation and the test results will be documented.

EMERGENCY STOP

Lasse Lykkegaard

In the regulative of machinery EN 60204-1[1], and in other regulations of emergency stops it is said that emergency stop may not be controlled by electronic equipment. However special designed and tested emergency PLCs have been granted the possibility to control such systems.

Our system is not allowed to do so, it is supposed to be a separate system with a dual connection of doubled forced out breakers and a doubled main breaker, which shuts off dangerous elements. This main could notice our control system.

We see some relevance for the end-costumer in implementing such a system, but little relevance for us considered the extra expenses, big contactors and emergency breakers are.

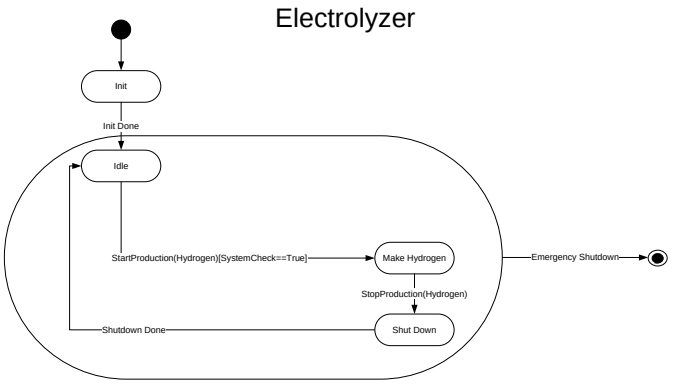
Therefore we've chosen to control our system with software, since all converter's main passes a relay controlled by the system. This relay will do it for the emergency shutdown.

State Machine

Lasse Lykkegaard

As for now nothing is implemented as state machine – there are various reasons for this approach.

An implementation with state machines are not required or described in the requirements. But the functionality described in the state machines must be preserved.



[1] http://www.ds.dk – Dansk Standard

EMC FOR BOX DESIGN

Lasse Lykkegaard

In order to ensure low emission and immunity towards electrical interruption a cabinet with sufficient EMC specifications should enclose the different PCB's. Profitability issues and design factors often eradicates this solution. In many situations a EMC enclosure is over-kill and not needed. A emission and Immunity-test, falsifies or verifies the need for EMC safe enclosure

Rittal A/S for example makes suitable EMC cabinets with EMC gaskets. Shielded cables are led into the box through EMC glands, and filters are placed best in the glands or just inside to avoid emission from "unfiltered" wires. Seldom all of this is done; it is not feasible or necessary.

The cabinet and glands protects noise from entering through air. Much noise enters superimposed on cables needed for the system to function.

Some form of filtering is needed when entering the box. If noise is not removed instant all wires acts as antennas and spread more noise.

Single point (side) entry guides all shield noise traveling from one cable to another around without disturbing the components inside.

A rough sketch of how the different PCBs are placed is shown in Figure 1.

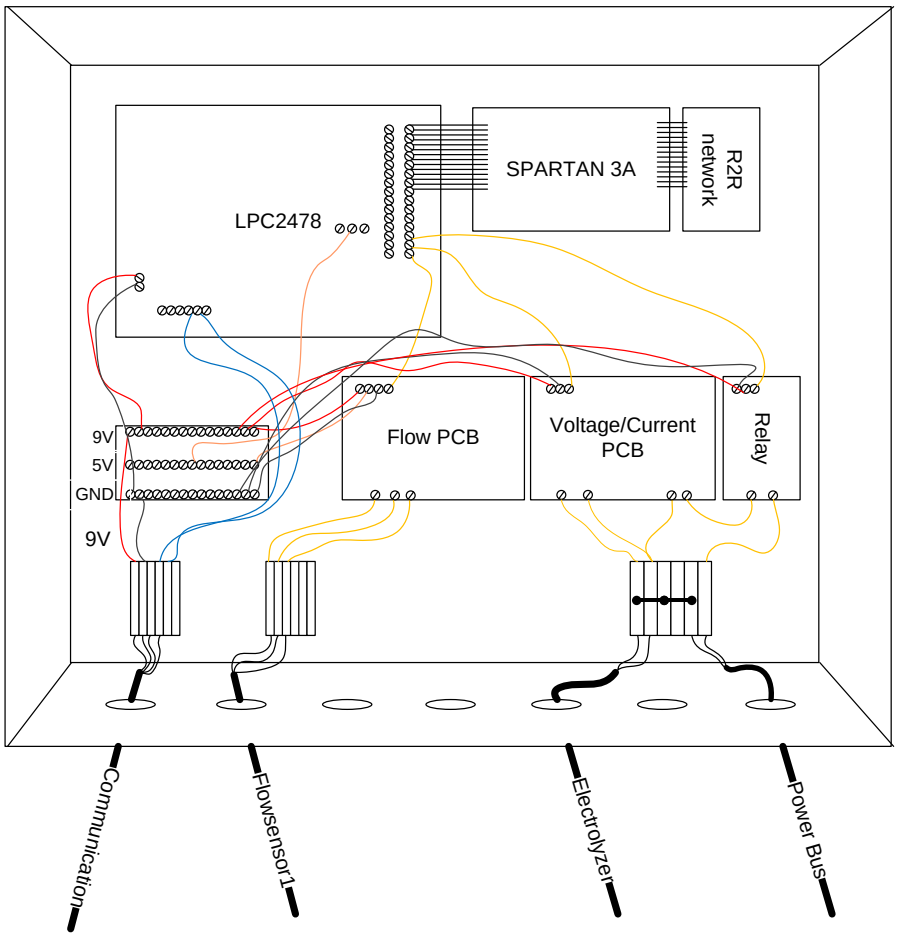
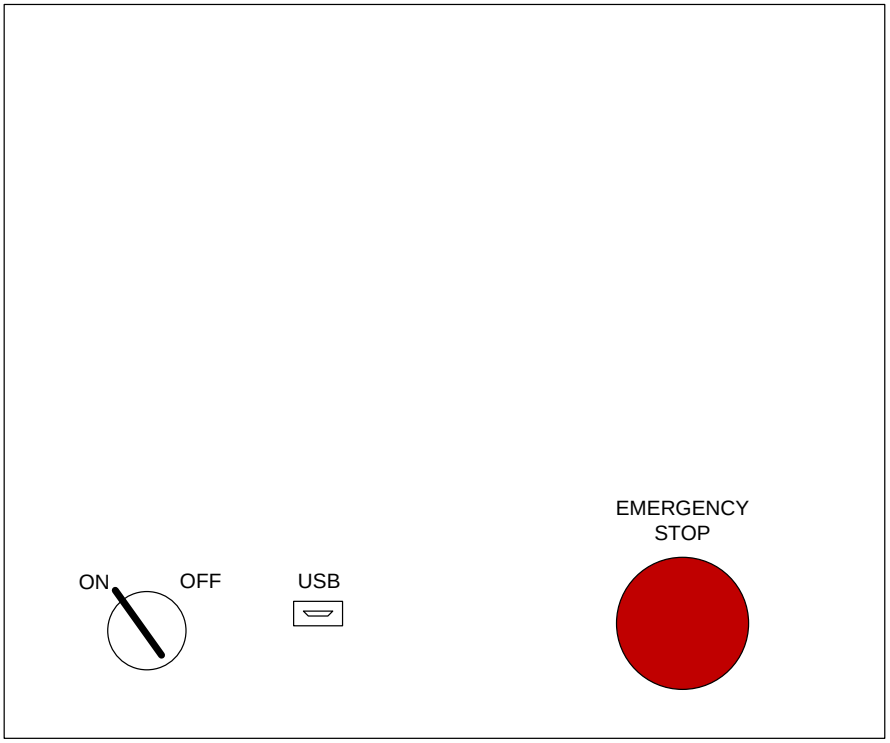


Figure 1 – Box layout



Figur 2: Lid for the EMC box

The original idea of having a transparent plexi glass area in the front of the box was not a good idea regarding EMC. To eradicate noise components applied from radiation, an EMC-net must be mounted across the hole. This though is barely transparent and you would barely be able to see through.

SW DESIGN

All

The software updates include the possibility to update sensor data into a MySQL server DB, which allow the data to be presented on the H2 Webpage.

The program is expanded with threads to maintain sensor data and an averaging filter to smoothen out flicker.

Update of sensor data to database

Knud Baastrup

The H2 system has been updated with the capability to update the sensor data in an external MySQL database when sensor data are requested via the already existing getSensordata function. The update of sensor data into the MySQL database will, however, only be performed if there is an interface toward the MySQL database. This interface can be created using a new createDbRelayIf function.

The interface towards a MySQL server can be quite cumbersome to build up from scratch, so a better solution is to either interface a stand-alone application like Connector/C or by using an API like the MySQL C API library.

It was not succeeded to find a Connector/C client program or a MySQL C API library prepared for uCLinux that is used on the LPC2478 Development kit. The source code is however available and the porting could be straight forward, but just as well cumbersome and very time consuming.

It was decided to create a separate application to manage the SQL updates, which could run on a more common Linux distribution like bren.hih.au.dk, our web server lene-lass.dk or a Windows/Linux laptop as the first step and then later port this functionality to UcLinux once the desired functionality were up running.

Figure 3 show how the H2 application performs its SQL updates via a separate SQL relay application running on a Linux server or just a windows laptop.

