Digitala projekt, större kurs

Väderstationen

MotorOla

Henrik Kjellberg, et05hk6
2009-08-24

Table of Contents

Abstract

The project goal was to build a module based weather station. The system should have one module for measuring and sampling different weather information types, like temperature, relative humidity, air pressure and so on. One module for displaying the sampled and calculated information. And one module for handling and controlling the other units, to store the samples and to calculate average values and predictions.

The system should support one calculation unit, up to seven display units and up to eight sampling units.

Introduction

The system is based upon modules. One module for measuring and sampling different weather information types (Measuring unit, MU), for example temperature, relative humidity and air pressure. One module for displaying the sampled and calculated information (Display unit, DU). And one module for handling and controlling the other units, to store the samples and to calculate average values and predictions (Calculation unit, CU).

The measuring unit can sample an arbitrary numbers of the supported information types.
The display unit can display an arbitrary numbers of the supported information types.

The calculation unit asks the measuring unit for a new sample, the measuring unit takes a sample and pass it over to the calculation unit, which saves it and updates its average values and predictions.

When a display unit wants some samples to display, it asks the calculation unit for the wanted values and the calculation unit sends them over.

For a full specification of the features, see Table 3: Initial technical specifications in Appendix A on page 10.

Technical solution - Hardware

The circuit diagrams for the CU and DU can be found in Illustration 1: Calculation Unit, Circuit diagram and Illustration 2: Display Unit, Circuit diagram on pages 11 and 12.

The CU is based on a Motorola 68008. The program memory is a 27C256, a 32kB EPROM memory. Since the unit is to store samples for as long as possible, a vast memory is desirable. The working memory selected is therefore a KM684000, a 512kB SRAM memory selected for its vast amount of space. As general inputs/outputs, two regular 74HC573 D-latches are selected. One is connected as input and one as output and they are not interconnected. That means that outputted values cannot be read in again, the current value has to be remembered in software. By putting a third D-latch connected to the D-latch connected as output, the read in feature could have been achieved, but that was found superfluous. The i/o's are used for the software SPI interface. The SPI interface is used for the nRF24L01 radio module and, eventually, a µALFAT SD-card module. The SD-card module were never implemented though.
For time-stamping the samples, a real time clock is used. The clock is a ICM7170, an easy to use but still very functional real time clock.
The peripherals are addressed using two PALCE22v10 programmable logic circuits. They also handles the interrupt request signals.

The DU is based on an Atmel AVR Atmega168, with a Batron 128*64 pixels graphical display and three pushbuttons. The built-in hardware SPI interface is used to connect the nRF24L01 radio module.

Technical solution - Software

The Calculation Unit, CU

The CU software is built around three tasks:

- Communication handler
- Isochronous requests
  - Sample
  - Refresh MU-list
- Calculator

Communication handler

The communication handler takes care of the communication. When the radio module announces that a new packet has arrived, the communication handler retrieves the received byte string, decodes it to a packet and takes action according to what was received.

The communication handler mainly consists of a receiver and a packet inspector, which runs from the radio interrupt service routine, a sender, which is called by other methods when needed, a coder which encodes data for the radio packets and a decoder which decodes the radio packets.

Isochronous requests

The CU holds a list of all available MU's and all samples from the corresponding unit.
At isochronous intervals the CU shall request samples from the connected MU's. This is done in the timer interrupt handler.

The CU shall keep track of all the MU's currently connected. This is done by sending a status request to the unit. This is also done in the timer interrupt handler.
The CU shall also search for new, unknown MU's. When a new unit is found, it is given an address of 4bits. The unit is added along with its address to a list of units. The unit changes its receiver address to 0x6950c17e0<address>.

The CU shall also keep track of all the DU's currently connected. This is just so that addresses can be leased correctly. In contrast to the addressing of MU's, the CU don't actively search for DU's, but rather waits for an address request. When the address is leased, the CU is isochronously checking so that the DU is still needing the address. If not, the address is reclaimed.

Calculator

The calculator is to be run from the timer interrupt handler and its task is to calculate the average values from all the enabled MU's. It shall also calculate the tendency for the weather based on the air pressure tendency.

