Low-Power Wakeup Receiver

Semester Thesis

Authors
Cédric Renda, Manuel Tischhauser
Supervisor
Prof. Dr. Heinz Mathis
Assistant Supervisor
Selina Malacarne
Subject
Wireless Communications

HSR Hochschule Für Technik Rapperswil
December 19, 2019

Abstract

Introduction

In company buildings, universities and other similar facilities, it is usual to attach occupancy schedules next to a room entrance. These schedules are most of the time printed on paper, or in more modern buildings, displayed on a screen. Using paper, it is impracticable to take short-term changes into account, since every adjustment needs to be made by printing a new schedule. Hence, screens need to be connected to the power grid, which entails extensive installation work, or powered by a battery, which needs to be replaced from time to time. All these disadvantages could be avoided by using a screen which harvests its energy from the indoor lightning, and is updated via a wireless interface.

Approach

The system is powered by a energy storage system that uses solar cells to harvest energy. This energy will then be stored in a super-capacitor to ensure minimal power leakage. As this specific type of energy harvesting unit cannot provide vast amounts of energy over a longer time period, the whole system fully runs in short bursts. One solution introduced to save energy is by using an E-Paper Display (EPD). The EPD only uses energy when updating its screen but not for maintaining, the image displayed on the screen, which results in a high energy reduction. Furthermore, in addition to the EPD a constantly listening ultra-low power wakeup receiver is used. This will further enhance the system, as it will generate an interruption to enable the power supply of the remaining system only when receiving a specific wakeup sequence. The remaining part then receives data via Bluetooth, which will then display the data and eventually cuts its own power supply to enter a new cycle of low-power listening mode.

Conclusion

This semester thesis was designed to elaborate a sustainable screen in order to display a room's occupancy next to its entrance. Upon analysis of potential solutions, it can be concluded that the outlined energy harvesting does provide enough energy to sustainable run the introduced E-Paper Display (EPD), if up-dating the schedule does not occur too frequently. Moreover, it has been identified that a wakeup receiver is the most suitable tool to support the display. The implementation of a latch circuit makes it possible to switch off the whole system (except for the wakeup receiver) when not in usage. In total the wakeup receiver consumes a maximum of $4.5\,\mu$W during listening mode while the system harvests around $485,52\,\mu$W.

Contents

Abbreviations

ASK Amplitude Shift Keying

BLE Bluetooth Low Energy

CPU Central Processing Unit

EPD E-Paper Display

FIFO First In First Out

FRAM Ferroelectric Random Access Memory

GPIO General Purpose Input/Output

GUI Graphical User Interface

HSR Hochschule für Technik Rapperswil

IIS Fraunhofer-Institut für Integrierte Schaltungen

IoT Internet of Things

OOK On-Off Keying

SPI Serial Peripheral Interface

SRAM Static Random Access Memory

TTL Transistor-Transistor Logic

UART Universal Asynchronous Receiver Transmitter

USB Universal Serial Bus

Chapter 1

Introduction

Today, companies, universities and similar facilities, are using a simple sign next to the entrance of a room to display its occupancy schedule. In more modern buildings, the schedule is presented on a screen; however, most of the buildings still use schedules printed on paper. This approach creates the infeasibility to take spontaneous changes into account, as every minor modifications leads to a newly printed schedule. Additionally, buildings that already use displays have the obstacle of the initial installation of the device, which entails extensive installation work by connecting it to the power grid. One solution could be to replace the power grid by batteries, which would lead to additional costs of the batteries (due to frequent replacement) plus the human labour costs (as employers have to change the batteries). The disadvantages outlined above could be avoided by using a screen (following named "E-Paper Display" or EPD) that harvests its energy from indoor lightening. Additional to this self-sufficient method, the screen would also up-date via a wireless interface, providing the opportunity to take short-term changes into consideration and display a timely room schedule.

In this semester thesis, a device has been developed which will be further outlined in the following six sections. In the first chapter, the task analysis as well as its approach will be evaluated. In the second chapter, the theories around the device are described in more detail. The third chapter, looks at the evaluation of the wakeup receiver as well as the microcontroller. The fourth chapter contains the development of the hardware and software part. The fifth chapter presents the results followed by a conclusion.

1.1 Task analysis

The primary goal of this semester thesis was to create a prototype by combining both benefits of the paper method and the screen powered by battery. Moreover, the aim was to avoid an expensive initial installation by introducing a method that makes it possible to up-date the room schedule via a wireless interface. However, it should still be possible to place the new screen in the desired location equal to the paper schedule. Finally, to guarantee the device would consume the smallest amount power possible, the microcontroller, display and interface should be switched off when not in usage.

1.2 Approach

In order to clarify the application of the terms in the following sections, the term “receiver” is used for the display including all its components. Additionally, the term “transmitter” is used for the computer and its additional hardware. Henceforth, it has to be stated that this does not imply that the communication is uni-directional. The receiver as well as the transmitter should be able to receive and transmit data. It should rather suggest that the general data transfer appears from the computer to the screen.

1.2.1 Transmitter

This end solely consists of a computer, a wireless interface (for the main communication) and the transmitter module (for the wakeup receiver). A short python script enables the user to define which data the receiver should display over a fixed template. The information is transferred to a wireless interface, which transmits it to the receiver end after the receiver is woken up.

1.2.2 Receiver

This part of the prototype has been designed to reduce the power consumption as far as possible. Therefore, ans EPD is used, which has the advantage of being a static display (it maintains its screen image even when disconnected from the power supply). In combination with the EPD a wireless module receives information from the transmitter. It can also response with a feedback about the status. This connection is bi-directional. In order to process the incoming data and displaying it correctly on the screen, a microcontroller is used. All components are completely disconnected from the power supply when not used. The power is provided by a super capacitor, which is charged by solar cells. When the device is switched off, the only component that actively consumes power is the wakeup receiver. This module is constantly in listening mode and waits for a wakeup event. If the wakeup receiver is given such an event, it activates a self-holding circuit resulting in the connection to the power supply of all other components. Now data can be exchanged through a different channel, established by the wireless module. After the data is processed and the display is refreshed, the microcontroller deactivates the self-holding and therefore disconnects all components except the wakeup receiver from the power source.

Chapter 2

Theory

2.1 E-Paper Display

There are several different technologies which are applied in E-Paper Display. Since the developed prototype uses a microencapsulated electrophoretic display, this section only describes this specific implementation.

A film consisting of microscopic capsules is coated onto a backplane, which basically is a matrix of electrodes. On top of that comes a common electrode, called the frontplane. This setup is shown in Figure 2.1.

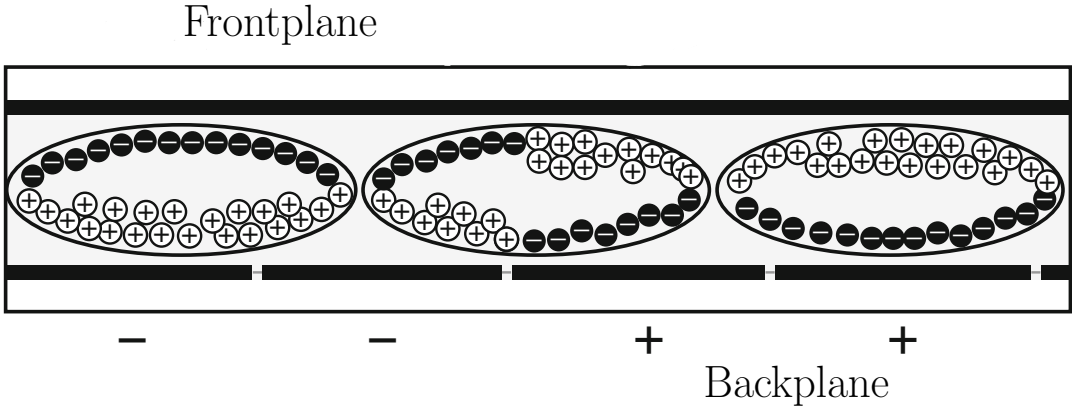


Figure 2.1: Capsules between to electrodes (taken from [1] with adjustments).

Each capsule in the film itself contains a transparent fluid and particles of two opposite charges. One type of particle scatters the light while the other one absorbs it. The particles now begin to move in opposite directions through the fluid when exposed two an electrical field. The pixel (defined by the electrode size) appears white, if the scattering particles are near the frontplane and black, if otherwise. If no voltage is applied to the electrodes, the particles maintain their last position. This makes it possible, to switch off the power supply when no image update is needed. This is the reason why the power consumption can be strongly reduced compared to other types of display [1].

Figure 2.2 shows an e-paper display with 250x magnification.

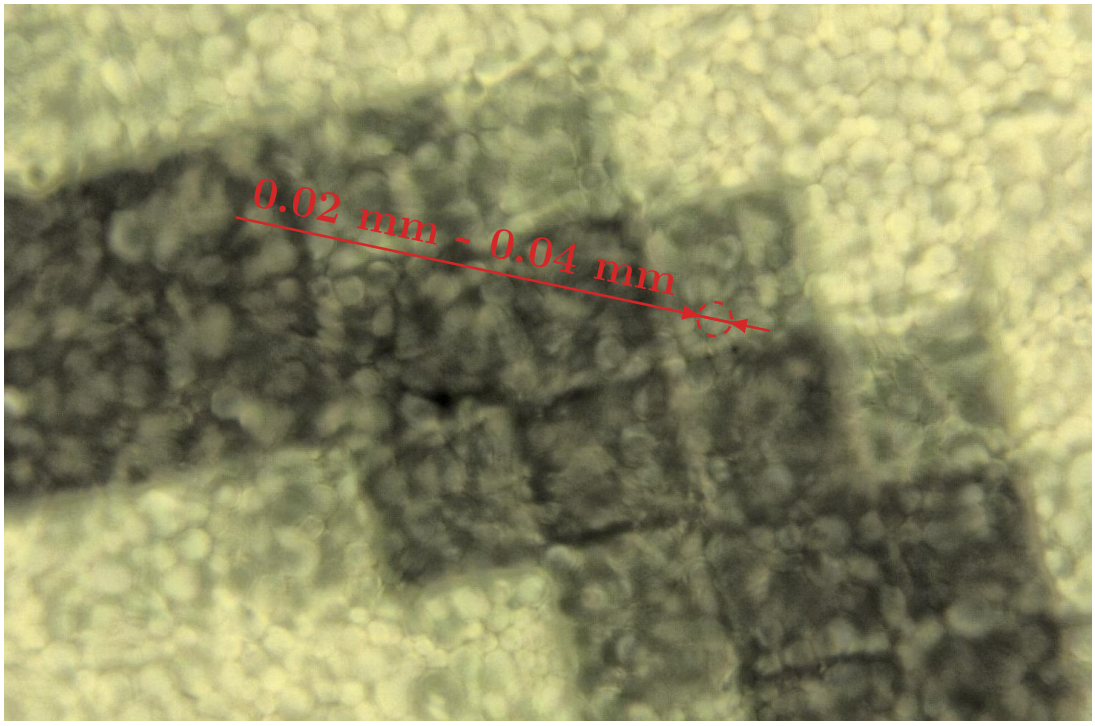


Figure 2.2: E-Paper Display under microscope with 250x magnification.

