DIGITALA PROJEKT

WEATHER STATION

LUND UNIVERSITY

Department of Information Technologies

Jürgen Böhm

February 2005

Table of Content:

1. Abstract:

This document is describing the project work done during my exchange program. My goal was to build a weather station that is able to measure the temperature and the humidity. It should also store the minimum and maximum values with the time and date. As Processor is chose the AVR Atmega16 because it is widely spread and still in use. Further it is easy to use because everything like the memory is included. I chose all the other devices from a list of devices, which are available in the department.

1.1 Introduction

I decided to do this course because I want to combine my knowledge of programming and electronics into one piece of work. I chose a weather station as my project work because the worked seamed for me in a reasonable scale. The project took six weeks.
In the first few days I had to make my mind up which kind of devices I want to use. The second step was to build the schematic on a computer. For this kind of work we used PowerLogic from Innoveda. Afterwards started the actual work with equipping the board with the components.
I used following active components:

- Atmeaga16 (Processor)
- SHARP Dot-Matrix LCD Units LM162XXX (Display)
- ICM7170 (Real Time Clock)
- LM555 (Oscillator)
- LM35 (Temperature Sensor)

Furthermore I used a lot of passive devices around the just mentioned for controlling and driving.

I decided to assemble the board step by step to get first one part working until I continue with the next step. Now the programming started which was done in C. As the software development kit I used a simple program called Programmers Notepad produced by Mingus. As debugging tool we used the AVR Studio 4 produced by ATMEL. I did the following main steps:

- Connecting the microprocessor with the JTAG interface to the computer.
- Connecting the display to the microprocessor and display text.
- Connecting the RTC to the microprocessor to read/write the time and commands.
- Connecting the temperature sensor to the ADC of the microprocessor.
- Connecting the humidity measuring circuit to the Input Capture Unit of the microprocessor.

In between a lot of small steps had been necessary to adjust the values and for testing. After connecting all devices it was time to create one big program. To keep the programming for myself clearly I decided to change the already existing C files into header files and include them in the main program. I know that this kind of programming is not state-of-the-art but in this case I just want to have the most important functions separate for a better understanding.

2. Description of the Circuit

We can divide the board into the five already mentioned parts and I'm going to explain these in the following pages.

When I started to design the schematic it was soon clear that there maybe not enough ports. To save pins I decided to design a little bus system (*Figure 1*). It's connecting the pins D0 to D7 of the LCD and the RTC with the pins PB0 to PB7 on the microcontroller. On the board it is very easy to find since I used just for this connections blue wire. For the JTAG interface and command lines like E or RD/WR I used white wires. Yellow are all kind of other connections. Black is ground and red is phase. A picture of the backside of the board is shown in the end of the description