The Display Unit, DU

The DU is built around three tasks:

- Navigation
- Communication Handler
- Data updater

Navigation

The navigation task is driven by pin change interrupts for the buttons. The navigation handles the user interface (UI). The UI is built around a structure, were each object (called a node) has a name, a list of sub-nodes, an integer that keeps track of which sub-node that is selected and a function pointer that points to the function that is to be executed when the node is “entered”.

When pressing the up or down buttons, the selected-value is changed. When pressing the OK button, the sub-node corresponding to the current selected-value, is set as current node and the corresponding function is called. The sub-node nr. 0 is always the parent node.

Normal menus are simply nodes which has its buttons as sub-nodes and a display-menu function in its corresponding function pointer. The display-menu function takes the sub-nodes array and displays one button for each sub-node.

To change a integer value or something alike, the nodes function launches a new screen and takes control over the buttons by disabling the corresponding interrupts and polling the button inputs.

Communication Handler

The communication handler takes care of the communication. Due to physical constraints, the interrupt request pin of the radio cannot be used. Instead the status register of the radio has to be polled to see when data has arrived. When the a new packet has arrived, the communication handler retrieves the received byte string, decodes it to a packet and takes action according to what was received.

The communication handler mainly consists of a receiver and a packet inspector, which is run from the timer interrupt service routine, a sender, which is called by other methods when needed, a coder which encodes data for the radio packets and a decoder which decodes the radio packets.

Data Updater

The data updater simply updates the data to be displayed on the screen. An array of structures, containing a value, what MU it origins from and what Weather Information Type(WIT) it contains, is kept up to date. The array has one field for each field of the display. This way the content of the fields can easily be changed. The history fields, which is showing graphs with more than one value, has its own array and structure. The structure of the history fields contains an array of eight values instead of one value.

The Protocol

Packets:

To	From	Ctrl	Unit	Length	Data0	Data1	...
4bit	4bit	4bit	4bit	8bit	8bit	8bit	...

To/From:

The To and From addresses are 4bits. The CU has the address 0000. The other addresses is assigned by the CU. Addresses 0001 to 0111 is assigned to DU's and 1000 to 1111 is assigned to MU's.

Ctrl:

The control field is 4bits long. The following commands are defined:

Bits	Name	Explanation

0x0	STATUS	Current status of this unit. The status consists of:
		Data0=Address
		Data1=Type, {DU,MU,CU}
		Data2=WIT-mask
		if(CU):
		Data3=High byte of WIT-mask
		Data4=year
		Data5=month
		Data6=day
		Data7=hour
		Data8=min
		Data9=sec
		if(DU):
		Data3=High byte of WIT-mask
		If a MU sends a STATUS packet to the CU, with 0x0 as address, the CU responds with a SET_ADDR and adds the MU to the MU-list. No more than one MU can perform a connection at a time.
		If a DU sends a STATUS packet to the CU, with 0x0 as address, the CU responds with a SET_ADDR. No more than one DU can perform a connection at a time.
		The WIT-mask is a bit field, telling which of the 11 weather information types defined in Table 2 – Weather information types, WIT's that are available on the unit.
		The CU and the DU needs two bytes for the WIT-mask. This is not needed for the MU since it cannot be equipped with the history WIT's (see Table 2 – Weather information types, WIT's).
		The time of the system is maintained in the real time clock of the CU. If the DU is to display the time, it has to ask the CU for the time.