2.2 Wakeup Receiver

In order to reduce power with a conventional receiver it has to be programmed to stay in sleeping mode until it is used again. However, it has to check whether any data was sent, therefore, it has to wake-up periodically to check if any notifications were sent. This duty cycle is a trade-off between power consumption and the respective response time. Conclusively, if the period between the notification checks is longer, the receiver is in an extended sleeping mode using less power, but simultaneously potentially missing out on a new notification, as the data can only be received in the running mode. The wakeup receiver now allows the device to be constantly in the listening mode, while consuming just a small amount of energy. Additionally to the system, the actual microcontroller, which coordinates the data transmission and other tasks stays shut down with only the wakeup receiver in listening mode. After receiving a defined pattern over this channel, the wakeup receiver generates an interrupt resulting in waking up the microcontroller. Subsequently, it establishes now a channel over a different wireless module, or execute another task. When finished, the microcontroller puts itself and all other modules (except the wakeup receiver) in sleep mode again. Figure 2.3 shows a comparison of both the conventional approach and the solution with the wakeup receiver. One could argue, that the wakeup receiver module consumes in general more power, than the microcontroller in sleep mode. However, since the microcontroller only wakes up when a task needs to be done, the overall energy consumption (area underneath the curve) is going to be smaller if the occurring wakeup event comes infrequently or over longer periods of time. The response time on the other hand can be kept in the microsecond range.

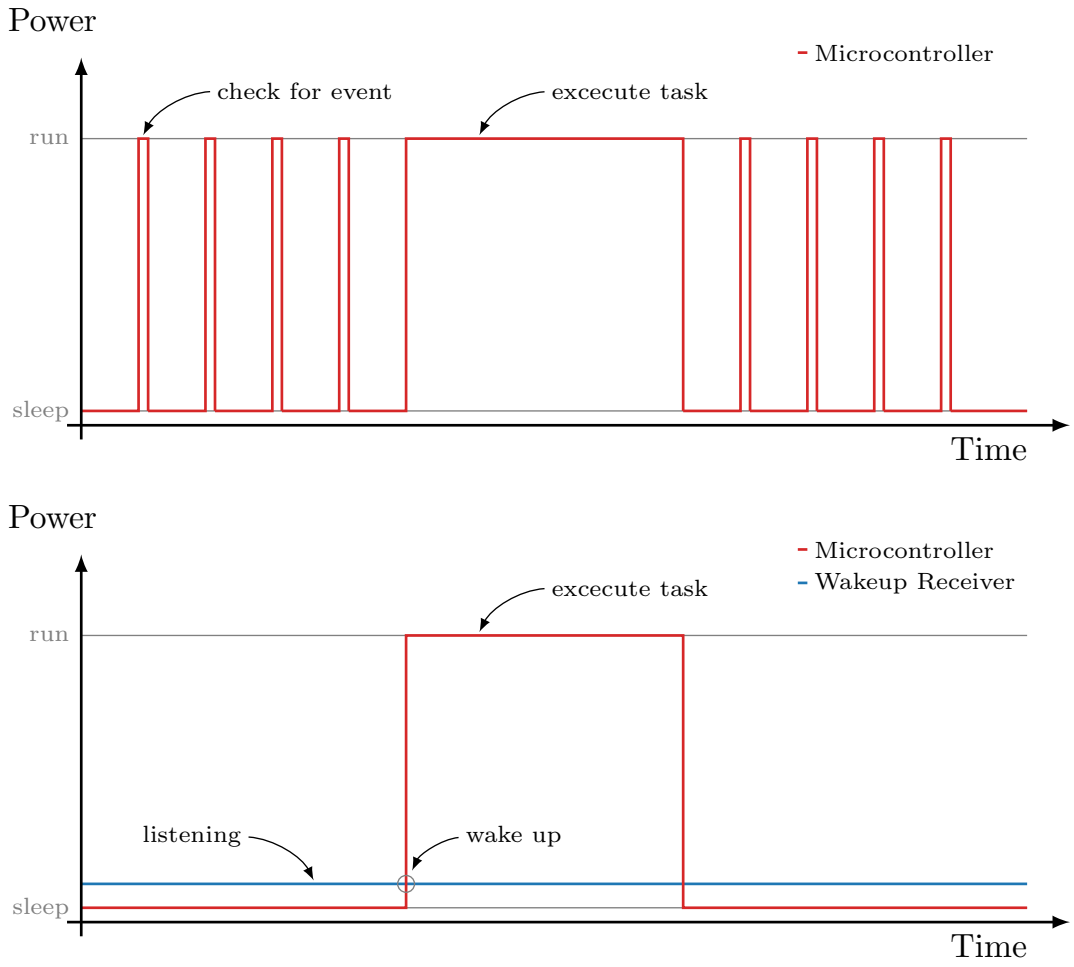


Figure 2.3: Top: Microcontroller checks periodically for incoming data. Bottom: Constantly listening wakeup receiver.

2.3 Drawing graphics on a graphics display

If a matrix style display is used with a microcontroller, there are a lot of different ways to draw graphical elements. This section covers only the ways used in this project.

Matrix displays

Since most displays produced have a rectangular shape, the amount of pixel they hold is calculated like the surface of an rectangular. The number of pixels on both axis are multiplied and we get the total amount of pixels we have to manipulate. So it is obvious that the memory size of a picture explodes with its resolution. A display with a size of 1200×825 pixels and a colour depth of 24 bit serves as an example. This resolution results in amount of $1200 \cdot 825 = 990'000$ pixels and thus $1200 \cdot 825 \cdot 24 = 23'760'000$ bits. This is a huge amount of data to process with a standard microcontroller. To handle the data transfer between memory and display, special parallel driver chips are used. These drivers are mapping the right memory location to the desired pixel on the display. More expensive driver chips have built in memory. To access it from a host CPU, parallel or serial bus interfaces are used. If a display is rectangular, the information with the pixel value can be ordered in a matrix. A good option to store an image is therefore an array.

The advantage of an array is the easy access and the memory allocation given in various programming languages. In C for example, memory of an array has to be allocated at consecutive addresses [2].

Double buffering

To write an image to a display, two separate memory accesses are carried out. The first writes the image into the buffer, the second takes this data and displays on screen. By using two buffers, it is possible to accelerate this process. This is shown in Figure 2.4.

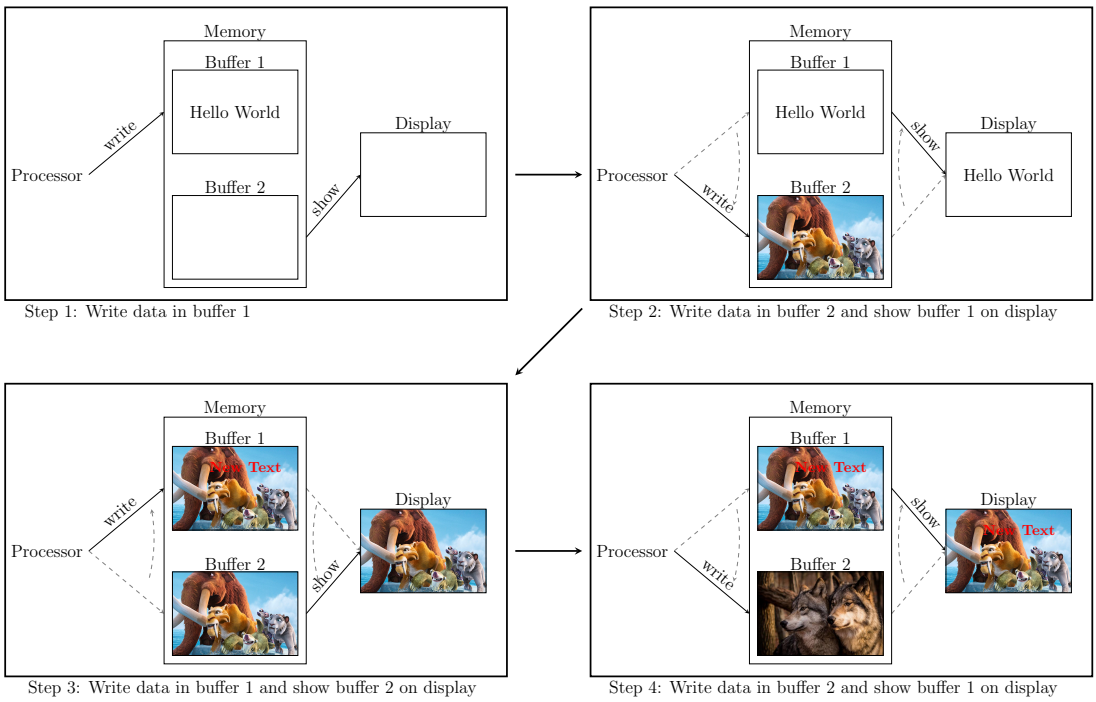


Figure 2.4: Double buffering.

While the new data is written in to one buffer, the screen displays the data located in the other buffer at the same time.

Compressed fonts

To write a specific character to the display, the controller needs to know how the font is structured. Figure 2.5 shows how the character A could look. The character is stored this way into the memory and can be accessed just like a look-up table. If fonts are stored with a 12×7 pixel mask, every character occupies 12 bytes memory.

To display characters in color, 8 more bits for the color depth are needed. this is visualized in Figure 2.6