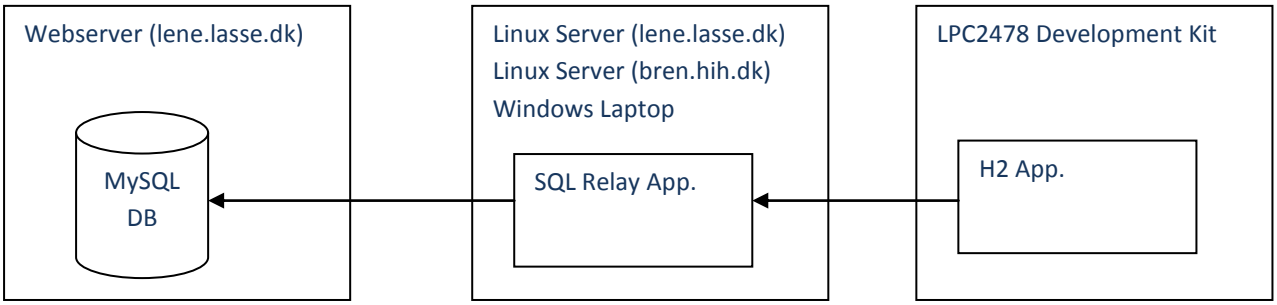


Figure 3: SQL updates from H2 application performed via SQL relay application on separate server.

SQL Relay Application

It was chosen to use the MySQL C API library in favor of the Connector/C to get a more self contained application. The Connector/C is also utilizing the MySQL C API library so either solution should not give any noticeable difference in performance.

The SQL relay application will listen on an agreed UDP port for any UDP datagrams sent from the H2 application. Once a valid datagram is received the data will be tokenized and build into a SQL Insert statement, which is forwarded to the MySQL server utilizing the library functions provided along with the MySQL C API.

The source code is available via doxygen generated html on the following link:

http://lene-lasse.dk/doxygen/h2_app_sql_relay/index.html

A separate application has been prepared for Windows where a few changes were required for proper use of the Windows sockets.

H2 Application updates

The H2 application has been extended with a public function that allows an interface to be created towards the SQL Relay application. The interface will ensure that a UDP socket is initialized and prepared for sending UPD datagrams towards the server hosting the SQL relay application.

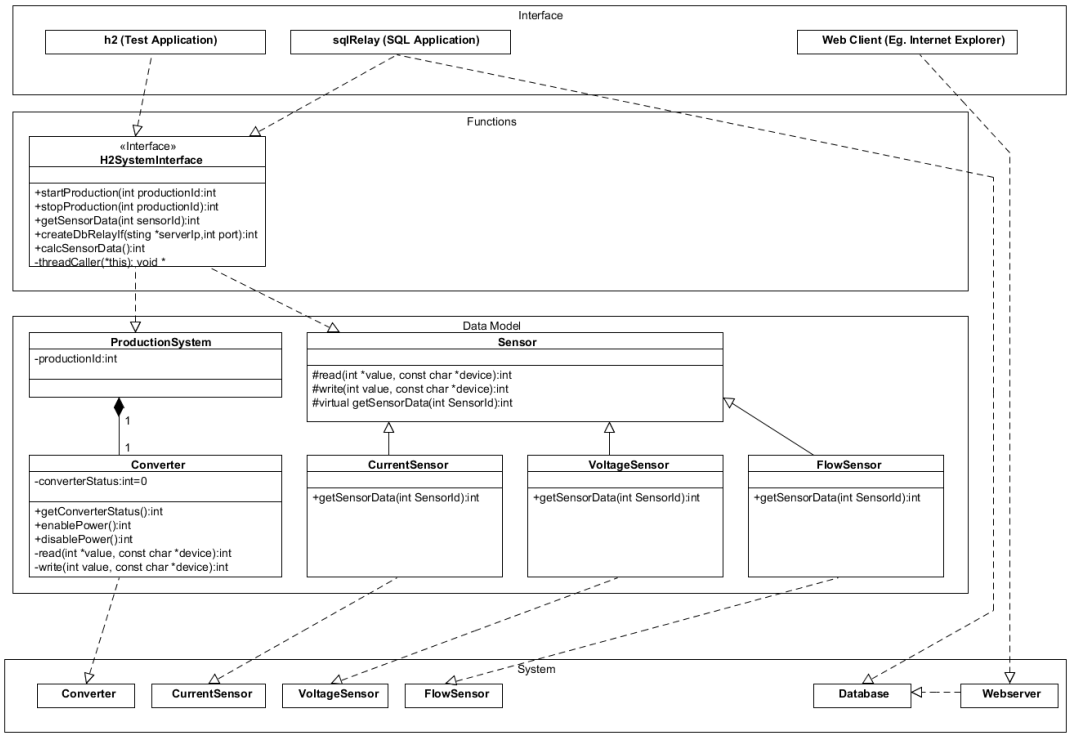
Once the interface is created, all follwing requests for getSensordata will result in an attempt to update the MySQL database.

The source code is available via doxygen generated html on the following link:

http://lene-lasse.dk/doxygen/h2_app/index.html

Class Diagram

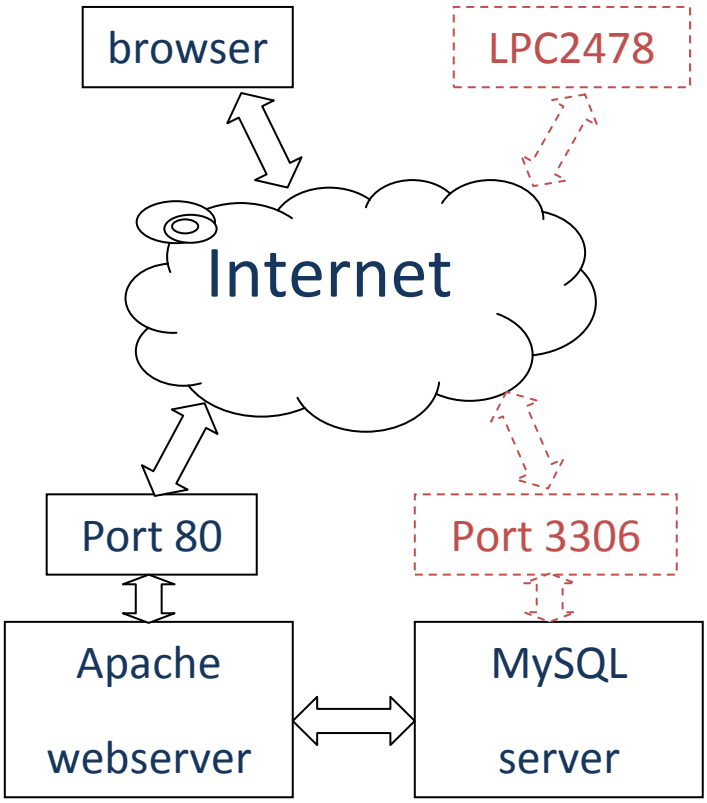
The class diagram has been updated with one additional function in the H2SystemInterface that allow sensor data to be updated into a MySQL database via a SQL relay application running on a separate server. The function is called createDbRelayIf and takes the IP and port of the server hosting the SQL relay application as argument.



MySQL server

Lasse Lykkegaard

Our server-solution is configured as a combined webserver and MySQL server system, which is very common for small budget systems, placed on an Ubuntu server. Normally it's designed to work the blue way, but we're trying to establish the red path to update our data. It required some workaround to gain access to the MySQL-server which were configured to listen to the right port but only calls from localhost directed to the server.



Continuously sensor sampling

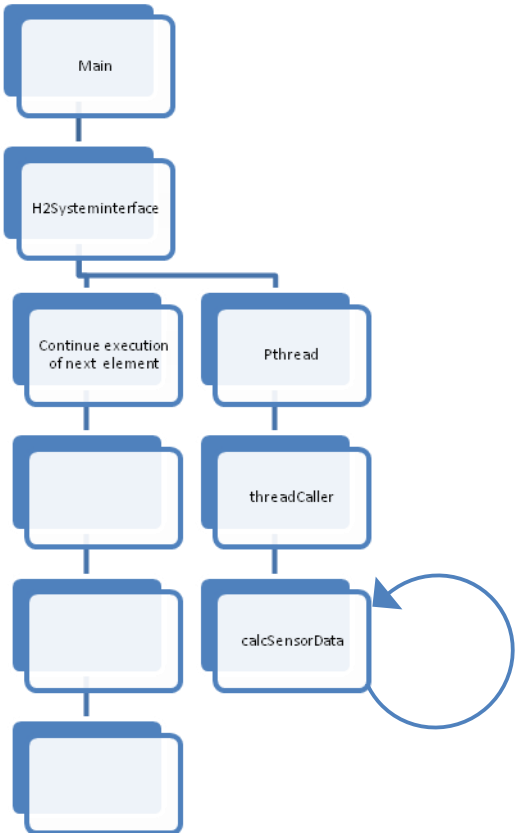
Lasse Lykkegaard

When a getSensorData() call is performed the system must go down through all classes to fetch the sensor data at that moment., multiple calls to kernel space, could eventually make our sampling rate very fast – if that speed exceeds the sensor sampling speed, system will hang and become slower. If the data is continuously sampled to a buffer, getSensorData()-users will be served much fast, but the data will be older, and perhaps less accurate. Accuracy of microseconds is not an issue in our case.