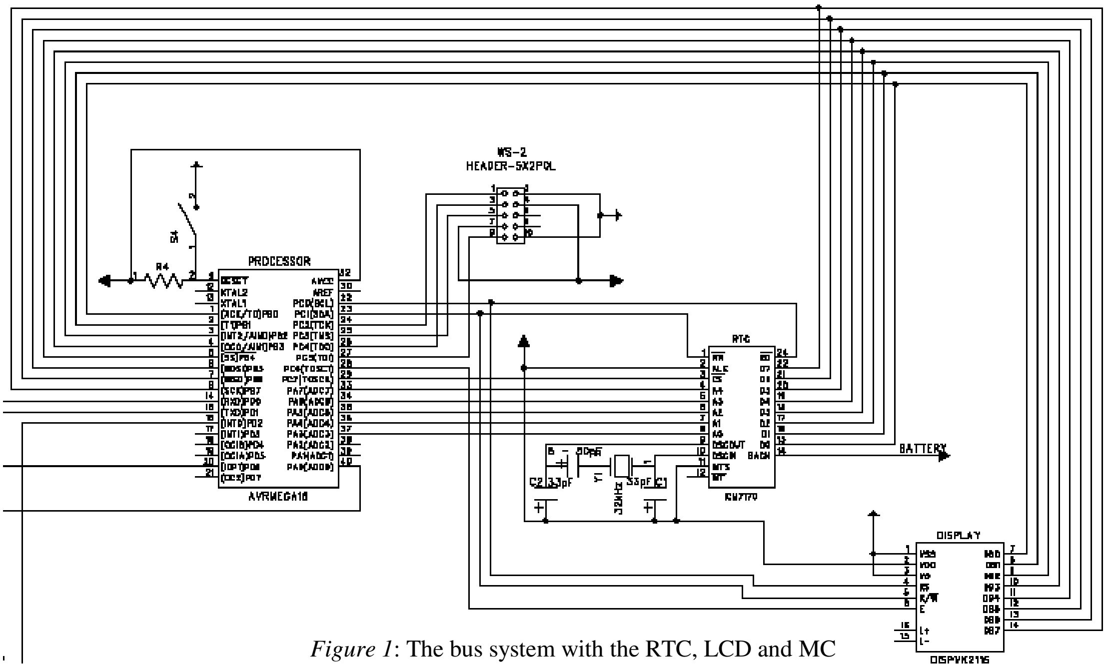


Figure 1: The bus system with the RTC, LCD and MC

2.1 The Main Circuit

The hardware is strongly connected with the software and vice versa. The program has to wait until the display is ready (keep the time frame) or has to run a routine when an interrupt is triggered. On the other hand is the display or the real time clock not able to run from it self and show or return the desired values. Therefore it is very important to keep a special eye on the timing characteristics. This is also very important because a user is pressing a button not just for a millisecond and releasing it instantly.

To run the whole circuit for development purposes it is necessary to connect the circuit to a computer. This is done with the JTAG interface which must be connected to certain pins. If this is done and the microprocessor is connected to GND and Vcc the computer (AVR Studio 4) should detect the microprocessor.

JTAG Connections	
TCK (Pin 1)	PC2 (Port C)
TMS (Pin 5)	PC3 (Port C)
TDO (Pin 3)	PC4 (Port C)
T DI (Pin 9)	PC5 (Port C)
VTref (Pin 4)	VTG
Vsupply (Pin 7)	Vcc
GND (Pin 2/10)	GND

2.2 The Display

The display was very tricky and I had to take a lot of things into consideration. Especially a lot of adjustment what the time in between the comments concerns was needed. The pins for the data in- and output are connected over the data bus with the microcontroller (PB0 – PB7). I connected V0 to ground since I didn’t want change the contrast of the Display. RS is connected to PC0. R/W is connected to PC1. And E is connected to PC6. To read or write to the display it is important that the display toggles all the time.

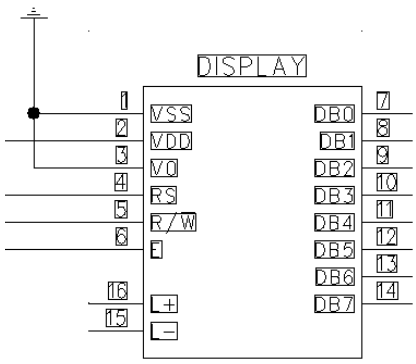


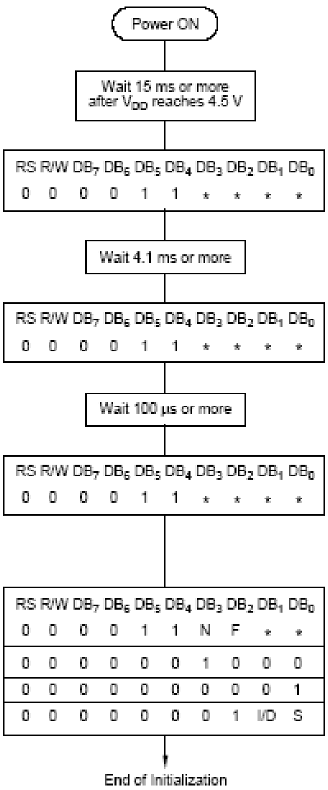
Figure 2: The LCD

In the beginning a special kind of initialization has to be made before anything else can be done. This is shown in the Diagram on the bottom of the page. The timing for the writing or reading activity is also shown in the two diagrams at the bottom of the page. Now I will give some explanations too the pins for controlling the circuit.

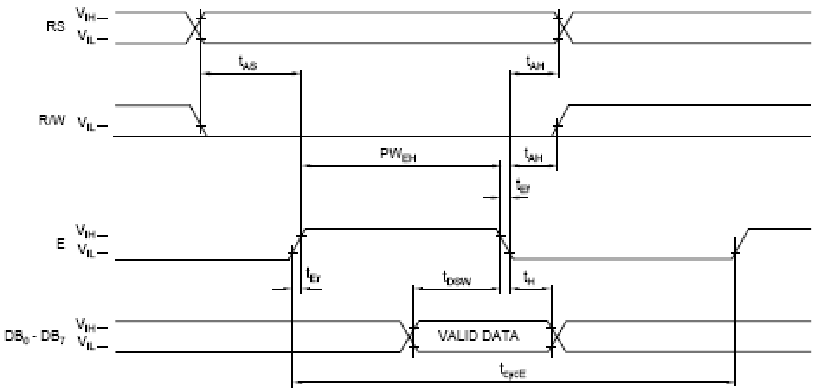
RS RS is the Register Select signal
“0” Addresses the instruction Register when we are writing and reads the Busy Flag and the address counter. “1” We address the data register (read and write).