0x1	STATUS_REQ	Request a status packet from a unit.
0x2	ACK	Acknowledge. This is not needed in the current implementation due to built in auto acknowledgment in the nRF24L01 radio module, but is rather intended for future use.
0x3	FIND_MU	Global shout out in search of new MU's. The address 0x10 is used to indicate uninitiated mu. Since the protocol address is 4bit long, 0x10 is outside address range and therefore not valid as an address. The MU responds with a STATUS packet, to which the CU responds with a SET_ADDR. No more than one MU can perform a connection at a time.
0x4	SET_ADDR	Set the address of a unit. Data0 contains the address.
0x5	SAMP_REQ	Request data. The CU requests data from the MU's by sending a SAMP_REQ to the MU, with the MU's address in both the To field and the Unit field. A DU requests data from the CU or any MU by sending a SAMP_REQ to the CU with the MU's/CU's address in the Unit field. The data-fields is a list of the WIT's to sample.
0x6	SAMP	The sample response. From a MU to the CU or the CU to a DU. The UNIT field is set to the MU's address, or when requesting average values, the CU's address. Data0 is a data-header for Data1, Data2 is a data-header for Data3 a.s.o. The header consists of the WIT as lower four bits and a sequence number as higher four bits. The sequence number is used for history types and >8bit data. The WIT's are defined in Table 2 – Weather information types, WIT's.
0x7	STOF	Soft turnoff-mask. When a MU's WIT is softly turned off, it is excluded from the CU's average calculations.
0x8	HTOF	Hard turnoff-mask. When a MU's WIT is turned off hard, it is not sampled at all.
0x9	ESD	Enable log to SD-card. When going from SD-card disabled to SD-card enabled, the CU saves all old samples in one directory (labeled <date>_backlog) and all the new samples in another directory (labeled <date>_log). Every WIT of every MU is logged into a separate file, labeled <WIT>_<MU>.txt The SD-card feature is not yet implemented.
0xA	DSD	Disable log to SD-card.
0xB	SET_CU_TIME	Set the time in the real time clock of the CU. Data0=year, Data1=month, Data2=day, Data3=hour, Data4=min, Data5=sec.
0xC	SAMP_INT	Set the sampling interval. The value is given in half minutes. The Data0 field holds the WIT and Data1 holds the sampling interval. The MU of interest is specified in the Unit field.

0xD	CLEAR_HIST	Clears the history for the MU specified in data0.
0xE	SET_HIST_INT	For later use.

Table 1 – Ctrl commands

Unit

The Unit field is to specify the unit of interest. If a DU wants information about a MU, it doesn't send the request directly to the MU but to the CU, with the address of the MU in the Unit-field.

Length

The length is the number of data bytes that follows.

Weather information types, WIT's

0x0	Temperature	The current temperature
0x1	Relative humidity	The current relative humidity
0x2	Atmospheric pressure	The current atmospheric pressure. 127 = 1000hPa
0x3	Rain	Amount of rain. For CU, the value is since last CLEAR_HIST. For MU, the value is since last sample sent.
0x4	Wind speed	The current wind speed
0x5	Wind direction	The current wind direction
0x6	Tendency	A indication of the weather tendency
0x7	History of relative humidity	A series of 8 relative humidity samples with the interval set by SET_HIST_INT
0x8	History of atmospheric pressure	A series of 8 atmospheric pressure samples with the interval set by SET_HIST_INT
0x9	History of temperature	A series of 8 temperature samples with the interval set by SET_HIST_INT
0xA	History of wind speed	A series of 8 wind speed samples with the interval set by SET_HIST_INT

Table 2 – Weather information types, WIT's

Results and discussion

The modularity of the system is, as intended, very high. The MU's and DU's can easily be changed and redesigned. During the development, a special DU was built. The unit had an RS232 interface instead of the LCD display and the buttons found on the main display unit. This made it easier to read out values from the radio module and give instructions. The step from a suchlike unit to a full feathered display unit based on a PC is not far.
The MU's can have sensors for one up to eleven different weather information types, and the available information types can easily be turned on and off. They can even be excluded from average value calculation and tendency predictions. This is good if, for example, one temperature sensor is located in direct sunlight.
The code is also designed with modularity in mind. Large portions of the code can be used in all units, with none or small changes.

The development using Motorola 68008 introduced a lot of problems not encountered before. The 68008 is a naked processor, without any i/o peripherals, timers or memory's. This meant that even a simple thing like toggling a pin introduced problems like converting an address to a chip-select signals and feed it to a latch. All memory mapping had to be made manually using programmable logic circuits. The processor has no on chip clock-generator like modern microcontrollers, instead it needs a clock signal at TTL-level.

The development using Atmel Atmega on the other hand, introduced some other problems. The Atmega has all peripherals needed, memory's, built in clock-generator, i/o's, timers and even SPI interface. But with built in memory, the memory area is limited. The project started using an Atmega88, with 8kB of flash memory. This had to be upgraded to an Atmega168, with 16kB of flash memory due to the massive volume of code needed for the graphical user interface. The code for the LCD takes very much space, due to the fact that every bit ever to put on the display, has to be stored in the microcontroller. An alphanumeric display would have had preprogrammed fonts, but that is not the case with graphical displays. On top of that comes the menu system, with all the menu alternatives and their names.
Another problem encountered using the Atmega was the limited number of i/o's available. The Atmega 88/168 has 23 general input/output pins. That is just enough for the graphical display, three pushbuttons and the required pins for the radio module. What did not fit was the IRQ-pin from the radio module. That meant that the radio had to be polled for status at frequent intervals instead of interrupting the processor.