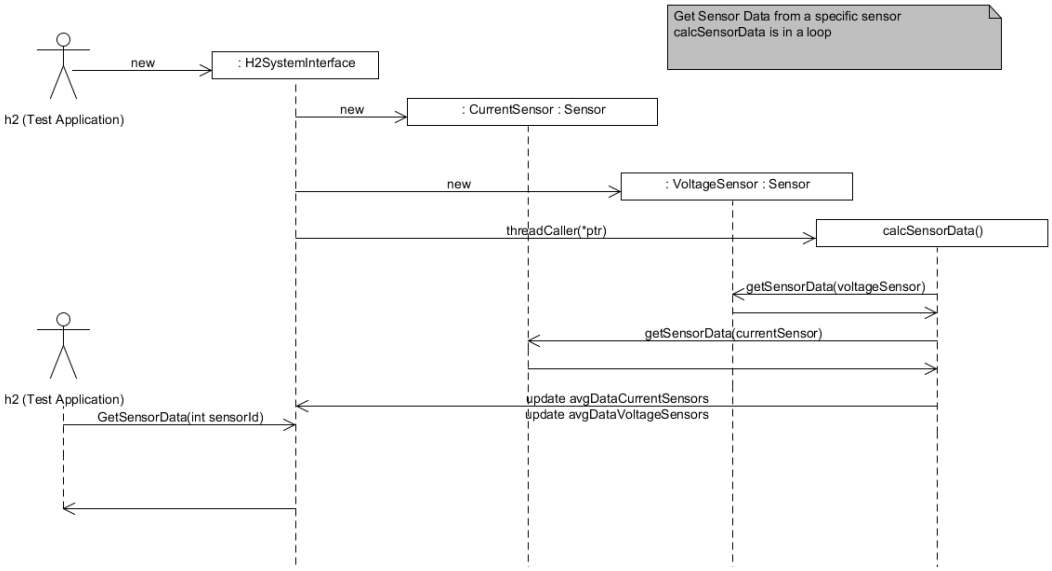
When the h2SystemInterface-object is constructed a POSSIX Thread is constructed calling a host function (threadCaller) of the h2SystemInterface-class with the current object-pointer (this) as parameter.

The host function creates another object pointing to the originally object. Now we're having two objects with the same variable location. Now this function eventually calls the calcSensorData() which are put in while(1) loop and updates the variables.

All this work around is needed because the host function needs to be static and we're making a callback to the original objects data. A static member function can be called, even when a class is not instantiated. Static functions cannot access (this) since it is not called from an object, therefore (this) are parsed along as a parameter.



Alongside this implementation, an alternation of the h2-main program is needed.
Previously the program terminated after each run – a termination of the program will also terminate the thread and all memory allocated. The program is now put into a loop – asking for instructions.



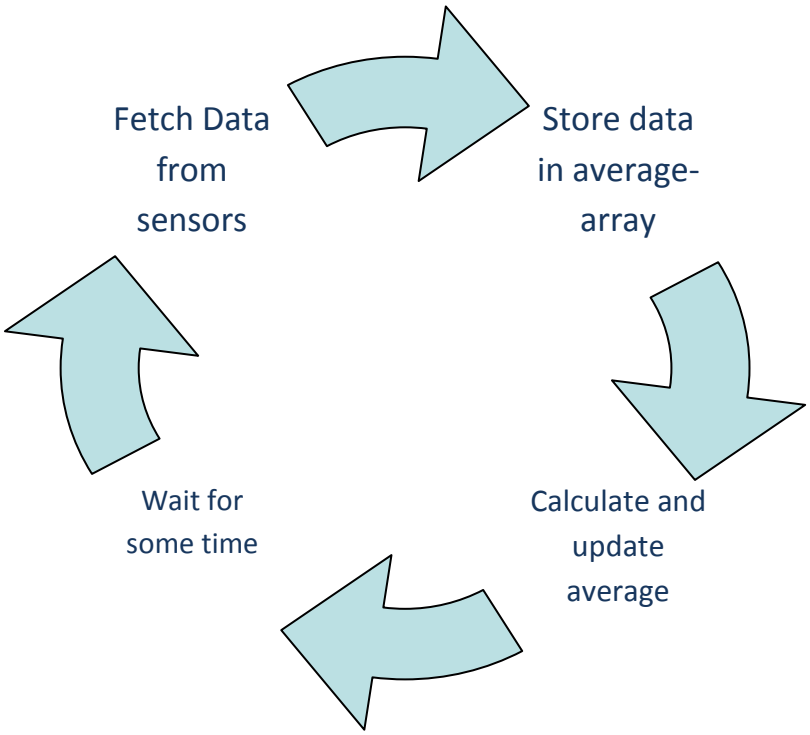
Averaging

Data from sensors through ADC without filters can be noisy. If a sampling is made when a single very large transient occurs and that sample is used to calculate, for example energy, large error I made. Therefore an averaging "filter" is implemented. A number of samples is stored in a buffer, and then the average is returned. This smoothens out flicker, but delays changes. This in fact is a digital lowpass filter.

This is DC mean averaging, what if the signal was AC, and a sine function – you could risk the average being zero. Think of the signal being sampled exactly at a ripple peak every time, keeping the average at an artificial high level.

Quadratic mean or RMS (root mean square) is often a "averageing " way used in sensors and measurement tools, ensuring a valid signal for both DC and AC signals. More calculations and memory is needed for this average.

What our system suddenly reports the voltage to be unrealistic low or negative – the reason for this readings is perhaps a undisclosed interrupts in the cpu or ESD. This one failed measurement is used in the calculation for as many sample chosen to average over. Consider such a sample for faulty and skip it would improve the filtering even more.



HW DESIGN

Flow/pressure circuit design

Dennis Thomsen

Electrolyzer flow circuit

Problem getting output under 3.3 V when adjusting on the potentiometer but the as maximum output can go as high as 5.7V, this can be fixed changing the voltage divider it will make for better adjustment options, as value will change on the print due the error percentage of the components.

The good thing about this is that the voltage divider wouldn't have to be that accurate as before, as the output will be adjusted in, also will allow change the 4.2 MΩ potentiometer to a lower value, as it was change to that high a value to try and solve the problem with getting output to be able to lie at 3.3 V previously.

Voltage divider for 4 to 2.9V

$R2 := R2 \quad R1 := R1$

$v_o := 2.9V \qquad v_{in} := 4V$

$R1 := 1\Omega$

$R2 := v_o = \frac{R2}{R1 + R2} \cdot v_{in} \text{ solve}, R2 \rightarrow 2.6363636363636363636\Omega$

$E12 := \begin{pmatrix} 1 \\ 1.2 \\ 1.5 \\ 1.8 \\ 2.2 \\ 2.7 \\ 3.3 \\ 3.9 \\ 4.7 \\ 5.6 \\ 6.8 \\ 8.2 \end{pmatrix} \quad R1 := E12 \quad R2 \cdot E12 =$

	0
0	2.636
1	3.164
2	3.955
3	4.745
4	5.8
5	7.118
6	8.7
7	10.282
8	12.391
9	14.764
10	17.927
11	21.618

$\cdot \Omega \quad R2 := \begin{pmatrix} 2.7 \\ 3.3 \\ 3.9 \\ 4.7 \\ 5.6 \\ 6.8 \\ 8.2 \\ 10 \\ 12 \\ 15 \\ 18 \\ 22 \end{pmatrix}$

$$v_o := \frac{R2}{R1 + R2} \cdot v_{in}$$

$v_o =$

	0
0	2.919
1	2.933
2	2.889
3	2.892
4	2.872
5	2.863
6	2.852
7	2.878
8	2.874
9	2.913
10	2.903
11	2.914

$\cdot V$

Picked the 10'th output so $R1 := 68k\Omega$ $R2 := 180k\Omega$

$$v_o := \frac{R2}{R1 + R2} \cdot v_{in} \qquad v_o = 2.903 \cdot V$$

$$I_1 := \frac{(v_{in} - v_o)}{R1} = 0.016 \cdot mA \qquad I_2 := \frac{v_o}{R2} = 0.016 \cdot mA$$

Some quick calculations in order to find what value the potentiometer should be set at to get 3.3V out of the system

$R1 := 68k\Omega$ $R2 := 180k\Omega$ $v_o := 3.3V$

$$v_{in} := v_o = \frac{R2}{R1 + R2} \cdot v_{in} \text{ solve}, v_{in} \rightarrow 4.5466666666666666667V$$

$$Gain := \frac{v_{in}}{4V} = 1.137$$

$R_1 := 2.2k\Omega$ $R_2 := 100k\Omega$ $R_3 := 100k\Omega$

$$Gain = \left(1 + \frac{2R_1}{R_{Gain}}\right) \cdot \frac{R_3}{R_2} \text{ solve}, R_{Gain} \rightarrow 32.195121951219504343k\Omega$$

That here is what we ended up using: 100k potentiometer which on the board gave a range from 3.013V to 4.978V

Inserted voltage follower, for EMC reasons, so as not to have 2 high impedance system connected by using a buffer the output becomes low impedances, doing this also fixed a problem that caused the voltage of the signal sent to the ADC inputs channels on the Spartan board, to be raised with about 0.5V when it was connected to the board.