R/W R/W is the read / write signal; “0” Writing, “1” Reading

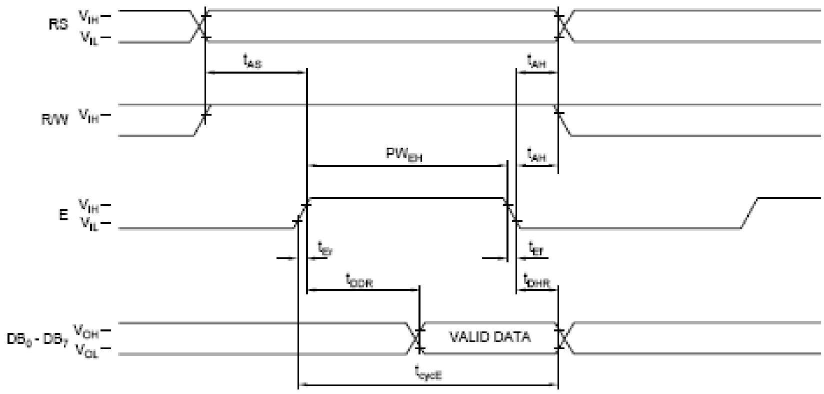
E E is the operation enable signal for read and write.



Initialize LCD



(For data sent from the external microprocessor to the LCD unit)



(For data sent from the LCD unit to the external microprocessor)

2.3 The Real Time Clock

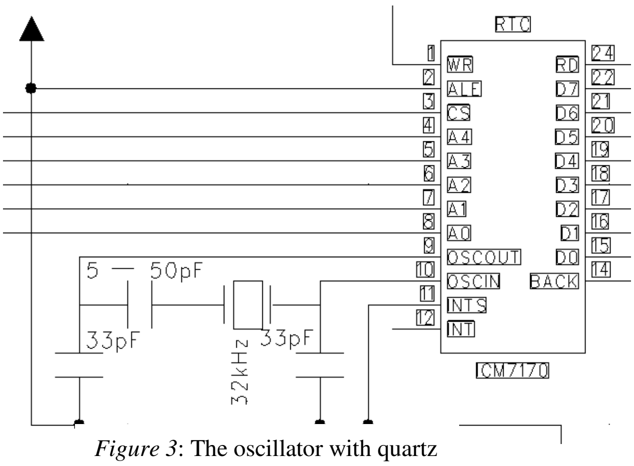
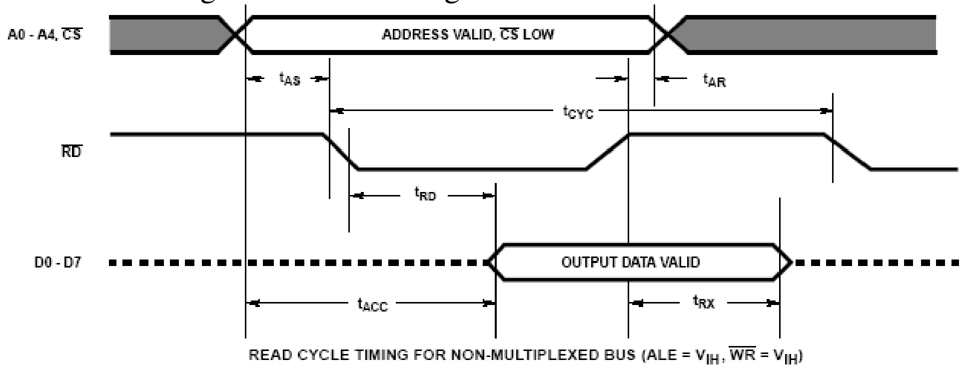
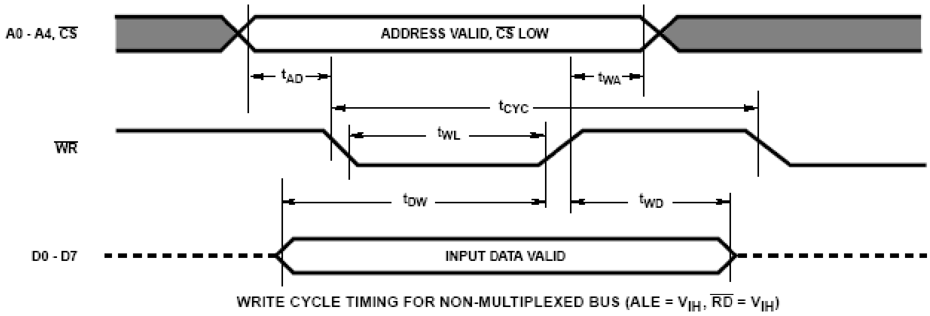