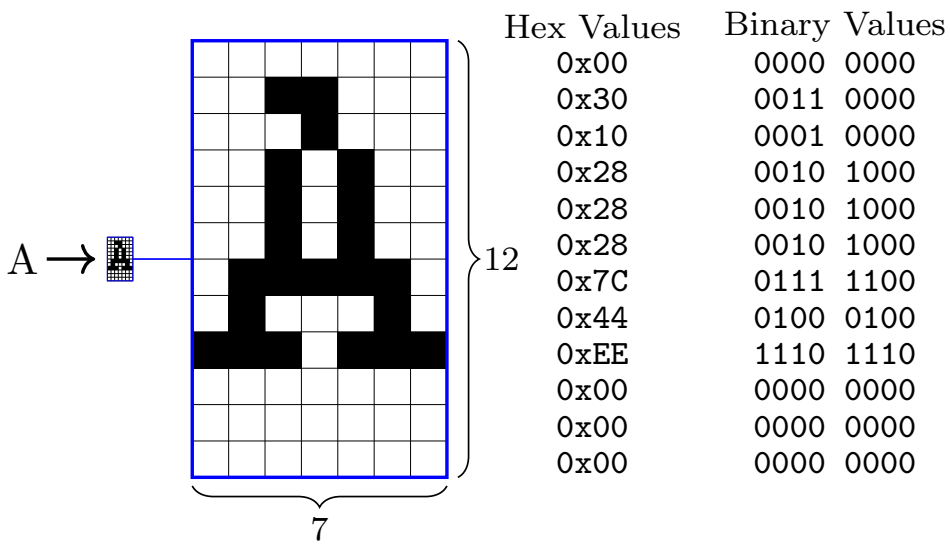
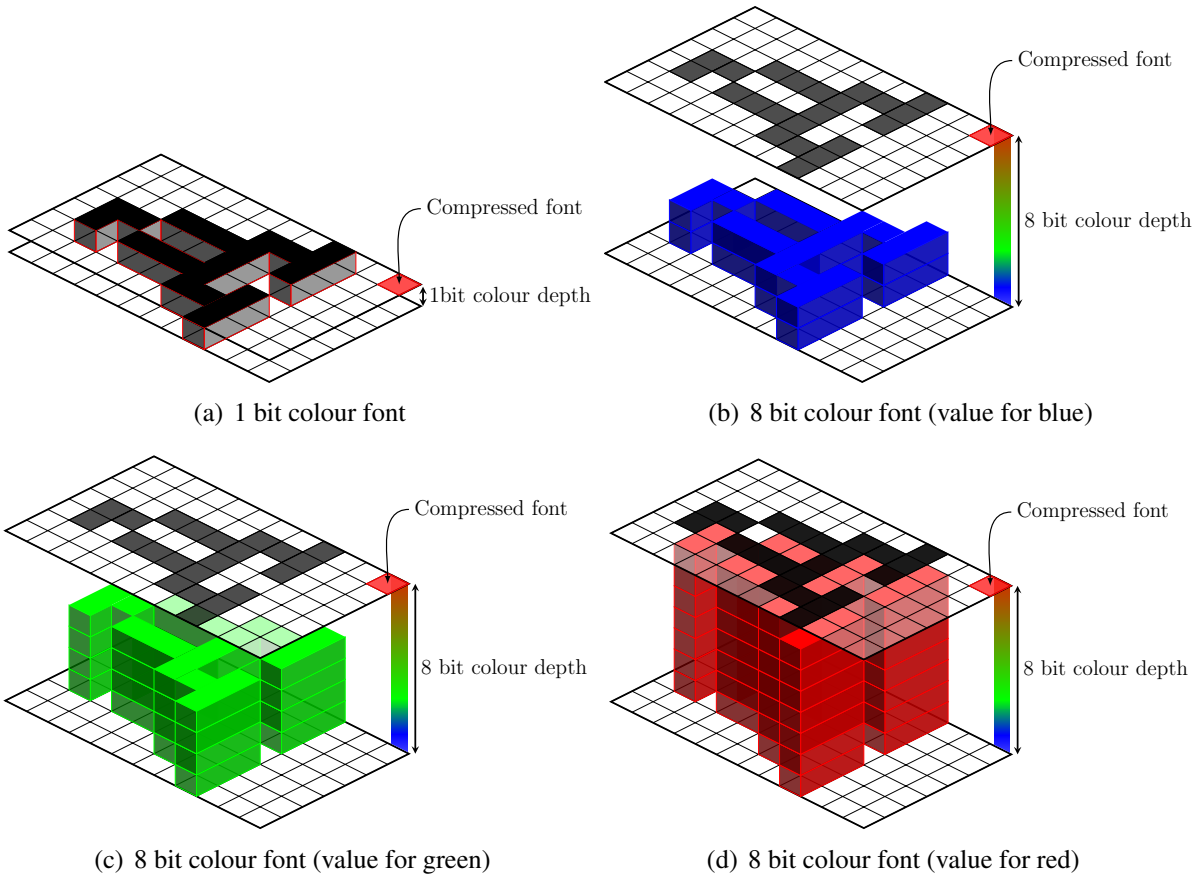


Figure 2.5: Creation of a simple seven to twelve pixel sized font.



(a) 1 bit colour font

(b) 8 bit colour font (value for blue)

(c) 8 bit colour font (value for green)

(d) 8 bit colour font (value for red)

Figure 2.6: 12×7 pixel font mask displayed with 1- and 8bit colour depth

Chapter 3

Evaluation

3.1 Wakeup Receiver

In this semester thesis, two different implementations of the wakeup receiver technology were compared. The AS3933 from AMS and the RFicient from the Fraunhofer-Institut für Integrierte Schaltungen (IIS), which kindly provided an evaluation kit before the actual release of the product.

3.1.1 AS3933

The AS3933 is a low frequency wakeup receiver, which uses ASK to modulate a carrier frequency between 15-150 kHz. The transmitter sends a manchester encoded, programmable wakeup pattern of length 16 or 32 bit. If this pattern is detected on the receiver end, a wakeup interrupt is generated. It is also possible to disable the pattern decoder to run the chip in a frequency detection mode, where a wakeup interrupt is generated as soon as the specified frequency is received. More important features on the receiver end are:

- Receiver sensitivity $80\,\mu V_{RMS}$
- Current consumption in 3-channel listening mode $2.3\,\mu A$
- Operating supply range $2.4\,V - 3.6\,V$
- Three antennas (enables 3D detection)
- Channels individually selective to reduce power consumption

The low power consumption makes it possible to run the receiver in listening mode below $8.3\,\mu W$[3].

The demo kit comes with a GUI, which enables the user to set the parameters as desired and address the registers directly. The range of the receiver of about 6 m as first measurement turns out to be very limited. Even with a 3db sensitivity boost on the receiver side, it is only possible to detect wakeup events from a distance of 9 m. The environment (indoor, outdoor) seems to make no observable difference.

As a result, the limited range makes the AS3933 unusable for the prototype.

3.1.2 RFicient

The Rficient from the IIS uses OOK to modulate a 868 MHz signal. It can either run in pure wakeup mode, where the receiver generates an interrupt as soon as a code is received or in a selective mode, where a 16 bit wakeup preamble needs to match the receiver. After the preamble is detected, the data rate can be changed to transfer more bits, which can be sent over an SPI-bus to a connected device. It is possible to transmit data bits after the actual wakeup in this way. Data rates can be set in a range between 256 bp/s − 32 kbp/s. The most important features are:

- Receiver sensitivity -80 dBm
- Energy consumption 3 μA at 1.5 V (data rate 1 kbit/s)
- Unidirectional data transfer possible

Therefore, the power consumption in listening mode (data rate = 1 kb/s) is 4.5 μW.

Just as the AS3933, the RFicient demo kit comes with a GUI, which enables the user to set the important parameters and access the registers [4]. First measurements showed that the range of the RFicient is far higher than the range of the AS3933. It is therefore used in the prototype.

3.2 Microcontroller

To process incoming data and write it to the display, some kind of microcontroller is needed. A derivate of the MSP430 and one of the STM32 were looked at more closely.

3.2.1 MSP430

The MSP430FR6989 was up for selection because it is specifically developed for ultra-low-power applications. It's main features are:

- 16-bit architecture with up to 16 MHz clock
- Supply voltage range from 1.8 V to 3.6 V
- Approximately 100 μA/MHz current consumption in active mode
- 128 KB Ferroelectric Random Access Memory (FRAM)

The display driver chip can be controlled over SPI [5].

During development, it turned out to be very useful to store a complete image. The required memory to do this, assuming 16 grey levels and a 1200×825 size display is:

$$M = 1200 \cdot 825 \cdot 4\,\text{bit} = 495'000\,\text{B} < 484\,\text{kB}.$$

Therefore is the FRAM not sufficient. This is the main reason, why a different microcontroller is needed.

3.2.2 STM32

The STM32L4R5ZI by STMircoelectronics was just like the MSP430 developed for low-power applications. The most important features are:

- 32-bit Cortex-M4 CPU
- Several clock sources between 4 and 48 MHz
- Maximum clock up to 120 MHz
- 1.71 V - 3.6 V input voltage supply
- 110 μA/MHz current consumption in run mode
- 2 MB Flash and 640 KB SRAM

The display can again be controlled over SPI. Additional to the SPI, an integrated LCD-TFT controller unit could be used to control the display in a parallel mode [6].

The STM32 is in general more flexible and, most importantly, has enough memory for a complete picture. That it consumes slightly more power than the MSP430 should not be to much of a problem, since it is only in run mode for a few seconds, before being completely switched off again. The standby mode of the controller is not used and therefore of no interest.

Chapter 4

Development

4.1 Hardware

Figure 4.1 shows the receiver schematics. Every component is described in more detail in this section.

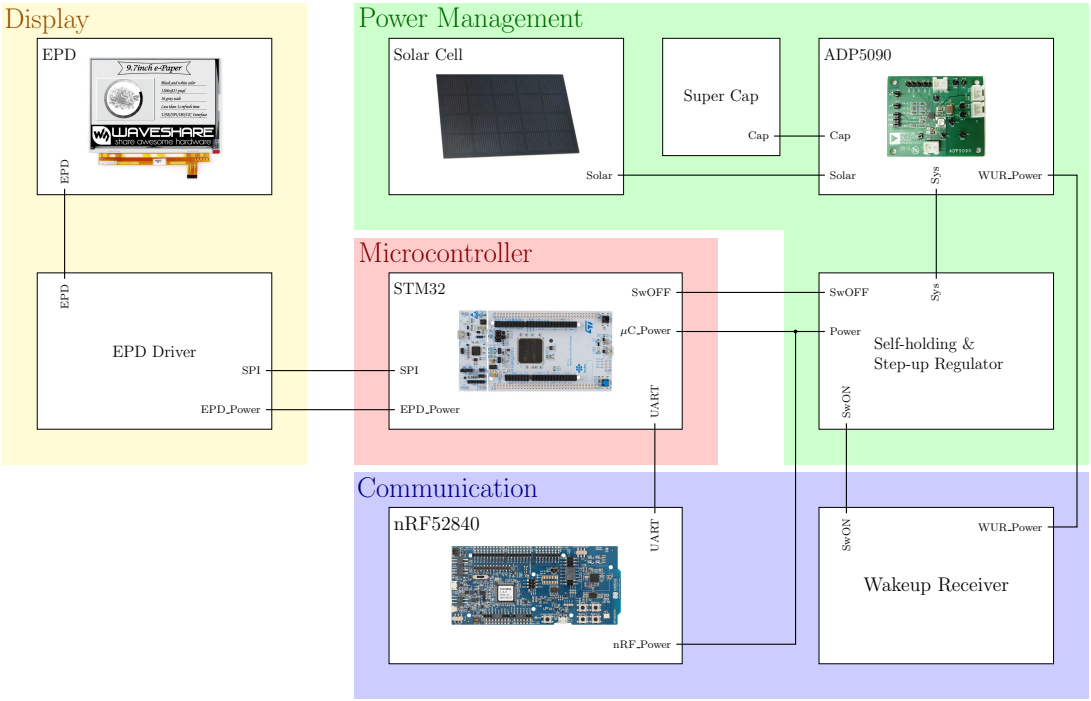


Figure 4.1: Receiver schematics.

4.1.1 Display

An 9.7 inch (202.8 mm × 139.425 mm) e-paper display is used which adopts the microencapsulated electrophoretic display technology just as described in Chapter 2. It has a 1200 × 825 resolution with 16 grey scales. The IT8951, which is the display driver used on the evaluation board, drives the EPD. It can be accessed from a microcontroller with a SPI-bus [7].