Another EMC concern with the board was how it should be connected to the sensor obviously the output from the sensor would go to the board, but the power and ground need could either come from the board or it could just be hooked up the same supply as the board is connected to, see illustration below

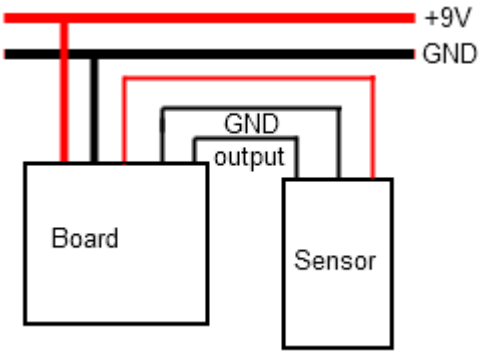


Figure 4 – Correct way to connect

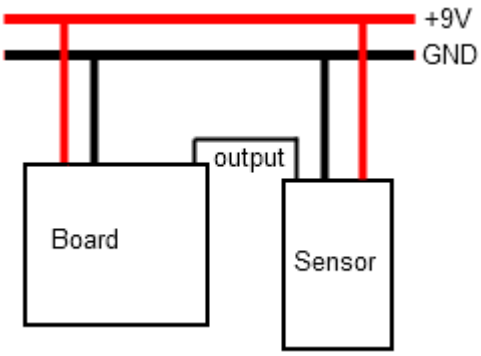
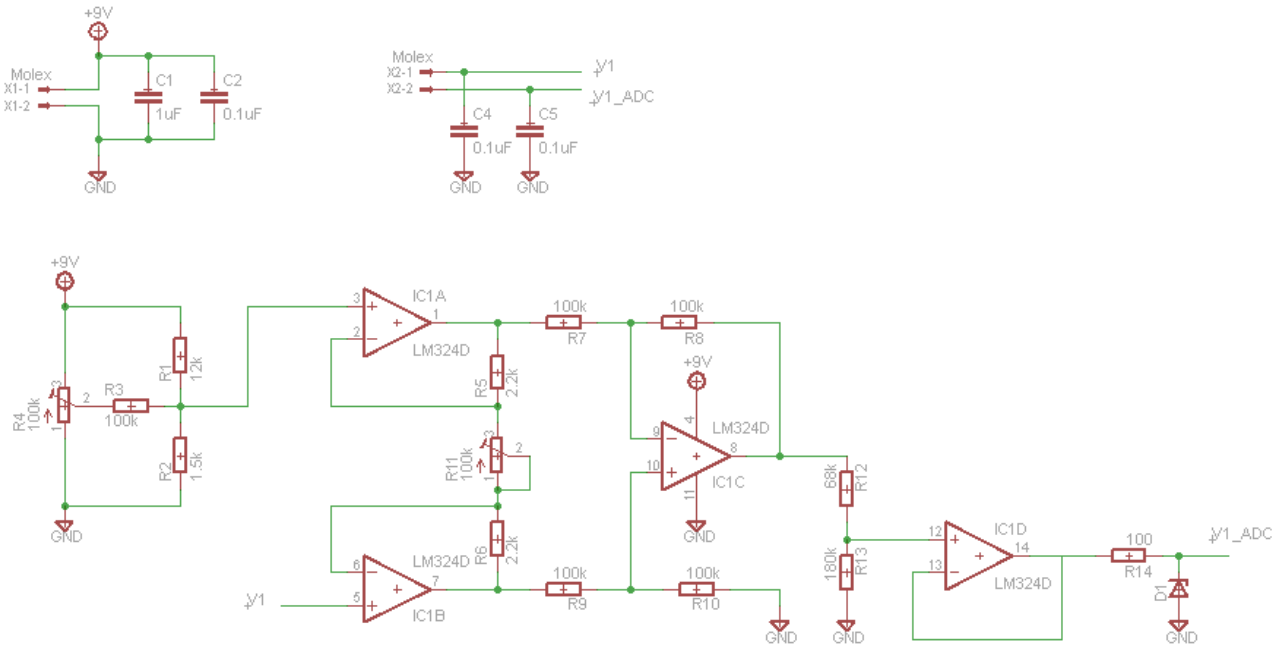


Figure 5 – Okay way to connect

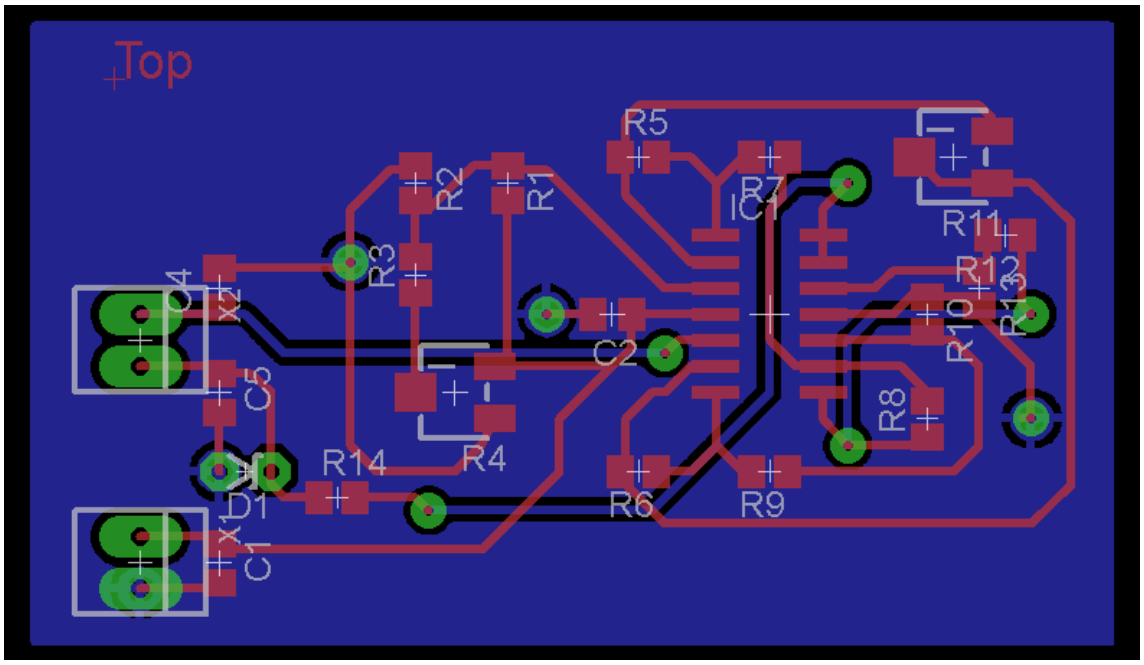
The way shown in Figure 4 would be the most correct way to connect the two, so as to follow a star topology, and because if the sensor was dragging a large ampere it could otherwise lower the voltage level of the ground, but as the sensor only use a few mA the method shown in Figure 5 will do just fine.

Additionally a Zener diode with a break voltage of 3.3V have been added to ensure that the output don't exceed 3.3 V and burn out the ADC inputs on the Spartan board. The resister connected in series to the diode have been chosen to have a low resistance of 100Ω, in order to keep the low impedance output.

Design is awaiting final EMC approval



Figur 6 – Schematic for Electrolyzer flow sensor board



Figur 7 – PCB layout for Electrolyzer flow sensor board

Generator (Fuel Cell) flow circuit

The flow sensor for the generator needs the following supply voltage

Vcc: 10.8-26.4 V DC

As our system get a 9V supply to run the circuits and sensors on, this means that in order to run it either a DC to DC voltage doubler have to be build for the sensor or batteries can be used, as batteries is by far the fastest option they will do for now, something more permanent could be build if the time allows it.

The Following table from the datasheet shows the connection between the flow and the voltage output

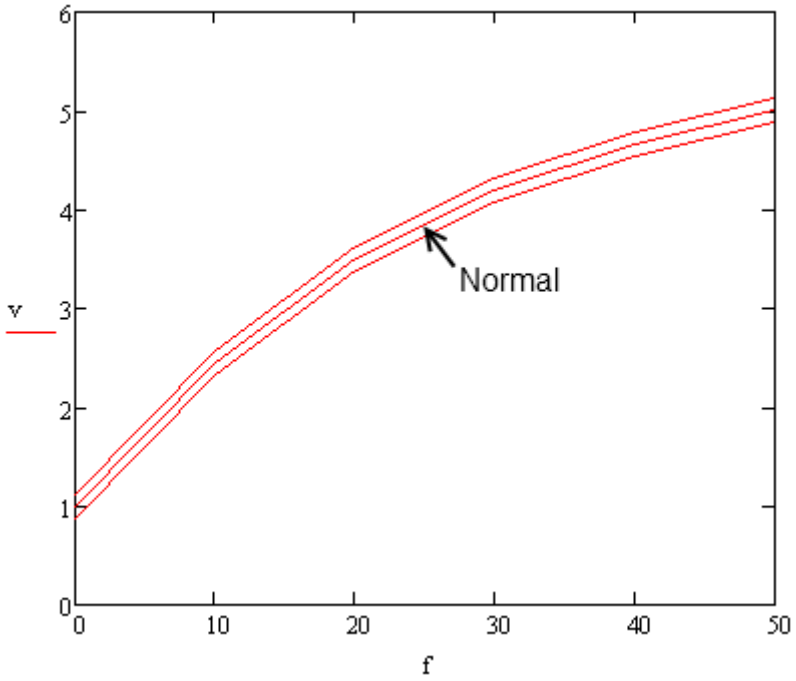
D6F50A5000

Flow Rate L/min (normal)	0	10	20	30	40	50
Output Voltage (VDC)	1.00 ±0.12	2.45 ±0.12	3.51 ±0.12	4.20 ±0.12	4.66 ±0.12	5.00 ±0.12

The output goes from 1V to 5V just as the output from the sensor used for the electrolyzer meaning that the same circuit can be used for this sensor as well.