Figure 3: The oscillator with quartz

The real time clock is used to give us the time and date on which time the minimum and Maximum values occurred and of course the actual time as well. Furthermore is it a clock to provide us an accurate interrupt every second. This function is especially needed for counting the frequency of the humidity sensor.
This time it was not necessary to watch on time characteristics because the device is fast switching. There is only on but very important fact that before we are going to read minutes, days or month we have to read the 1/100 seconds. By accessing the 1/100 of seconds counter, an internal store signal is generated and data from all the counters is transferred into a 36-bit latch and ready for demand. If it is not done the old values will be retrieved.
The pins D0 – D7 are connected in the same manner like the display with the pins PB0 – PB7 and the microprocessor. The interrupt pin goes to the pin PD6 (INT1). The pins A0 –A4 are for the address input. They are connected to the pins PA3 –PA7 on the microcontroller. RD goes to PC0. WR is connected to pin PC1. CS goes to pin PC7. ALE is connected to high because we connected the RTC directly to the microprocessor. You have probably notice that the RD/WR lines and the R/S and RW lines of the display go the same pins. I made this decision to save pins again. In the case of the display I select it with the Enable pin and the RTC with Chip Select pin. The RD, WR and CS signals are active low inputs. Data is placed on the bus from counters or registers when RD is logic “0”. Data is transferred to counters or registers when WR is logic “0”. RD and WR must be accompanied by logic “0” to speak with the device. The timing is shown in the figure below.





2.4 The Temperature Sensor LM35

The integrated circuit LM35 is a temperature sensor, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature (*Figure 4*). The problem was that the working range in the standard setup is just from +2°C to +150°C. This is of course for temperature measurements not appropriate. Due to this fact I had to find a solution to change the working range of the sensor. In the datasheets I found a circuit which range is suitable (*Figure 5*). Now I had a span from –55°C to 155°C. Measurements on an external however showed that the change of the algebraic sign is causing trouble at the microcontroller. After continuing measurements we came to the solution to us just the positive output (OUTPUT).

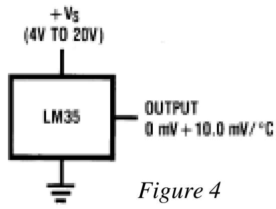


Figure 4

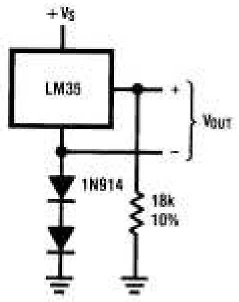


Figure 5

The output voltage is connected to the PA0 (ADC0). The ADC is by the program set to a clock frequency of 125kHz since we need for the successive approximation a clock frequency between 50kHz and 200kHz. Furthermore is an internal reference of 2,54 volt in use. The converted value can be found in the ADC Result Registers (ADCL, ADCH). It is important that we read first the ADCL (low) and than the ADCH (high) register because otherwise are the results lost. To recalculate the values a formula is given.

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}$$

The figure below shows the original circuit as how I use it on my board.

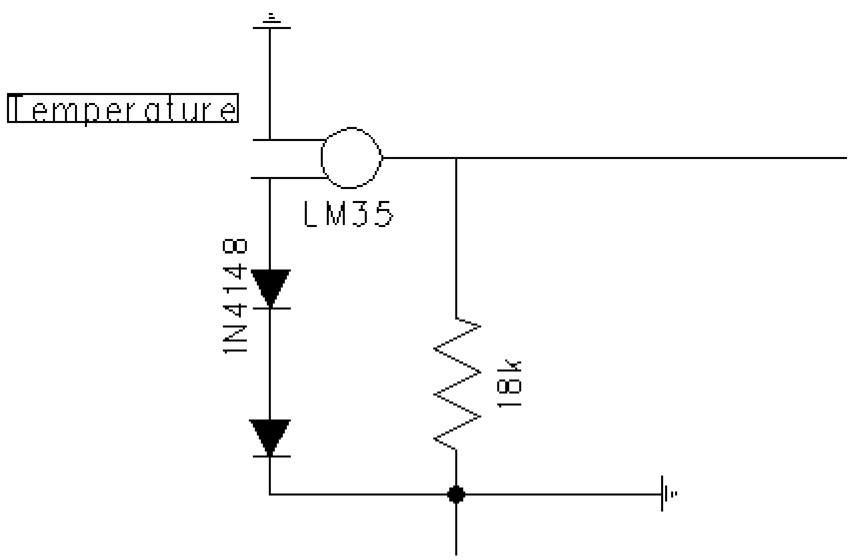


Figure 6: The bus system with the RTC, LCD and MC

2.5 The Humidity Sensor

For the measurement of the humidity I use a capacitor, which is open, and the air is able to flow through. The dielectric is therefore the surrounding air. With this alone it is of course not possible to measure the humidity. Thus I decided to generate a frequency, which would change with the changing C of the sensor. Since the microprocessor needs a digital frequency with proper positive and negative edges it is not possible to use an RC oscillator. In the database for electronic devices I found a timer LM555 that is also able to run as an oscillator. If we connect the circuit like shown in the figure we receive an astable operation. The circuit will trigger itself and free run as a multivibrator. The sensor charges trough $R_A + R_B$ and discharges through R_B. Consequently the duty cycle can be set by the ratio of the two resistors. The generated frequency can be calculated by the formula:

$$f = \frac{1.44}{(R_A + 2R_B) \cdot C}$$

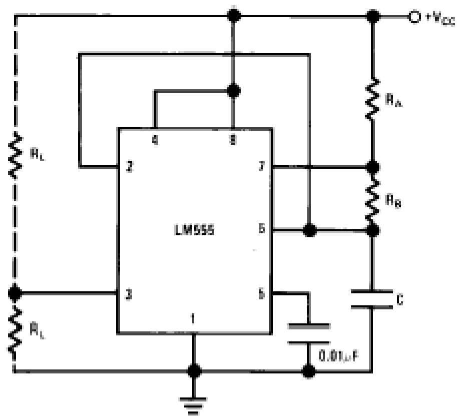


Figure 7: LM555 as an oscillator

$R_A + 2R_B$ is put together of both charge time ($R_A + R_B$) and discharge time R_B. Before I did the calculation I determined the range for the frequency at 1kHz. Afterwards I measured the capacitance of the sensor under normal conditions. I received the value 480nF. As a result of the calculation I received a value of 3MΩ R_A and $2R_B$.

In the figure below is the complete circuit shown how it is built on the board.

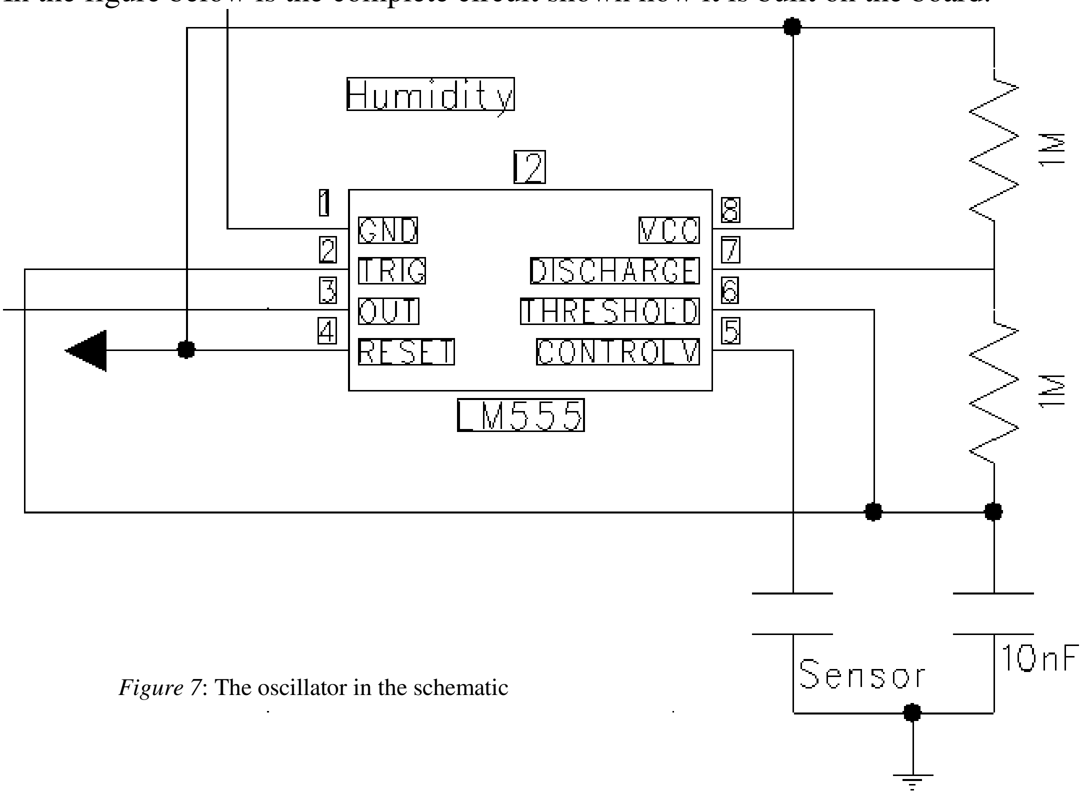


Figure 7: The oscillator in the schematic

3. Manual For Usage

On the board are four buttons placed close to the Display. Only the button R is fulfilling the same purpose in all situations. When we press R a hardware reset is triggered. All the other Buttons can have different functions in different situations.

Buttons Layout

1. Change Options Temperature or Humidity / Count up

2. Min / Max Min and Max value of the chosen option will be shown / Count down

3. Reset Reset Min/Max / Set time / Store time

Humidity or Temperature

To change from the temperature value to the humidity value we have just to press the one when we are in one of both menus.