4.1.2 Communication

RFicient

Transmitter As previously stated, the Rficient demo kit comes with a GUI. Is the transmitter module connected, simple steps make it possible to set the preamble and payload data rate. The payload itself can partly serve as an ID and as additional transmitted data. The power of the transmission can also be set between -30 dBm and 10 dBm. The transmitter-GUI is shown in Figure 4.2. To test if any data is received, it is helpful to activate the infinite loop (top right in

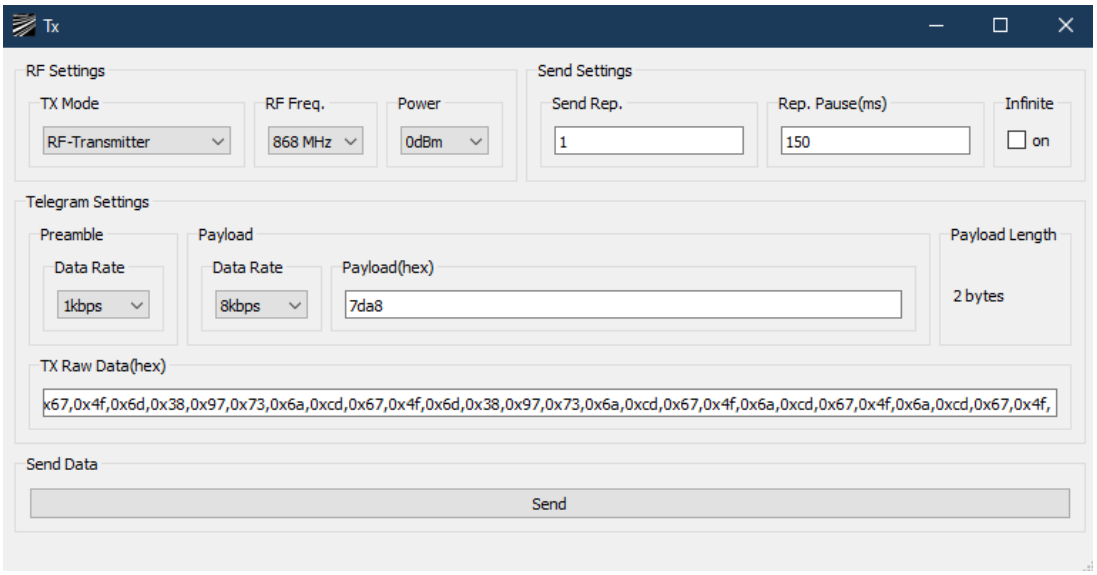


Figure 4.2: Transmitter GUI.

Figure 4.2). In this case, the defined data is sent repeatedly with an adjustable period.

Receiver In order to receive the data, the setting of the receiver module has to match the transmitter. The provided receiver-GUI makes this step very simple. As visible in Figure 4.3, one can enable the four possible interrupt sources: CodeA/B, ID, FIFO length and FIFO overflow. In the finished product, it should be possible to wake up the receivers individually. For this reason, only the ID as interrupt source is of interest. If an interrupt is generated, the RFicient sets a GPIO-pin to high, and transfers the received data to a FIFO-buffer. This data can be passed to a microcontroller over an SPI-bus. For the purpose of this thesis, only the GPIO high level is used to activate a self-holding circuit.

Tests in a realistic environment To test the RFicient in more detail, some indoor measurements were made. As test environment served several buildings of the Hochschule für Technik Rapperswil (HSR), since this type of facility represents a realistic condition for the developed schedule.

The purpose of the first test was to measure the achievable range. Figure 4.4 shows the hallway where this test was executed. With a transmission power of 0 dBm (1 mW) it is possible to receive interrupts over a 50 m distance (line of sight). Is the transmission power increased to 10 dBm (10 mW), interrupts can be sent over a distance up to 80 m. In this environment, the

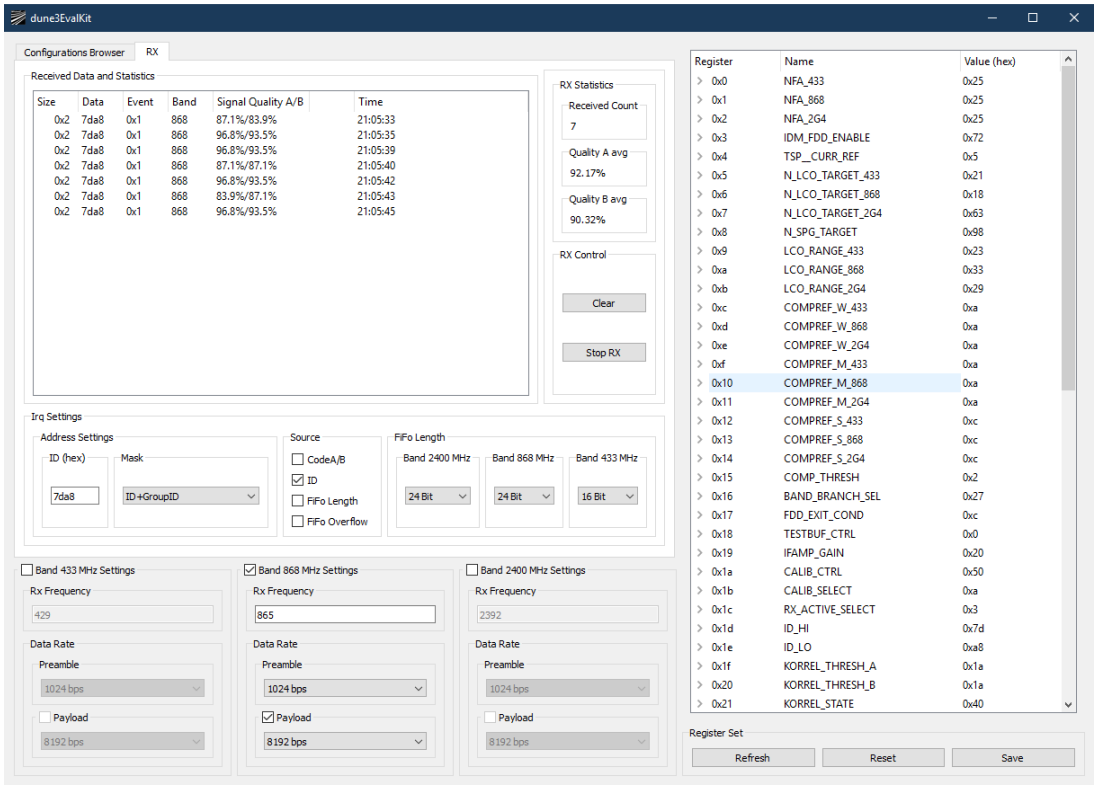


Figure 4.3: Receiver GUI.

Rficient has enough range. However, there is the possibility that reflections off the walls, floor and ceiling had an influence on this result.

Due to this, a second test was performed to check how the RFicient penetrates obstacles. As test environment served this time the HSR builing 8, as depicted in Figure 4.5. An attempt was made to send wakeup signals through the ceiling. To avoid reflections, the RFicient was placed in such a way that the zero point of the antenna pointed directly at the opposite walls. With a transmisson power of 0 dBm, two ceilings can be penetrated. Is the power again increased to 10 dBm, the signal reached through three ceilings.

These measurements further confirmed that the RFicient is suitable for the use-case of this semester thesis.

nRF52480

A nRF52580 from nordic semiconductor serves as the wireless module. It is used because it supports multiple protocols like Thread, Zigbee and, more important, Bluetooth 5, which includes Bluetooth Low Energy (BLE)[8]. Since BLE allows mesh networking and is applied in many existing devices, such as smartphones, the connection to exchange data is established over this protocol. This way, the developed prototype could later be extended to a mesh network.

Two nRF52480 development kits are used, one on the transmitter and one on the receiver end. Data can be transferred from a computer with a USB-TTL-adapter to the UART-interface of the transmitter development kit. This data is then sent over BLE to the receiver kit and passed to the microcontroller again over UART. In the same way, data can be transferred from the receiver to the transmitter kit. The data channel is therefore bi-directional.

Figure 4.4: Test environment 1 (HSR building 1).

4.1.3 Power Management

Since the screen should be self-sustaining, some sort of energy-harvesting unit is needed. It was apparent to choose light as the energy source, since the screen will be used in rooms, that are artificially illuminated most of the time. The energy obtained by solar cells is converted to a suitable voltage, using a ADP5090 chip. This way, a super-capacitor, which is used as an energy storage device, is charged.

Solar cell

The solar-cell AM-1522 by Panasonic is used. One panel has an area of 55.0 mm × 40.5 mm and delivers up to 57.8 μA when operating at an optimal voltage of 2.1 V, provided an illuminance of 200 lux. To keep a reasonable display to panel ratio, four cells where parallelly connected, which corresponds to an area of ca. 89.1 cm^2 (Display area $\approx$ 283 cm^2). Therefore, the solar cells should provide a power of

$$P = U \cdot I = 4 \cdot 57.8\,\mu\text{A} \cdot 2.1\ \text{V} = 485.52\,\mu\text{W}, \tag{4.1}$$

given a 200 lux illuminance [9].

ADP5090

The ADP5090 from Analog Devices is used to manage the energy harvesting process. This boost regulator makes it possible to charge storage elements, such as rechargeable batteries and super capacitors with the input dc-power provided by the PV-cell. Utilized features are:

- Maximum power point tracking
- Efficiency up to 90%
- Input voltage V_{IN} from 80 mV to 3.3 V

Figure 4.5: Test environment 2 (HSR building 8).

- Programmable voltage range (2.2 V to 5.2 V) for the storage element

To prevent the storage element from overdischarging, the ADP5090 enables the user to set a maximal Voltage with resistors:

$$V_{\text{BAT\_TERM}} = \frac{3}{2} V_{\text{REF}} \left(1 + \frac{R_{\text{TERM1}}}{R_{\text{TERM2}}}\right). \tag{4.2}$$

A similar procedure is applied to set a minimal voltage:

$$V_{\text{BAT\_SD}} = V_{\text{REF}} \left(1 + \frac{R_{\text{SD1}}}{R_{\text{SD2}}}\right). \tag{4.3}$$

While discharging, the ADP5090 will switch off the output V_{SYS} if $V_{\text{BAT\_SD}}$ is reached. This prevents the storage element from overdischarging. The output voltage V_{SYS}, where the load is attached, will therefore always stay in this programmed range ($V_{\text{BAT\_SD}} \leq V_{\text{SYS}} \leq V_{\text{BAT\_TERM}}$) [10].

For this prototype, the evaluation board for the ADP5090 was used, where the internal reference voltage (V_{REF} in (4.2) and (4.3)) is 1.21 V [11].

Super capacitor

As energy storage, a super capacitor from Taiyo Yuden (LIC1235RS3R8406) has proven to be suitable. This is a 40 F cylinder type lithium ion capacitor. The operating voltage range is between 2.2 V and 3.8 V. Discharging the capacitor lower than 2.2 V causes shorter lifetime and higher leakage current. The same unwanted behaviour occurs when charging the capacitor over 3.8 V [12].

Combined test

To test the behaviour of the power management, supercapacitor and solar cells, a couple of measurements are executed. Figure 4.6 shows the schematics of the test setup.