However there is one difference, which becomes clear if the content of the table is inserted into a graph.

The middle line is the normal voltage level for the flow, while the other lines are the maximum error deviation from the norm.



Figur 8 – output vs. Flow for D6F50A5000 flow sensor

As can clearly be seen this is not a linear output as the other sensor where, however this is not a problem for using the same circuit, as it is far easier to solve in coding rather than make circuit to linearise the output.

The best way to do this is either to make a huge lookup table for the program to use when converting from voltage to flow, or get a formula describing the graph.

Generator pressure circuit

To measure the pressure the sensor MPX4250AP is used, from its datasheet it has the following characteristic

Vcc: 4.85 - 5.35 V DC Normal: 5.1 V

	Typ	Max	Units
Supply Current (I_O)	7.0	10	mAdc

To power it, the +5V power supply from the Spartan board will be used, as it is far faster than making even a simple voltage divider to cut the 9V supply down to 5V.

Minimum Output Voltage: 0.133 - 0.264V DC Normal: 0.204 V
Maximum Output Voltage: 4.826 - 4.966 V DC Normal: 4.896 V

Graph from datasheet showing the correlation between the output and the pressure

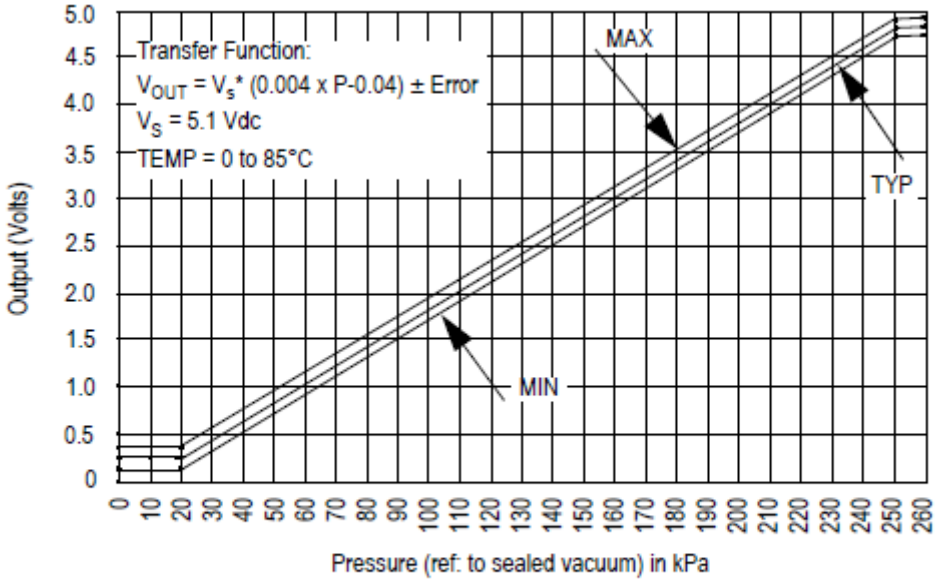
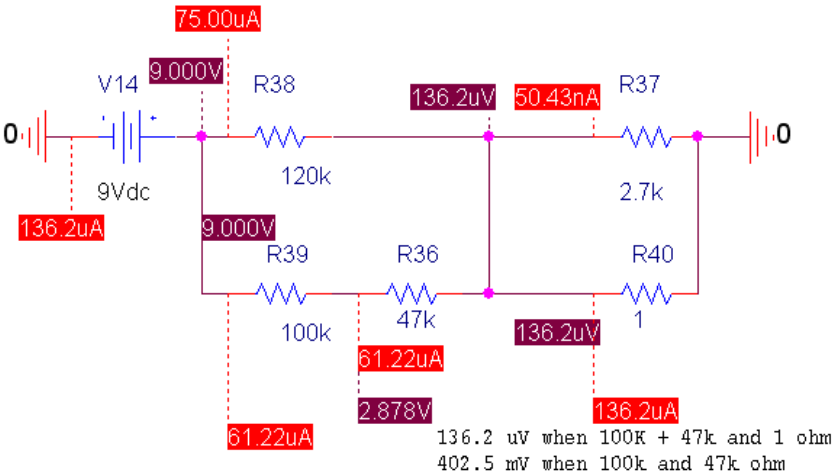


Figure 9 – Output vs. Absolute Pressure

As can be seen this is a linear correlation, making it easy to scale it in the code, and the gain and level circuit can be used almost as it is with just a few mirror corrections and recalculations.

Going out from the normal value to make the voltage divider for the offset adjustment

Having a resistor connected to the middle pin of the potentiometer, allow for small adjustments of the offset, too small for what is needed here, but by changing the position of the resistor, it can permanently limited the maximum voltage out of the divider.



Testing on the bread board gave Range: 0.004V - 0.408V

This is sufficient for the needs at hand.

Going out from the normal value for the calculations, and what was learned from the flow board to the electrolyser, what is needed is a voltage divider that drops the voltage to about 2.9 - 3.0

Instrumentation amplifier Voltage divider 4.896V to 2.9V

$v_o := 2.9V \quad v_{in} := 4.896V - 0.2V$ subtract the offset from the input value

$R1 := 1\Omega$

$R2 := v_o = \frac{R2}{R1 + R2} \cdot v_{in}\ \text{solve}, R2 \rightarrow 1.6146993318485523385\Omega = 1.6146993318485523385\Omega$

$E12 := \begin{pmatrix} 1 \\ 1.2 \\ 1.5 \\ 1.8 \\ 2.2 \\ 2.7 \\ 3.3 \\ 3.9 \\ 4.7 \\ 5.6 \\ 6.8 \\ 8.2 \end{pmatrix} \quad R1 := E12 \quad R2 \cdot E12 =$

	0
0	1.615
1	1.938
2	2.422
3	2.906
4	3.552
5	4.36
6	5.329
7	6.297
8	7.589
9	9.042
10	10.98
11	13.241

$\cdot \Omega$

Going with the last combo and the closest E12 value

$R_1 := 82k\Omega \quad R_2 := 150k\Omega$

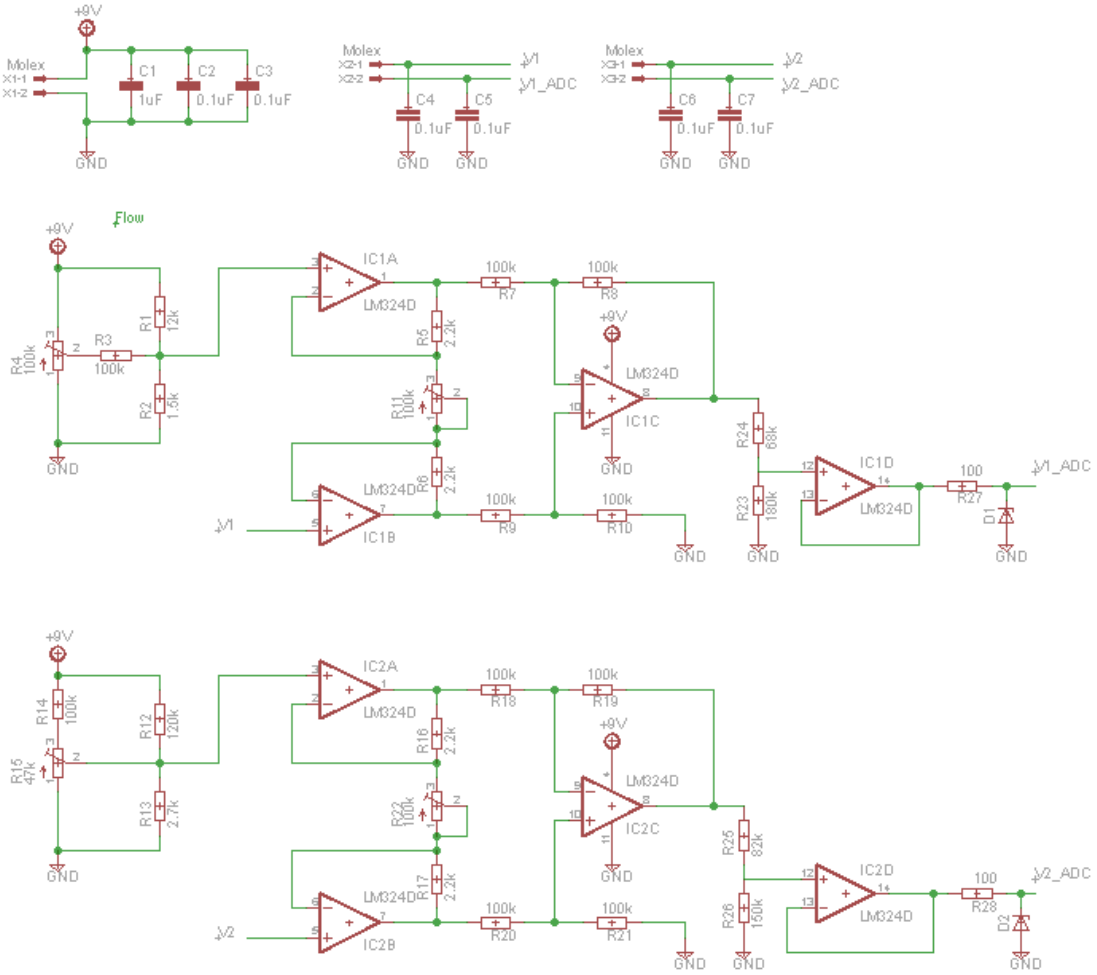
$v_o := \frac{R_2}{R_1 + R_2} \cdot v_{in} \quad v_o = 3.036 \cdot V$