The project started out as a project for two persons, but during mid-term the project devolved to a one man project. Due to this fact, the project had to be cut down drastically.
The MU was totally omitted and some parts of the other units were left out as well.
The support for SD-card were never implemented, but if it were to be, it would have been so using a SD-card module from µALFAT interfaced using SPI.
Since the MU was left out, the part of the CU that handles the MU's and corresponding samples, is not fully tested. It is implemented, but not tested on real MU's.
Large portions of the protocol is aimed for the MU, and is therefore not implemented in the communication handler code.

Appendix

Appendix A.

MotorOla-1	The system shall support the following weather information types (WIT): ● Temperature ● Relative humidity ● Atmospheric pressure ● Rain ● Wind speed ● Wind direction ● Tendency ● History of relative humidity ● History of atmospheric pressure ● History of temperature ● History of wind speed The system shall be able to sample, store and display listed weather information.
MotorOla-2	The system shall have three different types of units. 1. Display units, called DU 2. Calculation units, called CU 3. Measuring units, called MU The system shall support one CU to handle up to 8 MUs and 7 DUs.
MotorOla-3	The MU shall have sensors for one or more of the following WIT's: ● Temperature ● Relative humidity ● Atmospheric pressure ● Rain ● Wind speed ● Wind direction Each MU can, but does not have to, have one sensor for each WIT.
MotorOla-4	The, by MU, sampled data shall be sent to the CU.
MotorOla-5	The CU shall store the sampled data locally for up to a week.
MotorOla-6	The CU shall calculate tendencies and average values from the provided samples.
MotorOla-7	The CU shall be able to store the final calculated weather information, as well as sampled data from each individual MU, in text format with timestamps, on a SD-card, if such is provided.
MotorOla-8	The DU shall be able to display some, or all, weather information provided by the CU.
MotorOla-9	The DU shall provide a user interface, UI, for configuration of the system.
MotorOla-10	The UI shall provide the possibility to disable every single WIT from every single MU, individually. Both by disable sampling and by excluding from average calculations.
MotorOla-11	The DU shall provide the possibility to set the CU's time and sampling interval.