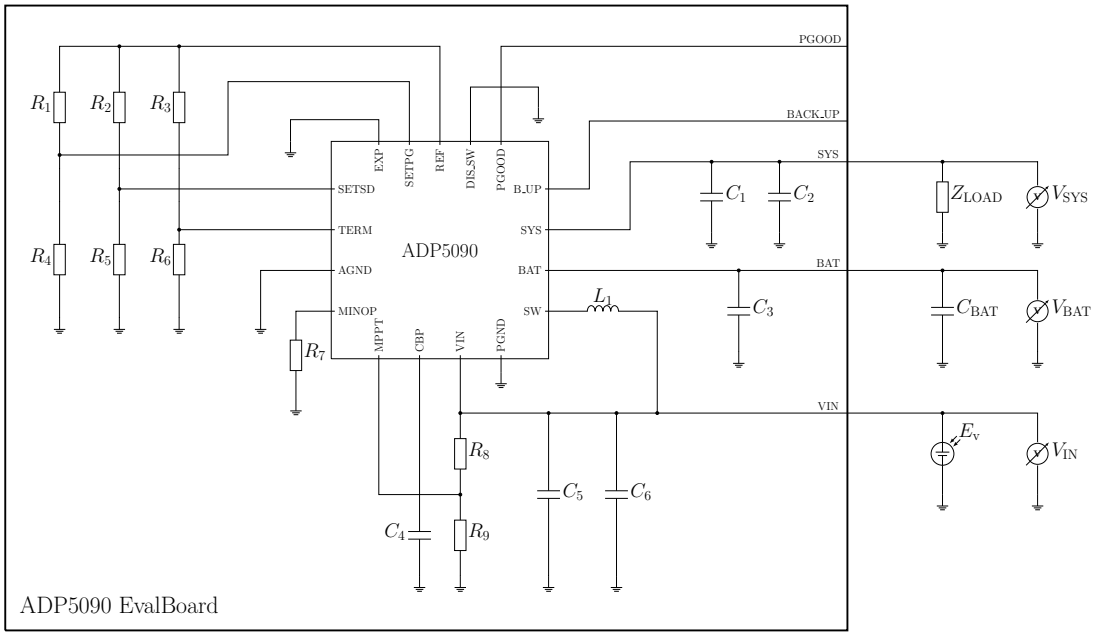


Figure 4.6: Schematics of the test setup.

To carry out these measurements, it is first necessary to adjust the minimal and maximal voltage of the ADP5090. The nRF58240 accepts supply voltages between 1.6 V up to 5.5 V [13]. The STM32 on the other side is less flexible with an input voltage range of 1.71 V to 3.6 V [6]. As stated in the section above, the super capacitors operating voltage is between 2.2 V and 3.8 V. Hence it seems reasonable, to set $V_{\mathrm{BAT\_TERM}} \leq 3.6\,\mathrm{V}$ and $V_{\mathrm{BAT\_SD}} \geq 2.2\,\mathrm{V}$, to satisfy all of these three elements. In order to do this, the four resistors have to be chosen as $R_3 = 4.3\,\mathrm{M\Omega}$, $R_6 = 4.7\,\mathrm{M\Omega}$, $R_2 = 4.3\,\mathrm{M\Omega}$ and $R_3 = 5.1\,\mathrm{M\Omega}$. Inserted in the equation (4.2) and (4.3) we get

$$V_{\mathrm{BAT\_TERM}} = \frac{3}{2} 1.21\,\mathrm{V}\left(1 + \frac{4.3\,\mathrm{M\Omega}}{4.7\,\mathrm{M\Omega}}\right) \approx 3.48\,\mathrm{V} \tag{4.4}$$

and

$$V_{\mathrm{BAT\_SD}} = 1.21\,\mathrm{V}\left(1 + \frac{4.3\,\mathrm{M\Omega}}{5.1\,\mathrm{M\Omega}}\right) \approx 2.23\,\mathrm{V}. \tag{4.5}$$

While testing, the input voltage from the solar cells (V_{IN}), the supercap's voltage (V_{BAT}) and the output voltage (V_{SYS}) are tracked. Additionally, the illuminance (E_{v}) near the PV-cells is recorded as indicated in figure 4.6.

The purpose of the first test is to check if the ADP5090 converts V_{IN} to a voltage $\leq V_{\mathrm{BAT\_TERM}}$. The measurements where taken over a couple hours and are plotted in Figure 4.7.

No load was connected to the output ($Z_{\mathrm{LOAD}} \to \infty$), which is the reason why V_{SYS} is overlapped by V_{BAT}. It can be seen that between 17:00 and 23:00, the super capacitor was being charged and that the ADP5090 controls the voltage V_{BAT} as expected to the adjusted maximum voltage $V_{\mathrm{BAT\_TERM}}$.

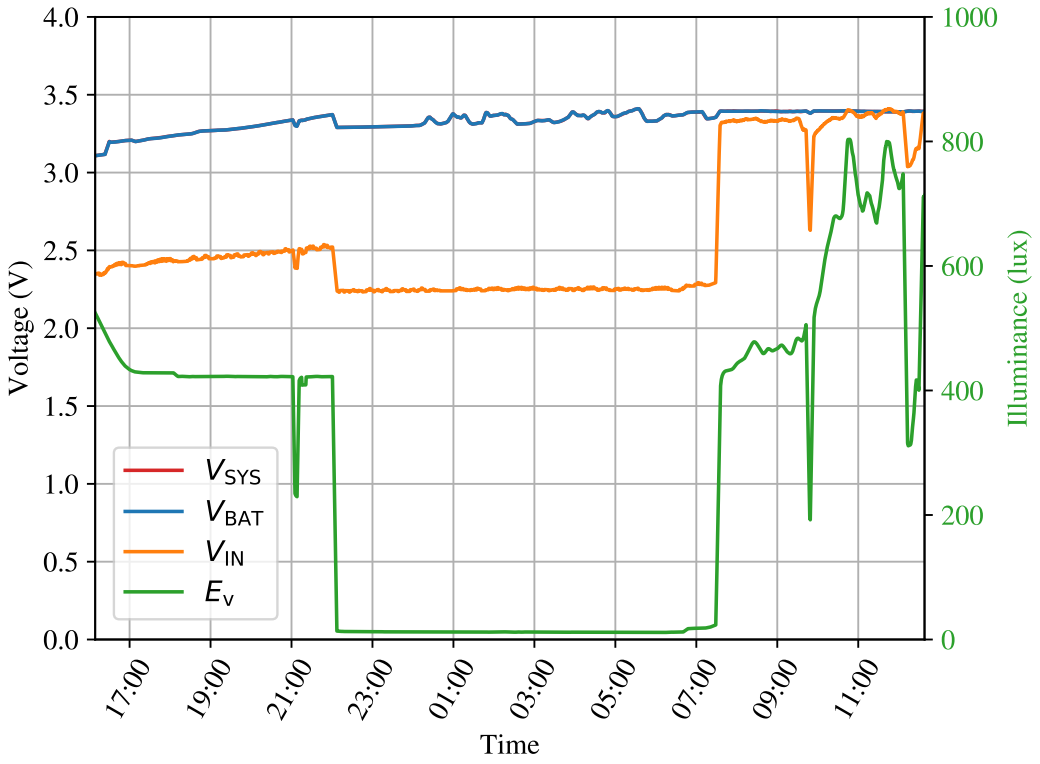


Figure 4.7: Charging behaviour.

The second test should simulate the discharging when a load is connected, after the capacitor was fully charged. It was necessary to estimate the consumed power by the electronic components of the prototype. A rough measurement with a power analyser showed that the microcontroller and the e-paper display together draw at its peak about 240 mA when connected to 5 V. The nrf52840 on the other hand, only consumes 6 mA with a 3 V source. Thus the expected consumed power at is's peak is:

$$P_e = 5\,\mathrm{V} \cdot 0.24\,\mathrm{A} + 3\,\mathrm{V} \cdot 0.006\,\mathrm{A} = 1.218\,\mathrm{W}. \tag{4.6}$$

If $Z_{\mathrm{LOAD}} = 10\,\Omega$ the load draws currents between 0.223 A and 0.348 A, which again lead to a power consumption that should approximately match the power consumption of the finished prototype. Furthermore, the solar cells where covered to observe the discharging without interference of additionally charging behaviour. Figure 4.8 shows the result.

As soon as the load is connected (after 25), V_{SYS} and V_{BAT} first drop by almost 1 V and steadily decrease subsequently. After ca. 100 s, V_{BAT} reached the value of $V_{\mathrm{BAT\_SD}}$ and the ADP5090 switches the output off (V_{SYS} drops to 0) to prevent the capacitor from overdischarging. The output now stays switched off, until V_{IN} again supplies energy, and $V_{\mathrm{BAT}} \leq V_{\mathrm{BAT\_SD}}$. It can also clearly be seen, that after 160 s, the ADP5090 controls V_{IN} to ca. 2.1 V. Recall that this is the optimal power point of the solar cell.

Power Latch

Considering the usage of an wakeup receiver, implementing a latch circuit to en-/disable the power supply is a suitable option. The first version of latch circuit designed has two control

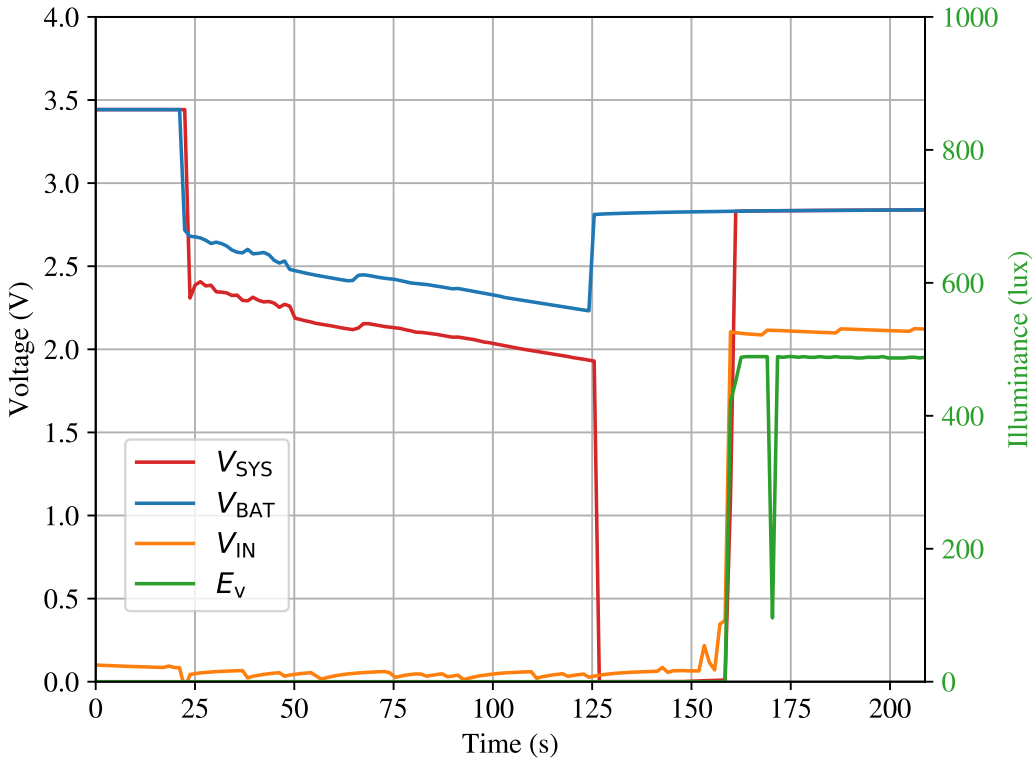
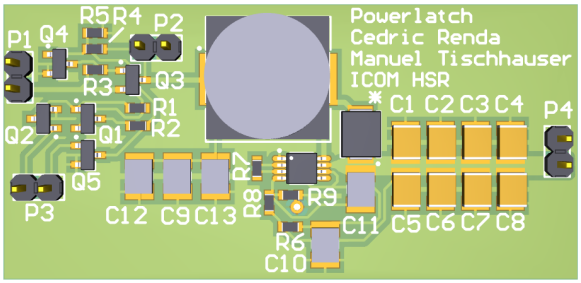