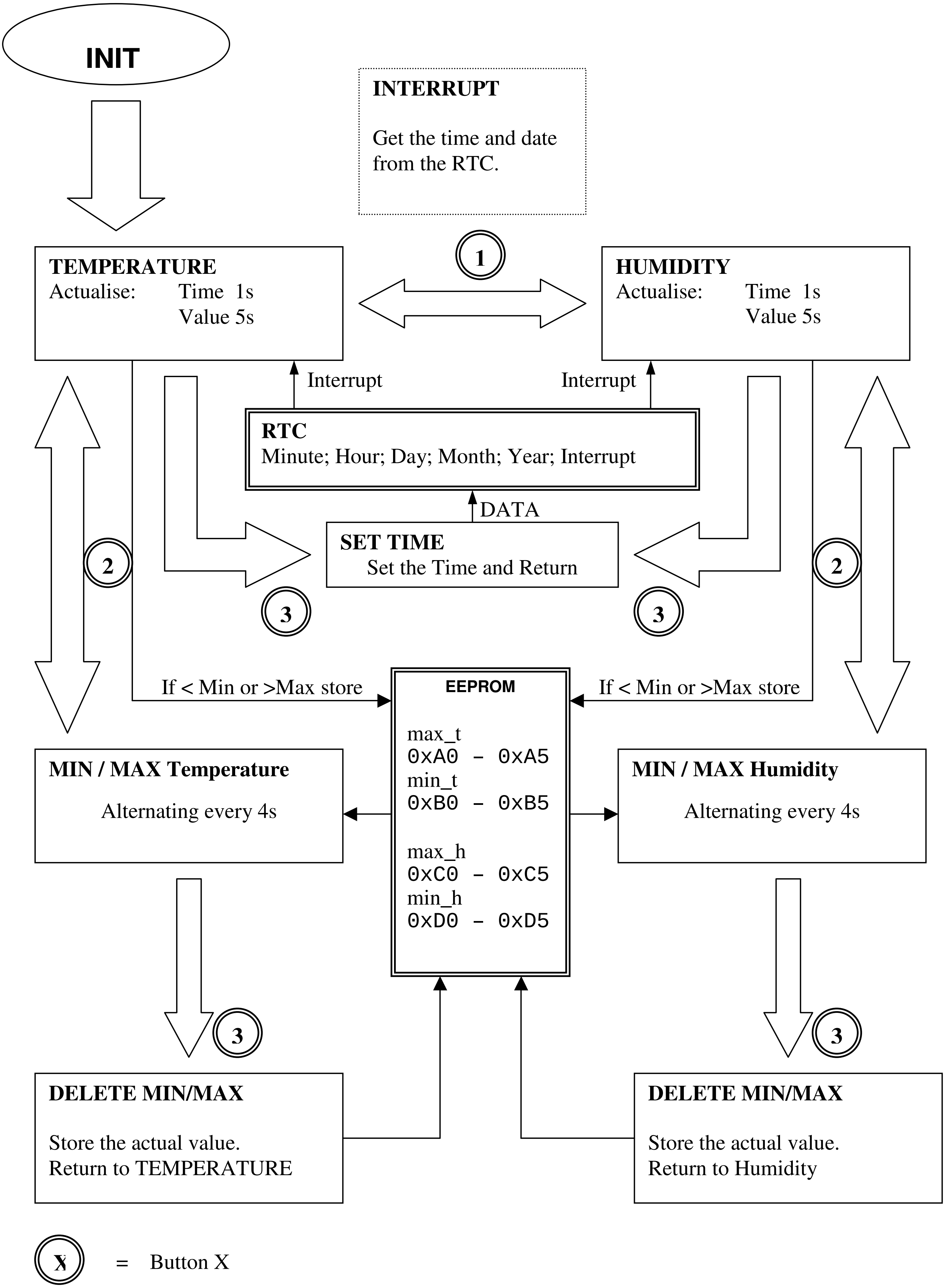
Set Time

- Press button 3 when the temperature or humidity is shown on the Display
- hh.mm will be shown on the display. First we set the hours with pressing 1 (up) or 2 (down). If the right value is set we have to press the 3. The value is stored and we can the set minutes. If the right value is set we have to press the 3 and we jump to the next step
- Now is dd.mm.yy shown on the display and we can set it in same manner like we did it with the time.
- After pressing the Button 3 after we entered the year we jump back to the main menu.