Table 3: Initial technical specifications

Appendix B

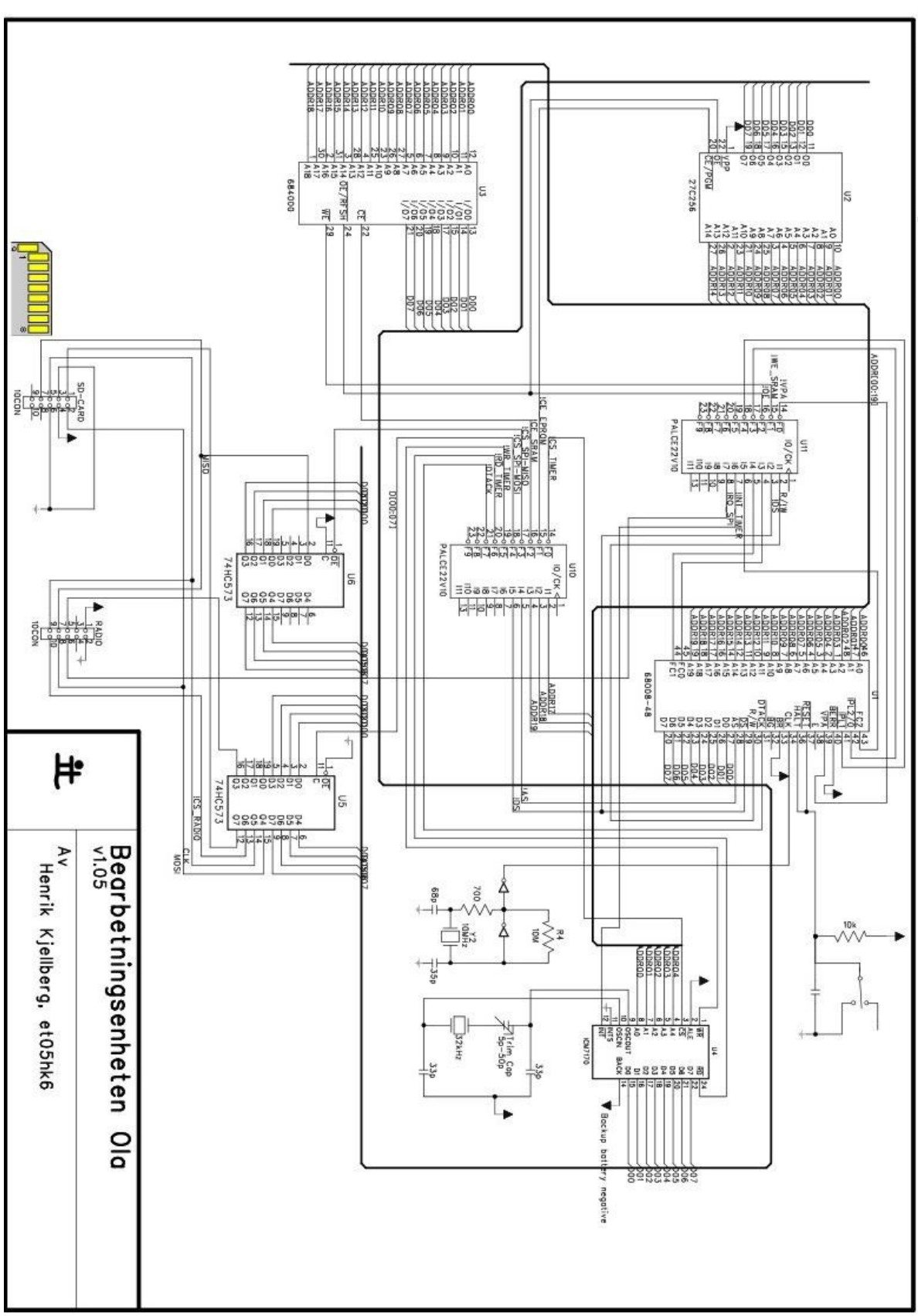


Illustration 1: Calculation Unit, Circuit diagram

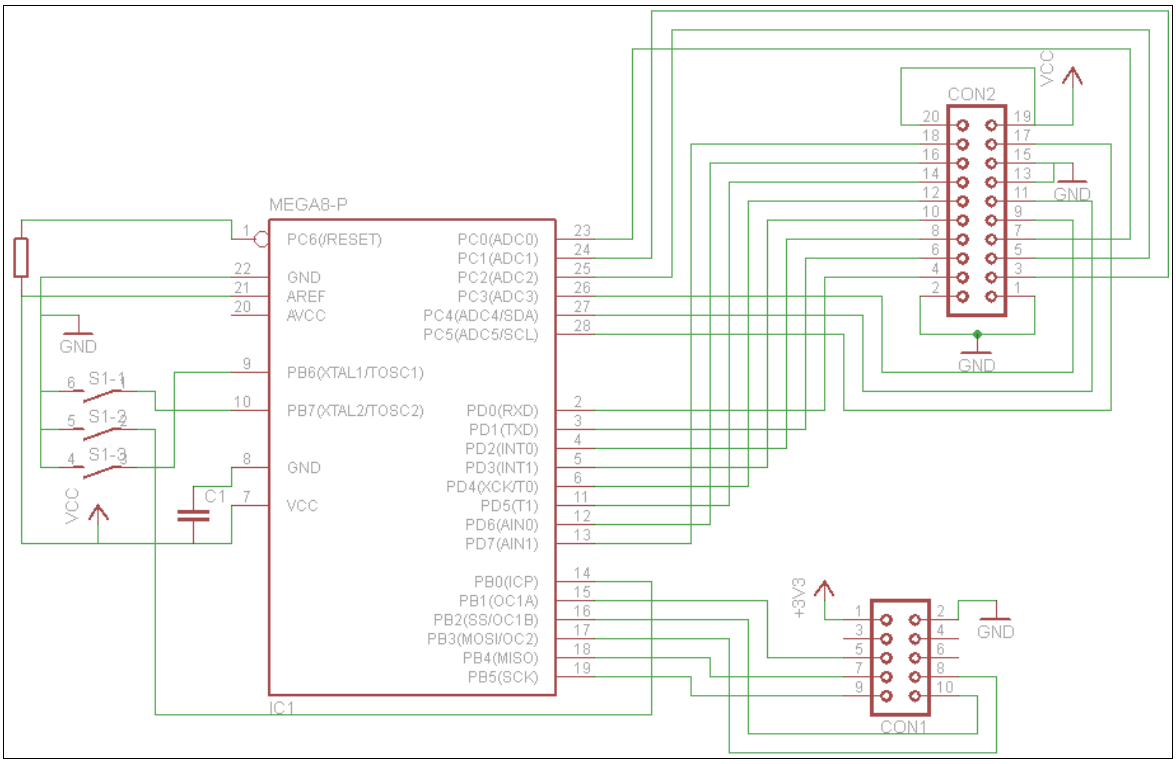