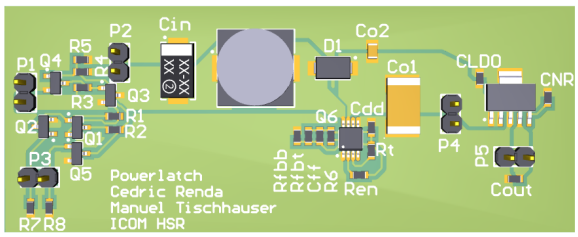
Figure 4.8: Discharging behaviour.

inputs. A short voltage impulse on one of these inputs en-/disables the power supply for everything except the wakeup receiver and the ADP5090.

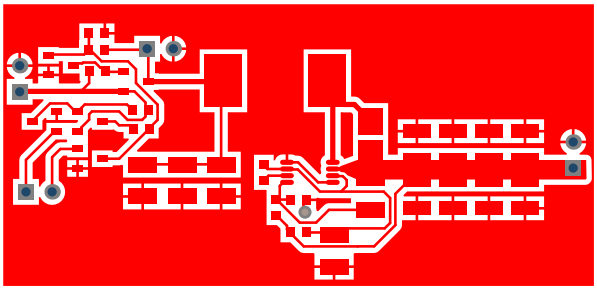
Since the driver board of the EPD requires a stable input voltage to run reliable, a DC-DC converter is needed to convert the minimum input voltage of 2.2 V to 3.3 V. The used component (LMR61428) did not work properly on the first PCB due to some design flaws. Because of this a second version correcting these flaws was designed produced.



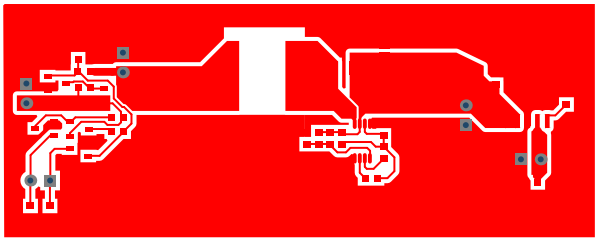
(a) 3D picture of version one of the power latch



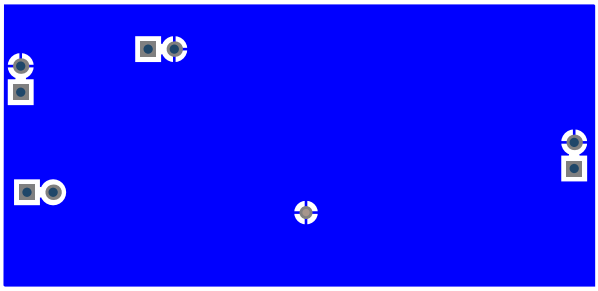
(b) 3D picture of version two of the power latch



(c) Top layer of the version one powerlatch



(d) Top layer of the version two powerlatch



(e) Bottom layer of the version one powerlatch

(f) Bottom layer of the version two powerlatch

Figure 4.9: Comparison between the first and second version of the power latch to show their difference of the layout and components

4.2 Software

This section describes the software, which was developed for the different components. Figure 4.10 shows the general sequence diagram of one display refresh cycle.

4.2.1 Receiver

Microcontroller

To develop firmware for the STM32, the Hardware abstractions layer provided by STMicroelectronics is used. The firmware running on the microcontroller is responsible for the following tasks:

- Initialize the IT8951 display driver chip
- Receive data via UART from the BLE-module

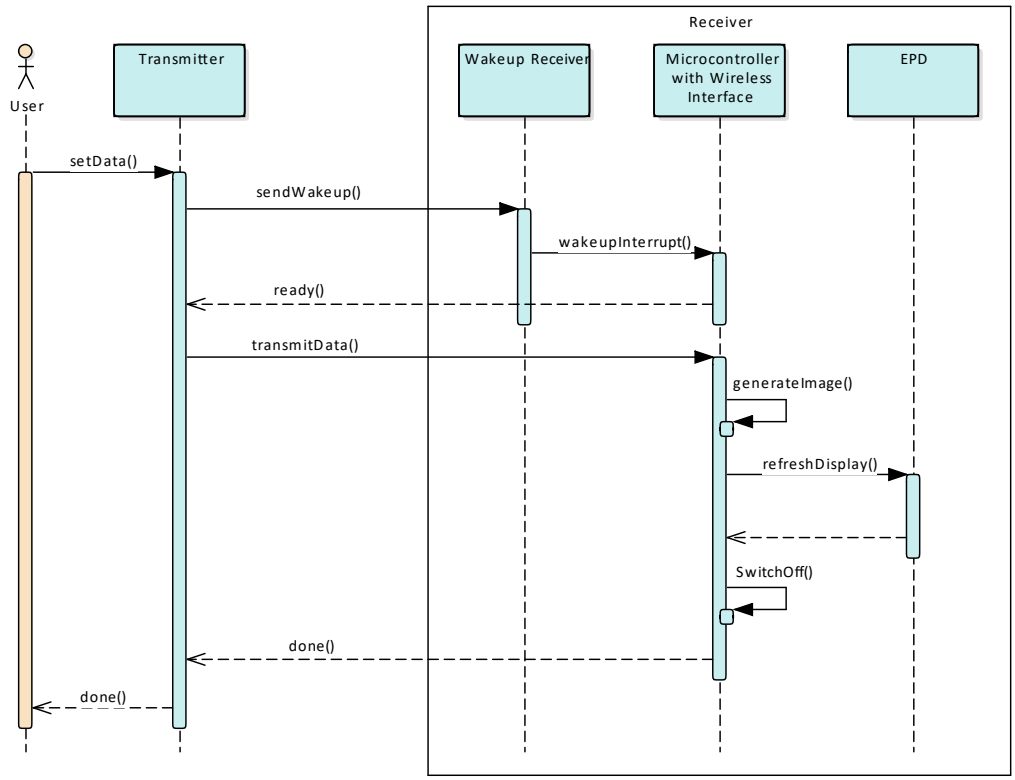


Figure 4.10: Refresh sequence.

- Generate the displayed image data from a template

- Write received text strings to the EPD

- Write the image data to the display driver via SPI

- Control the power supply of all the different components

Configurations The STM32 uses two bus interfaces and some extra GPIO's to communicate and control the EPD, nRF52480 and power supply. All used pins in these interfaces are listed in Table 4.1.

In the flowchart in Figure 4.11 is visible, that the controller itself never enters a deep sleep mode in the end of an program cycle. Instead the controller runs trough a power kill sequence which disables it's own power supply. As a result, the controller repeats the initializing process at the begin o every wakeup.

IT8951 Driver To program the communication between the STM32 with the IT8951, a programming guide by the manufacturer ITE Tech. Inc. was used. The interface is based on a 16-bit SPI-bus with three additional control pins. Since the IT8951 requires a big-endian format, the STM32 has to reformat all the bytes. The manufacturer recommends a SPI clock rate of 12 MHz, while the maximum given in the data sheet is 24 MHz. This leads to a maximum

Description	Function	Pin number	Speed	Properties	mode
SPI	Chip select	PA0	14.0 MBits/s	No Pull-up/-down	Output
	Clock	PA1	14.0 MBits/s	No Pull-up/-down	Push Pull
	MISO	PA6	14.0 MBits/s	No Pull-up/-down	Push Pull
	MOSI	PA7	14.0 MBits/s	No Pull-up/-down	Push Pull
UART	TX	PC10	115.2 kBits/s	Pull-up	Push Pull
	RX	PC11	115.2 kBits/s	Pull-up	Push Pull
Power kill	Enable	PC3	112 MHz	No Pull-up/-down	Output
	Disable	PC0	122 MHz	No Pull-up/-down	Output
EPD interface	Reset	PA4	112 MHz	No Pull-up/-down	Output
	HARDY	PA5	122 MHz	No Pull-up/-down	Output

Table 4.1: Table of the STM32 pin configuration.

transfer rate of 3 MBytes/s [14]. Writing an image to the EPD needs

$$t_{\text{write}} = \frac{(1200 \cdot 825)/2 \text{ bits}}{3 \text{ MBytes}/s} = 0.165 \text{ s}. \tag{4.7}$$

Note that this calculation assumes 4 bits (for the 16 grey scales) per pixel.

The measured time for one writing cycle, including initialisation and fetching of the buffer registers, resulted in 0.168 seconds. It is also possible to overclock the IT8951. With 28 MHz, it was still possible to read and write the display controller.

Template The schedule template is preloaded to the flash of the STM32. To save memory, the file is converted into an 1200×825 PNG with the colour depth of 4 bit. The PNG then is used to create a c-file which holds an one dimensional array with all the pixel data in it. The pixel location is mapped to the memory as shown in figure**??**.

```
const unsigned char schedule[]={0xff,0xff,0xff,0xff,0xff,0xff,0xff,...
                                0xff,0xff,0xff,0xff,0xff,0xff,0xff,...
                                .
                                .
                                .
                                0xdd,0xdd,0xdd,0xdd,0xdd,0xdd,0xdd};
```

Each Hex number in the array resembles two pixels of each four bit. In the template shown in image4.12, there are already some information given like the calendar week and weekdays. Storing the template like this uses 495000 bits of flash memory.

Fonts All fonts are c-files which hold two arrays. the first one contains all different ASCII characters, while the second array holds the font information like with and height. All characters have the compressed layout as described in Section 2.3.

Writing characters To write the desired characters to a image buffer, the pixel data needs to be converted and placed into the right spot. The code in Figure 4.13 shows the implemented algorithm to handle this task. The pixel depth used in the code example is 8-bits.

All other graphical functions implemented use the `drawChar()` function to access and write information into a given display buffer.

Communication

Due to the fact that the software on both sides of the bluetooth channel is very similar it is explained in this section. A BLE example project, which was provided by nordic, was only slightly changed to fit the purpose of this prototype.