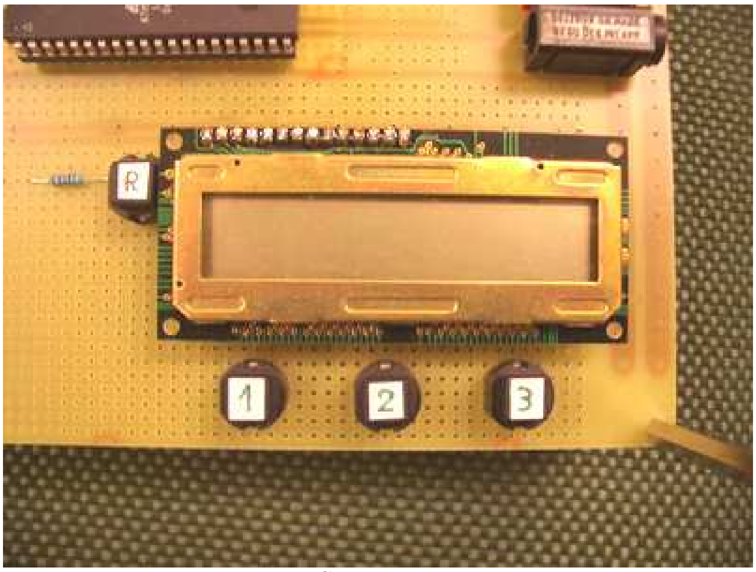
Min/ Max

Chose humidity or temperature and press button 2. You will see alternating the min and max values. To get back to the actual values press button 2 again.
To erase min and max values just press the button 3 when they are shown on the display.