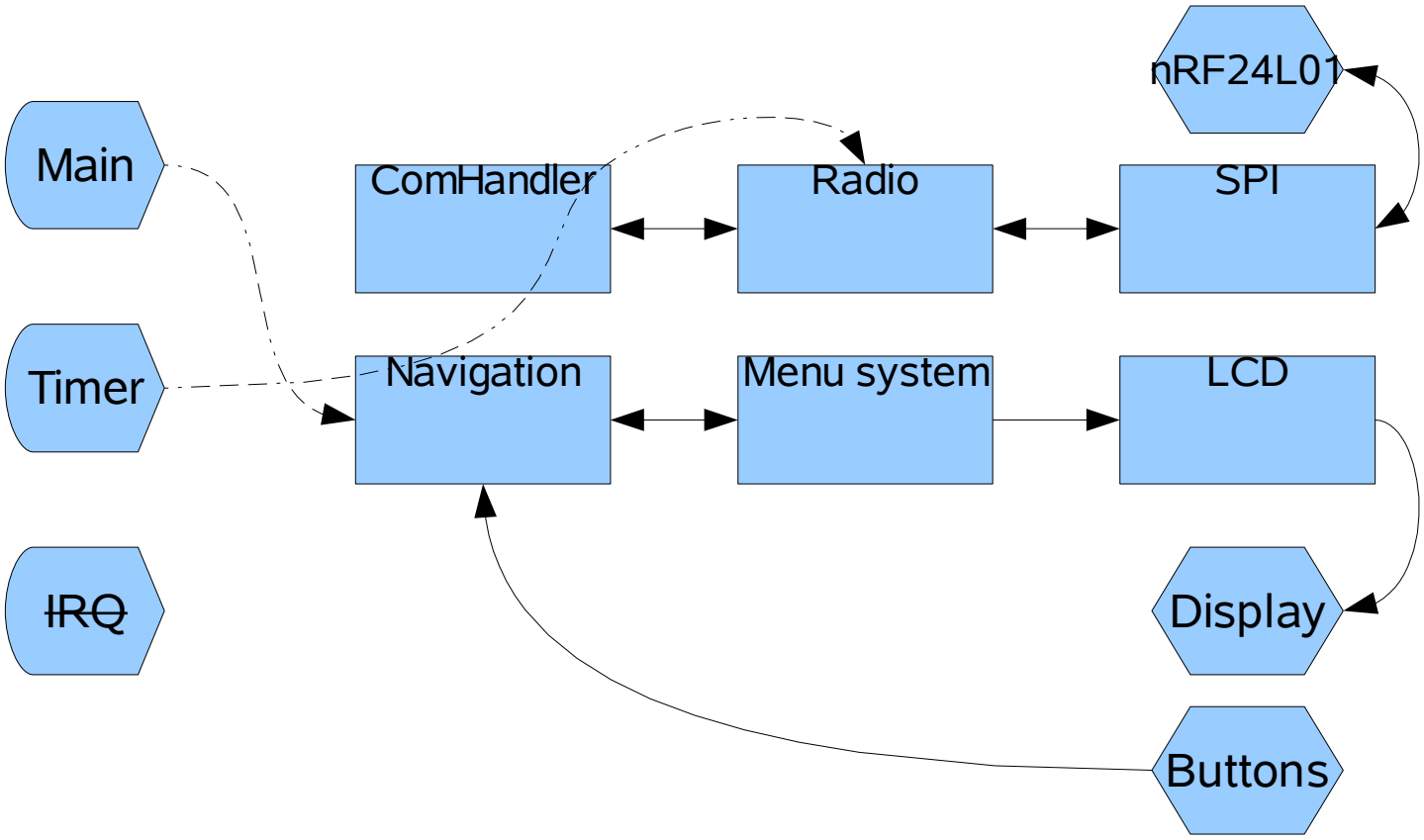
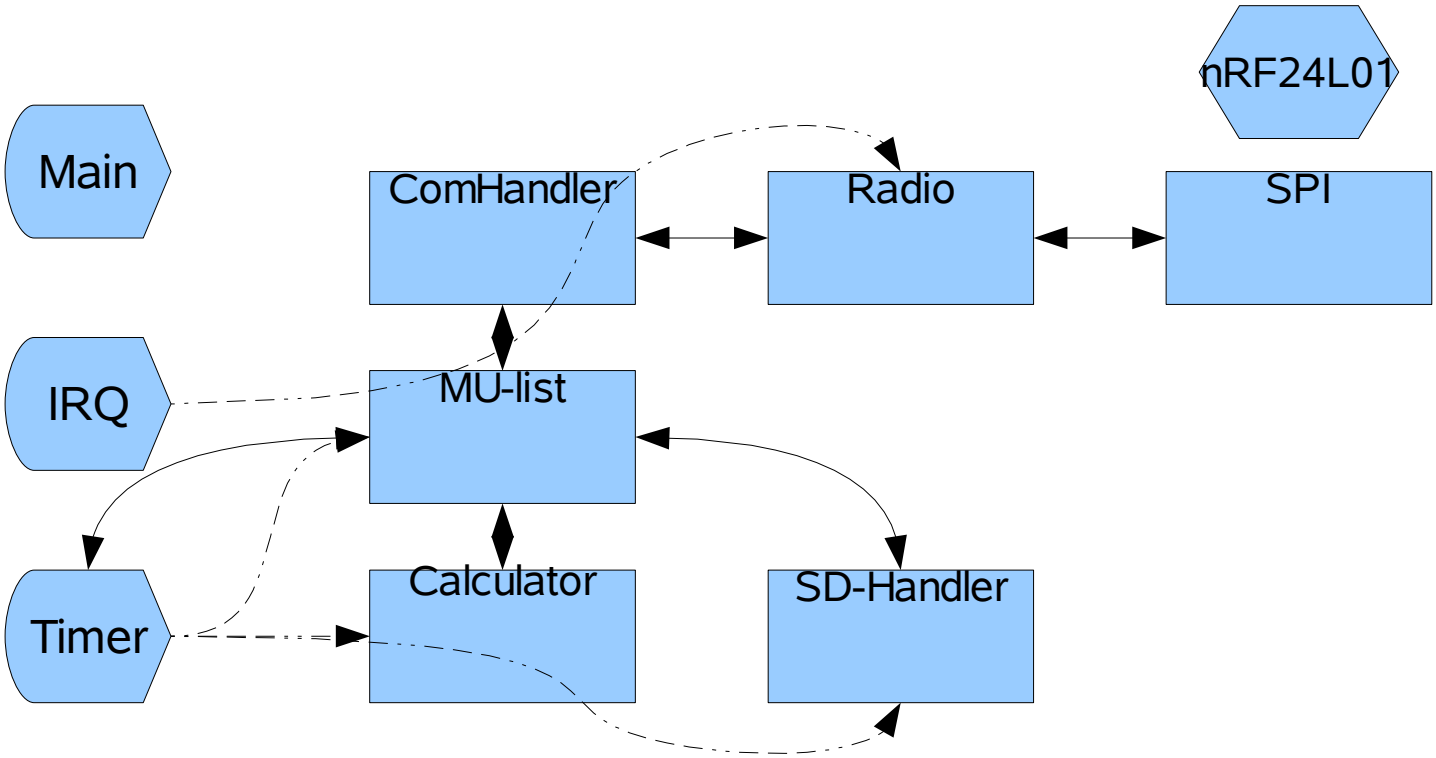


Illustration 2: Display Unit, Circuit diagram

Appendix C



Drawing 1: UML Display Unit



Drawing 2: UML Calculation Unit

References

µALFAT : http://www.alfat.co.uk/ualfat-sd.html

nRF24L01: http://olimex.com/dev/mod-nrf24L.html

Meteorology basics: http://www.bonetweb.com/oskarsson/meteor.html