Before a connection is established, the module on the receiver side is known as peripheral, and the module on transmitter end as the central device. After the receiver is woken up with the RFicient, the peripheral starts advertising. Advertising basically means sending data packets periodically with information for the central device. The central device is scanning for the this advertisement and can decide, whether it wants to connect based on this information. Is the connection established, it is now possible to exchange data through this channel. This process is illustrated in Figure 4.14.

The central device serves after connection as the client. It can make read and write requests to access the data on the peripheral, which acts as the server. The server on the other, hand can only send notifications to the client if new data is ready. It should be noted, that the roles of client and server do not in any case have to be assigned to central and peripheral, but also the other way around.

4.2.2 Transmitter

Python-Script

A short python script enables the user to send the data to the display. This script is presented step by step.

1. Include of the used packages, time and serial.

```
import serial
import time
```

2. Set desired baudrate, select which COM-port windows assigned to the development kit and open the port.

```
baudrate = 115200
com_port = 'COM14'

device = serial.Serial(com_port, baudrate, writeTimeout=0)
```

3. Create a list of lists with the desired data. The first list contains information about how many packages are about to follow. The following list contains the location on the template, the subject and the lecturer.

```
data = [[0, chr(3), '\0\0\0\0'],[11 , 'WsComm\0', 'MAH\0'],
        [3 , 'DigPro\0', 'SCU\0'], [22 , 'EmbSW\0', 'BON\0']]
```

4. Fill the list with `'\0'` so every package is 20 bytes long.

```
for i in range(len(data)):
    if len(data[i][1])<16:
        t = 14-len(data[i][1])
        data[i][1] = data[i][1]+'\0'*t
```

5. Store time when data transfer is started.

```
t_start = time.time()
```

6. Transfer data to the nRF52480 transmitter

```
    for i in range(len(data)):
        device.write([data[i][0]])
        device.write(bytes(data[i][1], 'utf-8'))
        device.write(bytes(data[i][2]+'\n', 'utf-8'))
        device.flush()
```

7. Print elapsed time on console and close COM-port.

```
    print('elapsed time: {:.3f}'.format(time.time() - t_start))

    device.close()
```

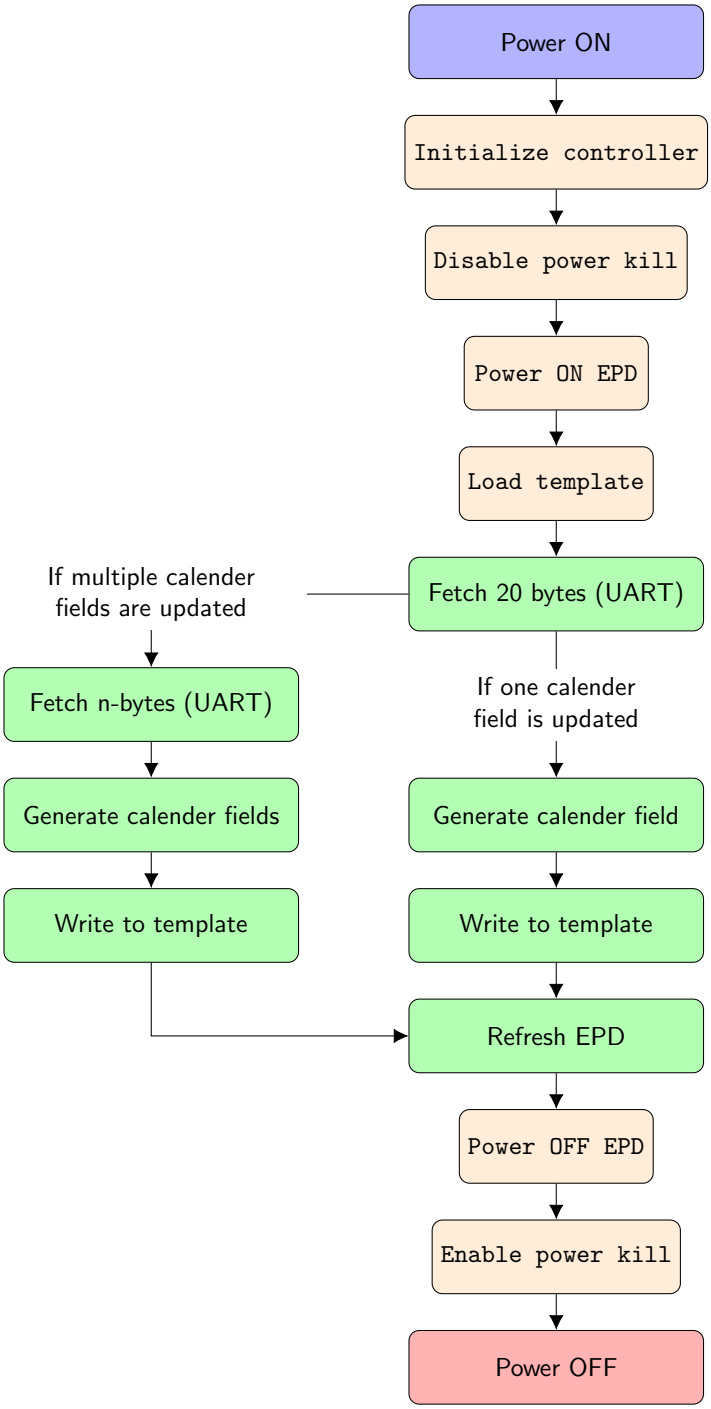


Figure 4.11: STM32 program flow chart.

ICOM INSTITUTE FOR COMMUNICATION SYSTEMS

KW46	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
8:10-8:55 9:05-9:50					
10:10-10:55 11:05-11:50					
12:10-12:55					
13:10-13:55 14:05-14:50					
15:10-15:55 16:05-16:50					
17:00-17:45 17:55-18:40					
18:00-18:45 18:55-19:40					

Figure 4.12: Example schedule used as an template.

```
const uint8_t Font12_Table[] =
{
    .
    .
    .,
    // @396 'A' (7 pixels wide)
    0x00, //
    0x30, //   ##
    0x10, //    #
    0x28, //   # #
    0x28, //   # #
    0x28, //   # #
    0x7C, //  #####
    0x44, //  #   #
    0xEE, // ### ###
    0x00, //
    0x00, //
    0x00, //
    .
    .
    .
    };

    sFONT Font12 = {
    Font12_Table,
    7, /* Width */
    12, /* Height */
    };
```

```
void drawChar(uint16_t Xpt, uint16_t Ypt, const char Acsii_Char,
                    sFONT* Font, uint8_t cBackground, uint8_t cForeground)
    {
        uint16_t Page, Column;
        if (Xpt > Image.Width || Ypt > Image.Height)
        {
            return;
        }
    uint32_t Char_Offset = (Acsii_Char - ' ') * Font->Height *
                           (Font->Width / 8 + (Font->Width % 8 ? 1 : 0));
    const unsigned char *ptr = &Font->table[Char_Offset];

    for (Page = 0; Page < Font->Height; Page ++ )
    {
        for (Column = 0; Column < Font->Width; Column ++ )
        {
            if (FONT_BACKGROUND == cBackground)
            {
                if (*ptr & (0x80 >> (Column % 8)))
                    setPixel(Xpt+Column, Ypt+Page, cForeground);
            }
            else
            {
                if (*ptr & (0x80 >> (Column%8)))
                {
                    setPixel(Xpt + Column, Ypt + Page, cForeground);
                }
                else
                {
                    setPixel(Xpt + Column, Ypt + Page, cBackground);
                }
            }
            //One pixel is 8 bits
            if (Column % 8 == 7)
                ptr++;
        }
        if (Font->Width % 8 != 0)
            ptr++;
    }
}
```

Figure 4.13: Write characters to image buffer.

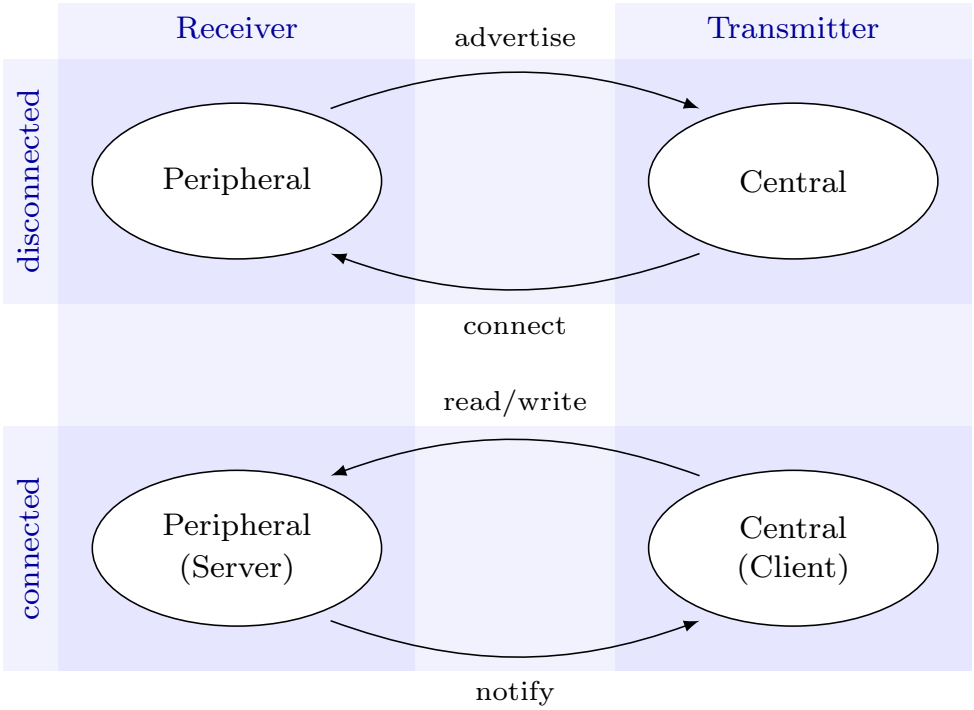


Figure 4.14: BLE-communication with the nRF52480.

Chapter 5

Results

5.1 Power supply

When a load is attached to the main output, there is a voltage drop over the ADP5090 ($V_{\text{BAT}} - V_{\text{SYS}}$ in Figure 4.8). This drop causes the chip to switch off the output way too early. It was necessary to bypass the ADP5090 and connect the receiver circuit directly to the capacitor due to this reason. This way, the capacitor is not protected from overdischarging, but charging of the capacitor is still controlled by the ADP5090.