$I_1 := \frac{(v_{in} - v_o)}{R_1} = 0.02 \cdot mA$

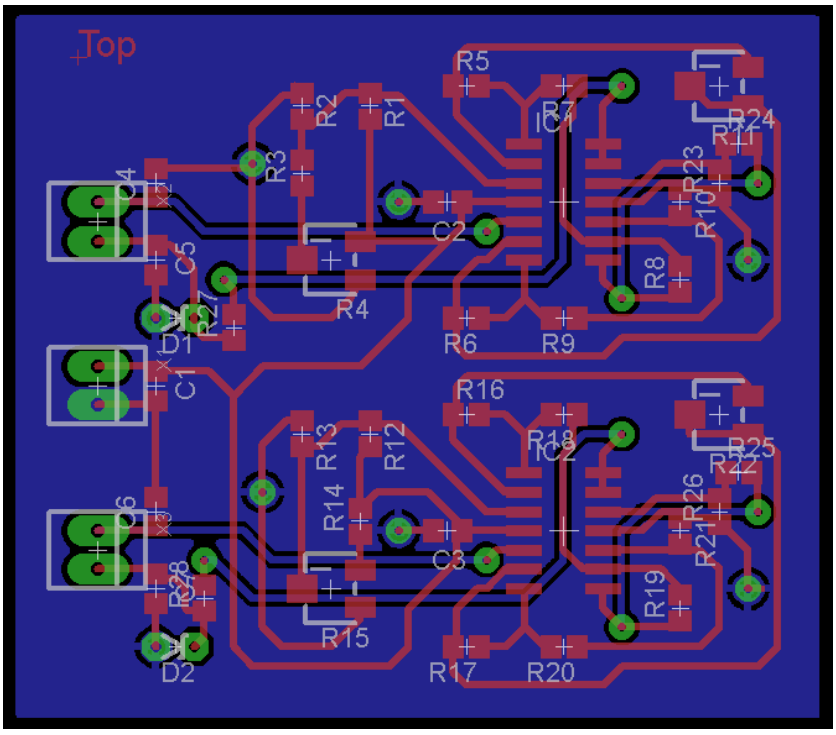
$I_2 := \frac{v_o}{R_2} = 0.02 \cdot mA$

With 82k and 150k as a voltage divider, the output range is then 2.845 V – 4.4V, with a 100 potentiometer.

Design is awaiting final EMC approval



Figur 10 - Schematic for Full cell flow, and pressure sensor board



Figur 11 – PCB layout for Full cell flow, and pressure sensor board

FPGA Module Design for 8-ch ADC

Knud Baastrup

The H2 system will include an option to use a FPGA for the A/D conversion that is currently done by the LPC2478 CPU for each of the analog sensors. The usage of a FPGA will reduce the load on the CPU and at the same time allow us to create a Wishbone compliant slave as required by requirement EMB1.1.

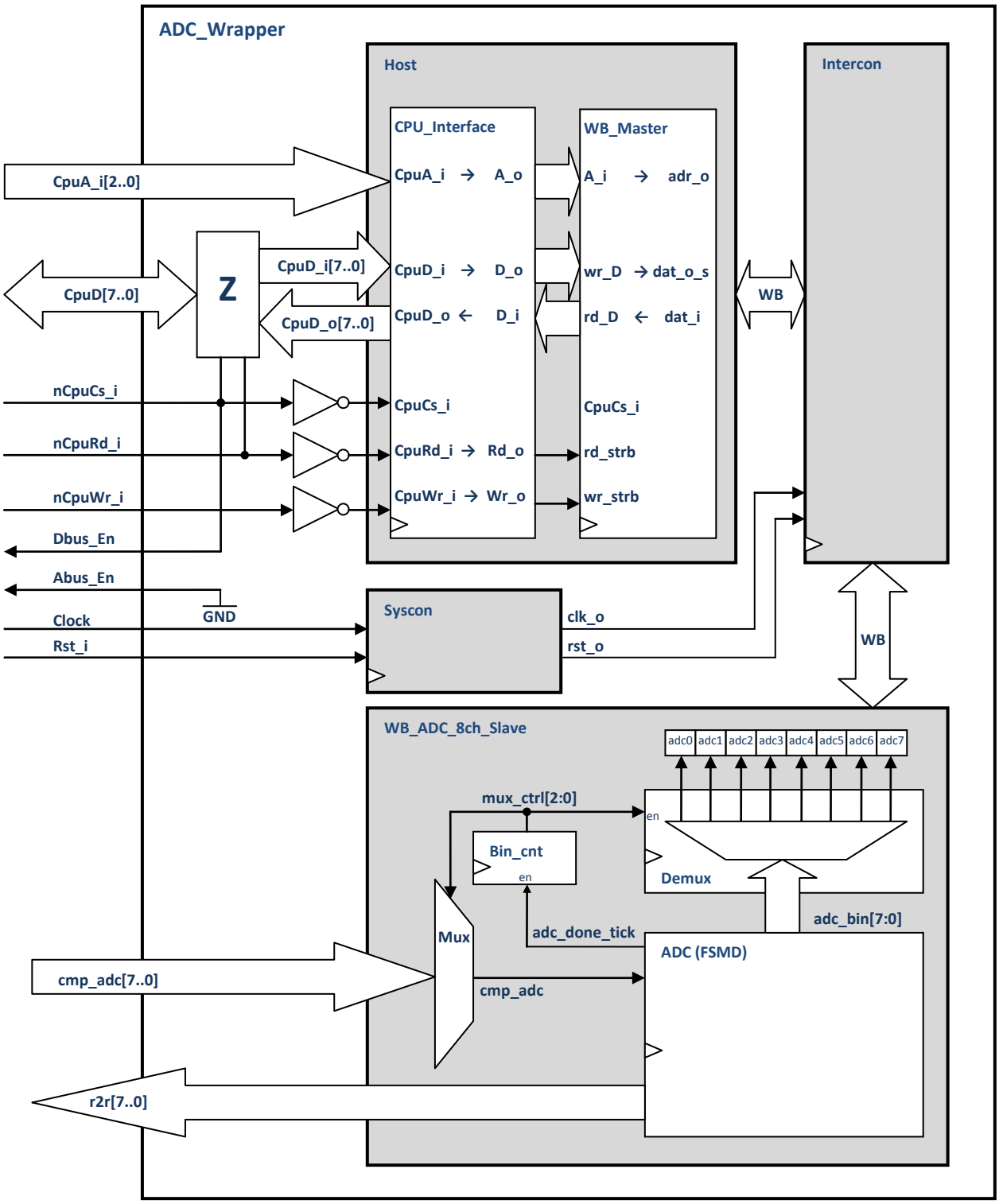


Figure 12: FPGA module design for 8 channels A/D converter

Figure 12 shows the FPGA Module design for an 8 channel A/D converter. The ADC_Wrapper module instantiate an entity of the Host, Intercon, Syscon and WB_ADC_8ch_Slave modules where the WB_ADC_8ch_Slave is the Wishbone slave that is in charge of the A/D conversion.

Host

The Host module instantiates a CPU_Interface entity that synchronizes the control signals from the LPC2478 CPU to the clock domain of the FPGA, and as well a WB_Master entity that manages the Wishbone bus cycles.

The synchronization performed by the CPU_Interface should ensure that Wr_o goes high on the rising edge of nCpuWr_i. The number of CPU clocks that nCpuWr_i stays low is configured via the EMCStaticWaitWr2 register of the LPC2478 CPU.

The desired tick for Wr_o is implemented using a edge detector on the falling edge of CpuWr_i (falling edge because nCpuWr is inverted by the ADC_Wrapper entity before it enters the CPU_Interface entity).

The WB_Master module is copied from code examples provided by Morten Opprud.

Intercon

The Intercon entity inter-connects the Wishbone Master with the Wishbone Slaves.

The Intercon module is copied from code examples provided by Morten Opprud.

Syscon

The Syscon generates the system clock and reset signals using a DCM

The Syscon module is copied from code examples provided by Morten Opprud.

WB_ADC_8ch_Slave

The WB_ADC_8ch_Slave module instantiates the ADC entity that perform the actual A/D conversion and a Mux, Demux and Bin_cnt entity that allow the ADC entity to support up to 8 channels.

The output from the ADC entity is a binary bit pattern (adc_bin) that assigned to an external R-2R ladder (via r2r bus) will produce a voltage that match the analog input voltage.

The binary bit pattern, to represent the analog voltage, can be found using the Successive Approximation (SAR) algorithm along with supporting HW in terms of an R-2R ladder and a comparator as depicted in Figure 13. In Figure 13 the r2r bus is connected to ADC[0..7] and the comparator result is given by CMP_ADC_1.