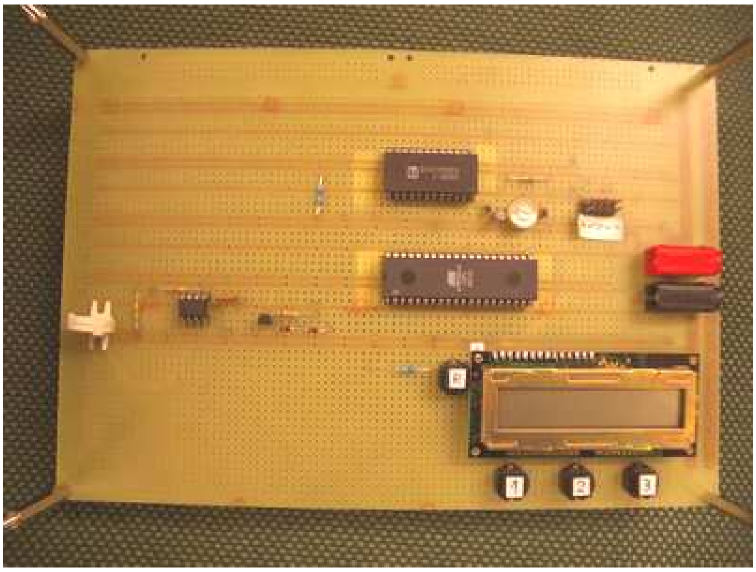
3.1 Flow Chart



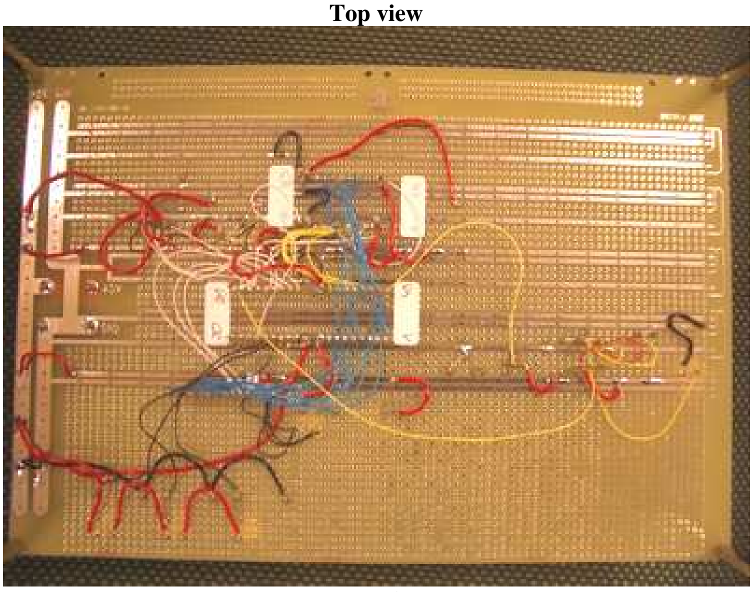
4. Pictures Of The Finished Board



The LCD and the buttons

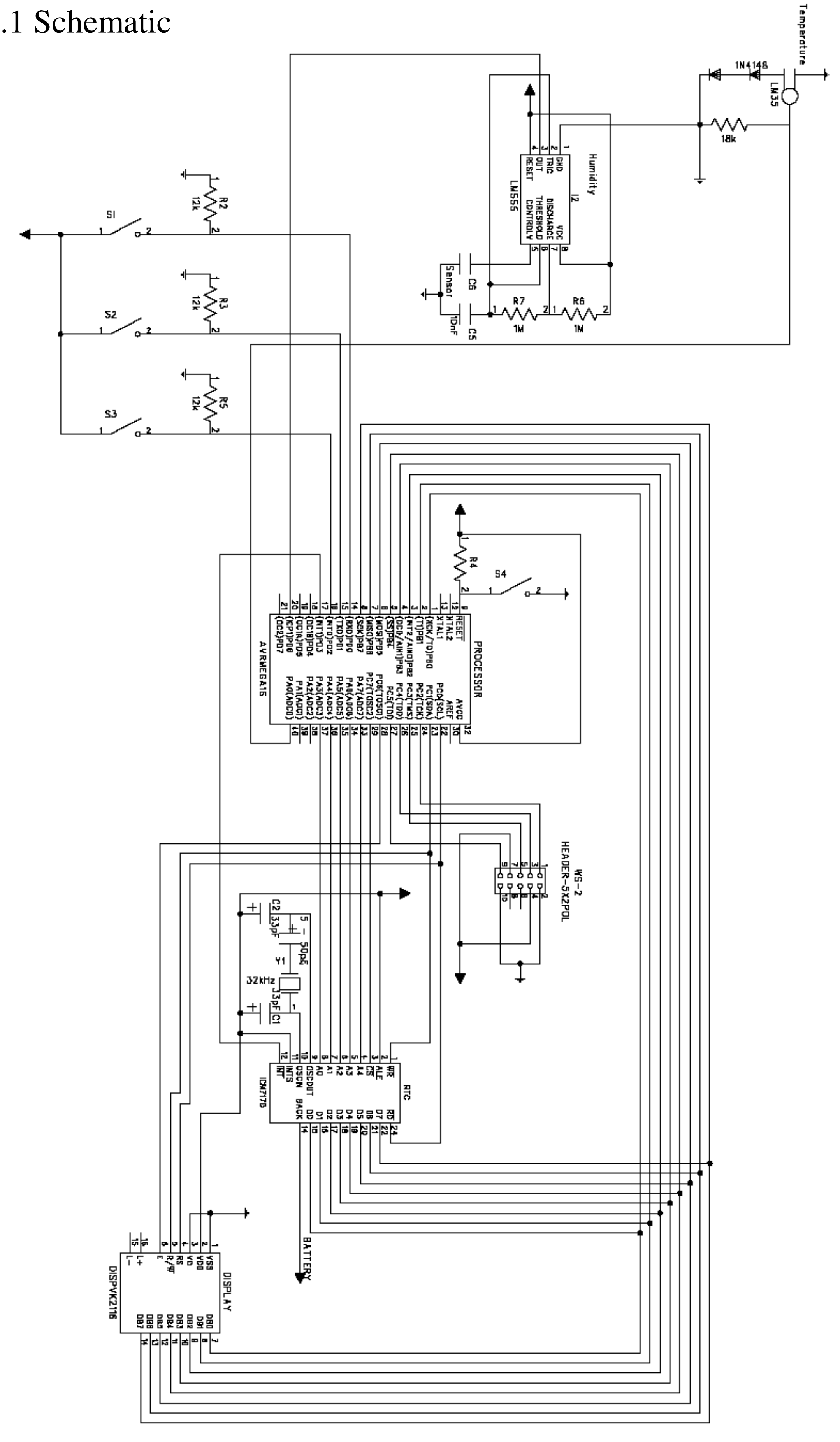


Top view



Back view

4.1 Schematic



5. Result

This was my first project and after three Semester of studies quite a challenge. It was not so easy for me to see where I've to start to work. It was for me very important to see how hardware and software work together. Debugging was now much harder since the error could be found in the software or hardware part. It became soon clear to me that each active device has a little specialty I had to take care of. The trickiest part was the timing of the display but also the latching of data in RTC was not so clear to understand.
Otherwise I had to change my way of thinking from time to time. A good example is the temperature measurement. First I thought that I have to take care of negative int value when we have a temperature below 0°C. This of course is not the case and voltage differs just between 1V (-20°C) and 4V (100°C) for example. All the small things are mandatory to know and so important for the job. Unfortunately are these things in normal university courses not taught and its just possible to learn it due to practical work.
A different thing was that I noticed that I have take care of the recourses. There are not an infinite number of ports. That's the reason I designed the bus. I also used pin PC0 for the RD (RTC) and RS (LCD) and PC1 for the WR (RTC) and R/W (LCD) to save pins.
If I would start to build the board new from scratch I would work and think completely different. The first thing I would change are the buttons because they should switch with interrupts instantly and not after an always different time span.

6. Reference & Addresses

Department of Information Technology
http://www.it.lth.se/

Course "Digitala Projekt"
http://www.it.lth.se/digp/information.asp

Datasheets for the Devices used in the course.
http://www.it.lth.se/digp/datablad/datablad_index.asp

My home university
http://www.fh-konstanz.de/