We start by assigning the r2r bus the bit pattern 10000000_b, corresponding to 0.5 * V_{ref}, and check if the comparator goes high or low. If the comparator e.g. goes high, we know that a higher voltage is required on the inverting terminal to match the analog value on the non-inverting terminal, so we try next with the bit pattern 11000000_b, corresponding to (0.5 + 0.25) * V_{ref} . If the comparator this time goes low, we know that a lower voltage is required, so we try with the bit pattern 10100000_b, corresponding to (0.5 + 0.125) * $V_{ref.}$. Continuing this approach the SAR algorithm will allow us to find the best match within 8 steps corresponding to the 8 bit supported by the external R-2R ladder.

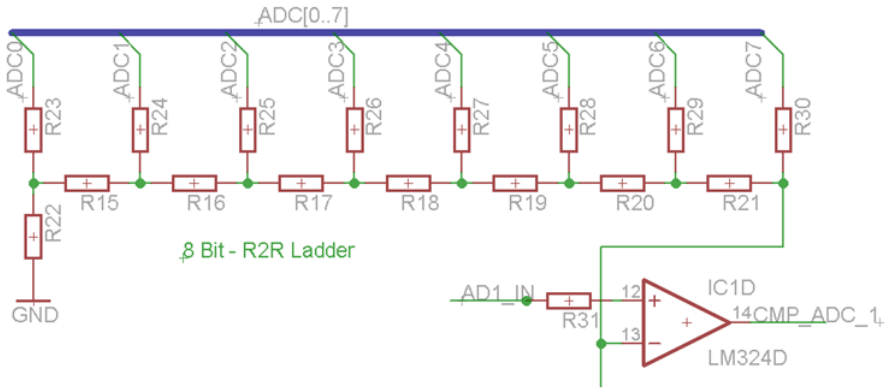


Figure 13: HW to support the ADC entity.

We can quite easy extend the ADC entity to support 8 channels by adding 7 additional comparators in the supporting HW along with a multiplexer, de-multiplexer and binary counter that can be implemented in the FPGA.

The Bin_cnt entity will count to 7 and increase by one each time an ADC conversion has been completed. The output from Bin_cnt will be used to control a Mux entity that combines the 8 comparator outputs and as well a Demux entity that split the adc_bin into 8 “registers” that can preserve its binary code.

The 8 “registers” (adc0 to adc7) will preserve the latest binary pattern and be available for read-out on request via a Wishbone read cycle.

ADC_Wrapper

The ADC_Wrapper defines the external interface and instantiates as earlier mentioned all the Wishbone components as depicted in Figure 12. The external interfaces, in terms of signals, are via user constraints mapped to physical pins.

We will implement the FPGA solution on the Spartan 3A that we used during 2nd Semester even though it only provide easy access to 35 FPGA pins (via the J4 connector) which put some limitations to the number of pins we can afford for data and addressing.

We can limit the databus to 8 pins, which is sufficient to transfer one adc register and we can as well limit the bus for addressing to 3 pins, which is sufficient for the addressing of the 8 adc registers. It seems possible to use 4 pins for addressing, which allow us to keep 8 addresses free for e.g. an additional Wishbone slave entity in the future.

Table 3 shows how the J4 connector pins from the Spartan 3A can be connected towards pins on the LPC2478 Development kit and as well the R-2R supporting HW.

Table 3: Spartan 3A connections towards LPC2478 and R-2R HW

Spartan 3A			LPC2478 OEM		R-2R HW	
Name	Signal	Pin J4	Signal	Pin	Signal	Pin
	GND	1			GND	
	5v	2			5v	
	3v3	3				
C4		4				
A14	CpuD<0>	5	D0	J2-60		
B14	CpuD<1>	6	D1	J2-59		
A13	CpuD<2>	7	D2	J2-58		
D13	CpuD<3>	8	D3	J2-57		
C13	CpuD<4>	9	D4	J2-56		
C12	CpuD<5>	10	D5	J2-55		
A12	CpuD<6>	11	D6	J2-54		
D11	CpuD<7>	12	D7	J2-53		
B12	r2r<0>	13			ADC[0]	J1-0
C11	r2r<1>	14			ADC[1]	J1-1
A11	r2r<2>	15			ADC[2]	J1-2
D10	r2r<3>	16			ADC[3]	J1-3
A10	r2r<4>	17			ADC[4]	J1-4
E10	r2r<5>	18			ADC[5]	J1-5
A9	r2r<6>	19			ADC[6]	J1-6
D9	r2r<7>	20			ADC[7]	J1-7
C9	Abus_En	21	ABUS_EN	J2-44		
C8	Dbus_En	22	DBUS_EN	J2-43		
A8	nCpuWr_i	23	WE	J1-35		
E7	CpuA_i<0>	24	A1	J2-41		
B8	CpuA_i<1>	25	A2	J2-40		
D8	CpuA_i<2>	26	A3	J2-39		
A7	CpuA_i<3>	27	A3	J2-39		
D7	cmp_adc<0>	28			CMP_ADC1	J2-0
C7	cmp_adc<1>	29			CMP_ADC2	J2-1
C6	cmp_adc<2>	30			CMP_ADC3	J2-2
A6	cmp_adc<3>	31			CMP_ADC4	J2-3
C5	nCpuRd_i	32	OE	J1-36		
B6	nCpuCs_i	33	CS2	J3-45		
D4	Rst_i	34	RESET	J3-46		
A5	cmp_adc<4>	35			CMP_ADC5	J2-4
B4	cmp_adc<5>	36			CMP_ADC6	J2-5
E13	cmp_adc<6>	37			CMP_ADC7	J2-6
D3	cmp_adc<7>	38			CMP_ADC8	J2-7
	GND	39	GND	J2-3		
	GND	40				

FPGA Module Implementation for 8-ch ADC

Knud Baastrup

The implemented modules have been documented in html using doxygen, which can be reached via the following link: http://lene-lasse.dk/doxygen/h2_fpga_ADC_8ch/index.html

The Wishbone skeleton in terms of Wishbone Master, Intercon and Syscon has been copied from a Wishbone code example provided by Morten Opprud.

The ADC entity is a simplified version of an ADC entity used during the PRO2 project in 2nd semester. This ADC entity will now just be a bit busier and support 8 channels by using the multiplexer and de-multiplexer as described in the module design.

Figure 14 shows the ASMD chart used for the implementation of the ADC entity. The ASMD chart shows how a sar register is used to assign a "1" to the r2r output for each of the 8 bit in the R-2R ladder one at a time and how the assigned "1" is kept if the cmp_adc goes high or skipped if the cmp_adc goes low. The state STAY will ensure that the bit pattern is assigned to the r2r output for sufficient time to allow the voltage on the inverting terminal to stabilize so we can trust the output from the comparator.

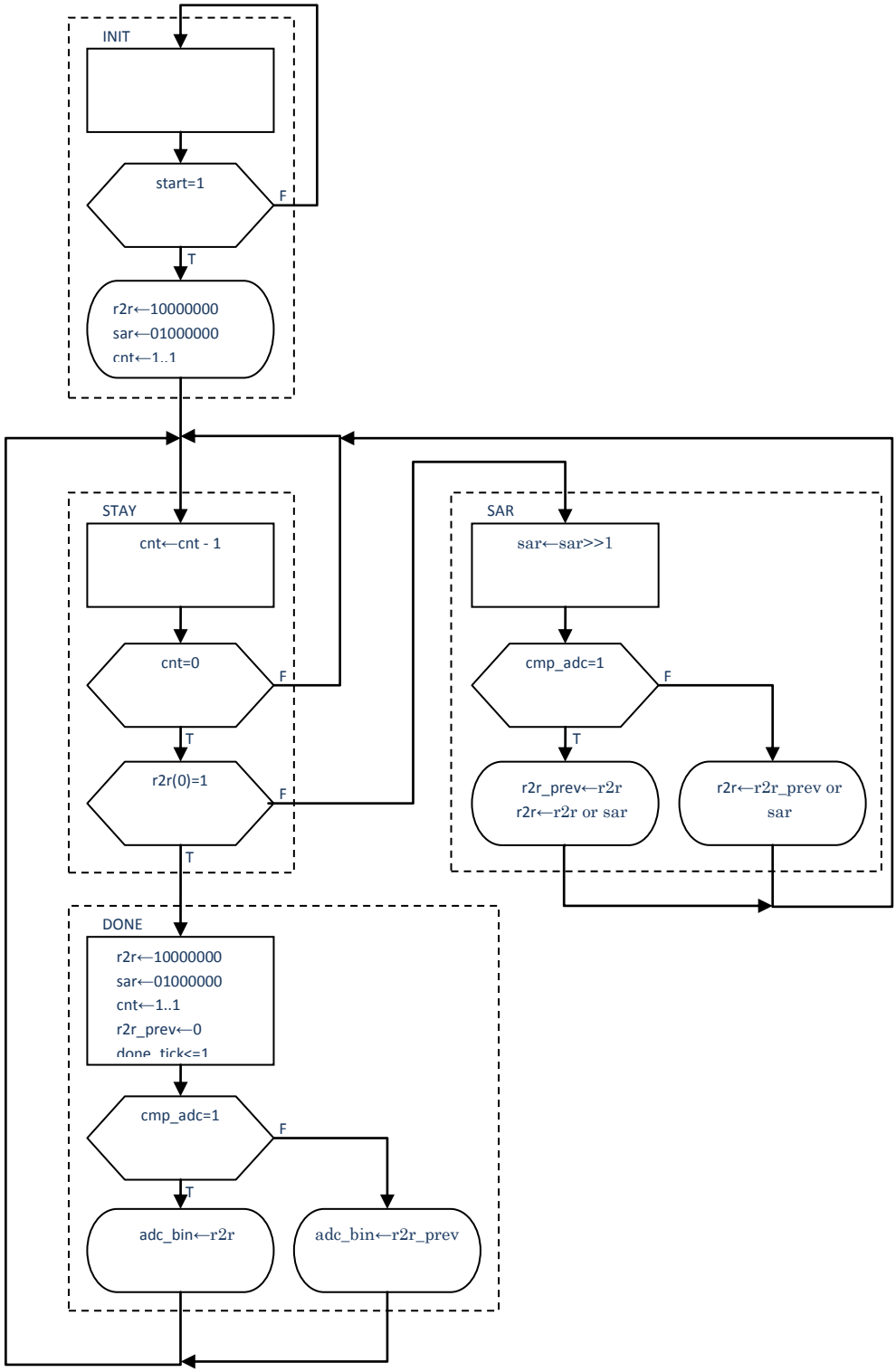


Figure 14 - ASMD chart used for the implementation of the ADC entity

FPGA Test Bench for 8-ch ADC

Knud Baastrup

Figure 15 shows the read-out of the 8 ADC registers maintained by the WB_ADC_8ch_Slave entity. The ADC registers are initialized with a number corresponding to the number of the register.

The last address is outside the address range for this slave and therefore no data is returned.

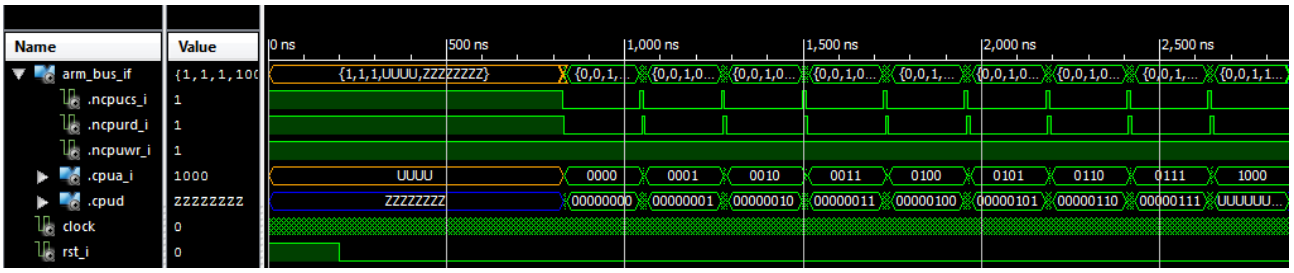


Figure 15: Read-out of the 8 ADC registers via Wishbone Interface

Next time timebox deployment will extend the test bench and as well provide the supporting HW required by the A/D conversion.

FINAL VERSION OF VOLTAGE AND CURRENT SENSOR

Knud Baastrup

The final version of the voltage and current sensor have been completed and is now also including a PCB layout as shown in Figure 17.

The very simple zener voltage regulator used in the prototype 1 version have been replaced by a LM2936 voltage regulator that opposite the zener voltage regulator acts in series with the load and therefore only dissipate heat in accordance with the actual load and not the assumed worst case load, which is the case for the zener regulator.

The LM2936 voltage regulator was mainly selected based on its capability to support an input voltage of 60V with a 50mA output and as well its low cost. The LM2936 also features low drop-out (can operate with a very small input–output differential voltage) and very low quiecent current (the current flowing through the system when no load is present), but these features are not important for this application.

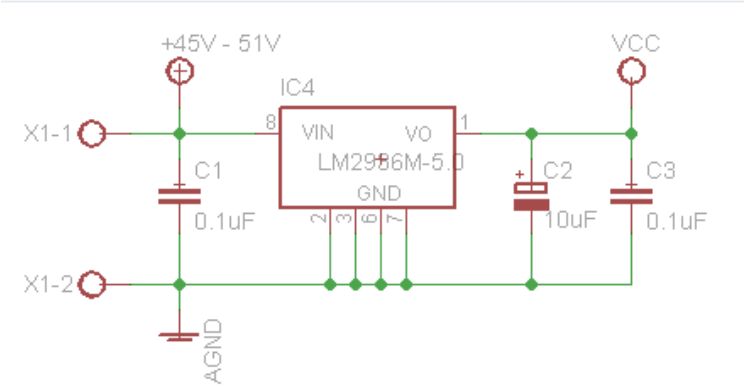


Figure 16: The LM2936 based voltage regulator that replace the zener regulator

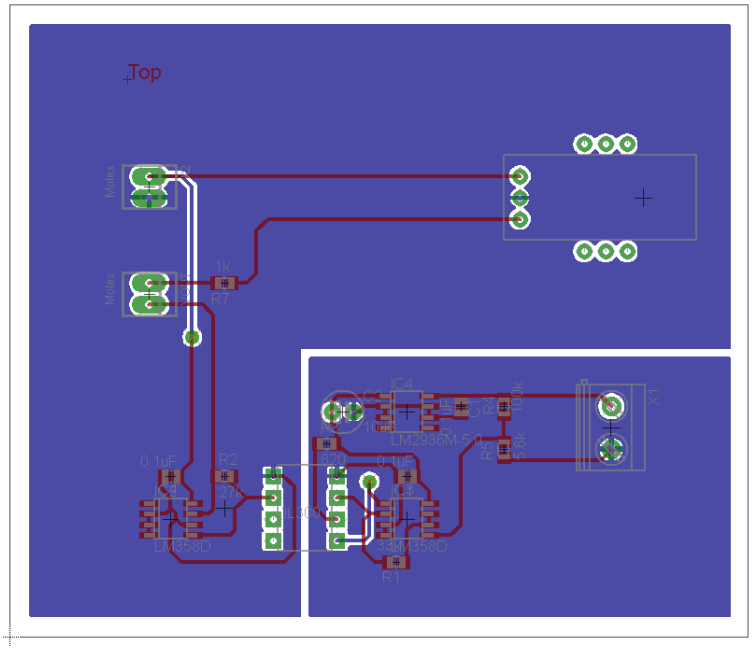


Figure 17: PCB layout for the Voltage and Current sensor

VERIFICATION

Knud Baastrup

The verification for timebox 4 is a re-test of the test cases prepared for timebox 2.

USABILITY TEST

Lasse Lykkegaard

In the current system there is no dedicated user interface designed to a none-skilled and ordinary user. The interface console (prepared as a low fidelity prototype above) is not done yet, and is anyway intended for programmers and senior service technicians maintaining the system, i.e. skilled employees with more than just a holistic understanding of the system. Such a system is not a good system for usability testing as test persons are expected to be without any knowledge of the system and with no conflict of interest.

The system interacts with the more regular users through a webserver which provides them with some basic facts and statistics of the system via a H2 website that is accessible from a browser like Internet Explorer. This way of providing information makes it possible for the user to obtain information through a medium with high accessibility of the user group.

Knud Baastrup

The Usability test will be used to measure to what extend the H2 website is usable and effective for the intended users which include maintenance personal, students and visitors at AU-HIH. The maintenance personal is skilled people that will visit the website regularly and in many cases acquire a login account while visitors in the other end might be people that just have an interest in green technology.

The test is performed in a controlled environment that allows the test to be repeated for each participating user in order to get some performance data that can be analyzed and compared. This is achieved by defining a number of task that each of the participating users must perform and as well some questions that is used (if needed) to get the users experience in using the website.

Participants

Lasse Lykkegaard

All test persons have basic or higher technical skills, based on their many years of experience in using a computer, technical machinery etc. and as a generation grown up with technique.

None of the test persons have any previous knowledge of the hydrogen system.

The system is to be useful for both trained personnel and first-time visitors.

Person ID	Name	Category
A	Esben Wolf	Maintenance Personal
B	Lene Lykkegaard	Visitor
C	Marit Søby Baastrup	Visitor
D	Poul Lehmann Thomsen	Visitor

Table 4: List of participants in usability test

Task Description

Lasse Lykkegaard

All test persons have gotten the same assignment, and was to carry out all tasks. All sessions ware video recorded for later analysis and documentation. Raw material is available at www.lene-lasse.dk/files/IDE

The test assignments are normal operations of the web-interface - some in greater extent used by maintenance personnel. All in all the main purpose was to identify problems and lack of information which we as developers take for granted, but is essential for visitors.

1. Check current system status on Electrolyzer:
    a. See average stats for the last week
    b. See efficiency graph for the last year
    c. Check current system status for Fuel cell /generator
2. Login:
    a. Login with user: ”test” and password=”test”
    b. Change email address to test@test.com
    c. Logout
3. Register as a user with own name and password:
    a. Login as that new user
    b. Join mailing list of the “voltageIn” –sensor. Set threshold value as 10 V
4. Retrieve information on max energy production of the Fuel Cell
5. Locate Knud’s email address and send an email to Knud.

REFERENCES

- EUDP: http://www.eudp.net