Figure 5.1 shows a plot of the voltages of the power supply. The voltage across the capacitor

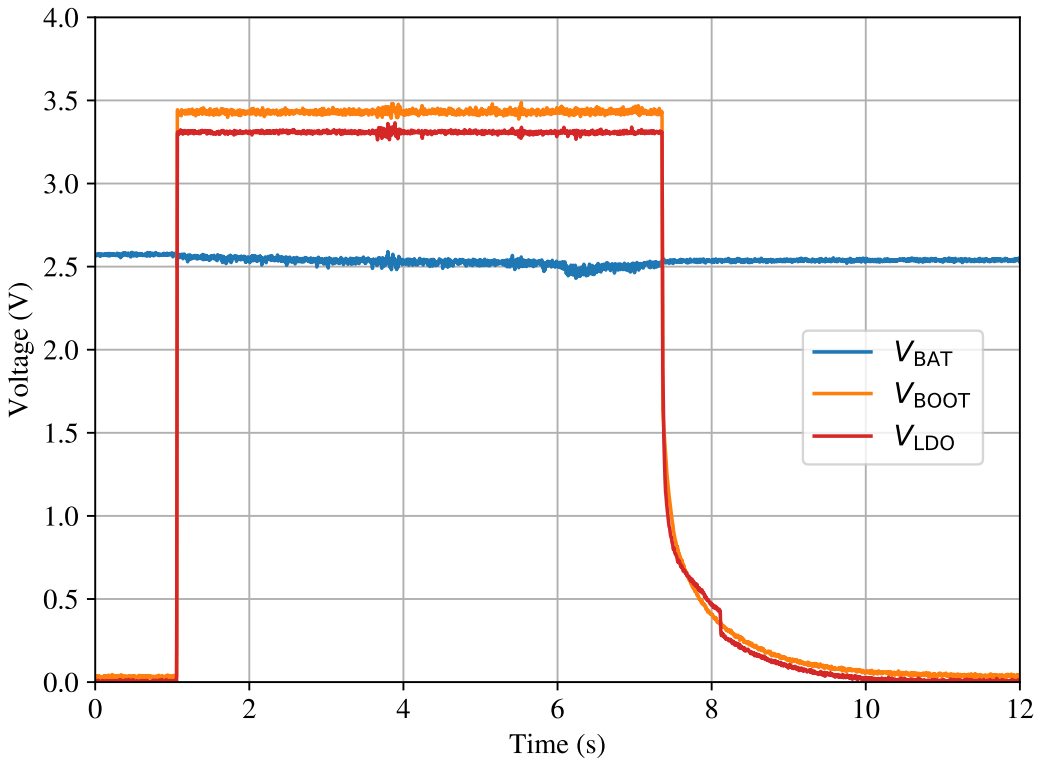


Figure 5.1: Power consumption during one write cycle.

is labelled as V_{BAT}. Additionally, the voltages after the step-up converter (V_{BOOT}) and LDO

(V_{LDO}) were tracked. The whole power supply manages to provide a stable voltage across the whole write cycle, despite high current peaks.

5.2 Power consumption

5.2.1 Power measurement

The N6705B power analyzer from keysight was used as the power supply, to look at the energy consumption in detail. To carry out measurements, the ADP5090 with the solar cells and super cap was disconnected and the remaining components connected to the power analyser with a fixed voltage of 3.3 V. This way, the voltage and current over one write-cycle could be tracked. The result is plotted in Figure 5.2. One write-cycle takes about 5.8 seconds. In this interval,

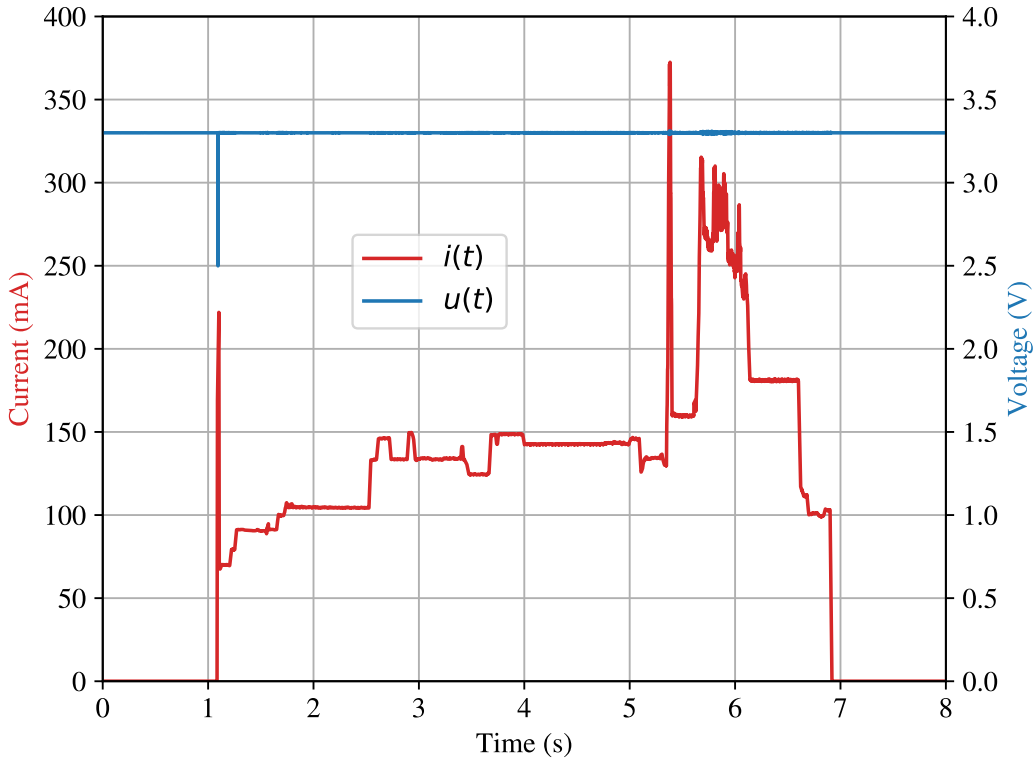


Figure 5.2: Current and Voltage during one write-cycle.

the STM32, nRF52480 and display driver are active. Clearly visible is the current peak, which occurs when the system is woken up. Short after the 5 seconds mark, initializing and data receiving is finished and the display is refreshed. The current peaks there are due to the charge pumps on the display driver.

With voltage and current given, the power over time can be calculated. This curve is plotted in Figure 5.3. obviously, the power follows the same course like the current since the voltage is constant. The energy needed for one write-cycle is given with the integral of the power over

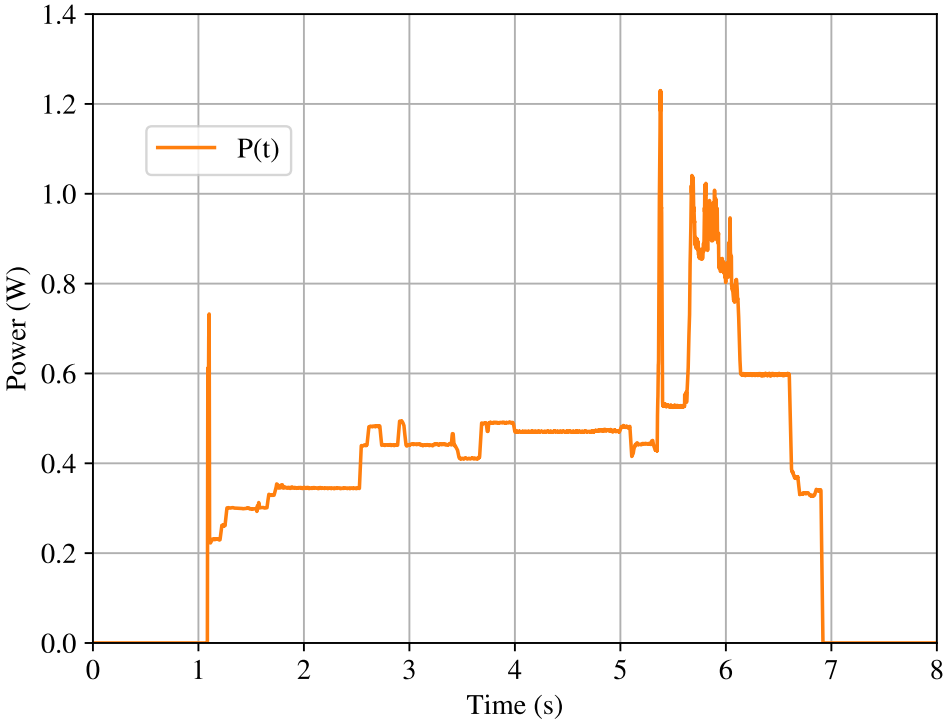


Figure 5.3: Power consumption during one write cycle.

time:

$$E = \int_0^{8\,\mathrm{s}} P(t)\mathrm{dt} \approx 2.7504\,\mathrm{J}. \tag{5.1}$$

Like mentioned in (4.1) the solar cells provide 485.52 μW at 200 lux. Considering the Rficient which consumes 4.5μW, it is possible to estimate the time necessary to harvest the power for one write cycle:

$$t_\mathrm{E} = \frac{2.7504\,\mathrm{J}}{485.52\,\mu\mathrm{W} - 4.5\mu\mathrm{W}} = 5717.85\,\mathrm{s} \approx 1\,\mathrm{h}\,35\,\mathrm{min}. \tag{5.2}$$

Not taken into account is the leakage of the super capacitor and losses in the ADP5090.

5.2.2 Energy storage

The stored energy in a capacitor is

$$E = \frac{1}{2}CU^2.$$

since the capacitor operates between 2.2 V and 3.6 V, the total stored energy is

$$E_\mathrm{ges} = \frac{1}{2}C(U_\mathrm{max}^2 - U_\mathrm{min}^2) = \frac{1}{2}\cdot 40\,\mathrm{F}\cdot((3.6\,\mathrm{V})^2 - (2.2\,\mathrm{V})^2) = 162.4\,\mathrm{J}. \tag{5.3}$$

By again neglecting the loss in the ADP5090 and the leakage of the cap, it is possible to estimate the time

$$t_{\text{func}} = \frac{162.4\,\text{J} - 2.7504\,\text{J}}{4.5\,\mu\text{W}} = 3.54877 \cdot 10^{-7}\,\text{s} \approx 411\,\text{d} \tag{5.4}$$

in which the prototype is fully functional when no energy is harvested. The 2.7504 J are subtracted because on should be able to refresh the display after this time. This result should be taken with caution because it neglects losses and does not consider the tolerance of the capacitor (±20% corresponds to ±8 F with a 40 F capacitor [12]). The order of magnitude, however, should be correct.

5.2.3 Amount of display refreshes

With the results from the Equations (5.1) and (5.3), the amount of display refreshes can be calculated:

$$n = \left\lfloor \frac{162.4\,\text{J}}{2.7504\,\text{J}} \right\rfloor = 59. \tag{5.5}$$

Note that additional charging is neglected and the super capacitor is assumed fully charged in this calculation.

To verify this value, the super capacitor was charged to 3.5 V and the whole refresh sequence was repeated until the voltage reached 2.2 V. Under this conditions, the display could be refreshed 46 times.

5.3 Communication

5.3.1 nRF52480

The data transfer with the nRF52480 programmed as described is very simple. The connection is stable in a similar range like the RFicient, and connecting the two devise after wakeup happens instantaneously.

5.3.2 RFicient

The development kit is a preliminary version which has to be sent back to the IIS after evaluating. Out of this reason, it was not possible to supply it directly from the super capacitor, because this would have involved soldering. The kit was therefore supplied over USB and it's power consumption assumed to be 4.5 μW as stated in the datasheet. Because the Rficient had to be controlled over the GUI, it was not possible to synchronize the wakeup event with the data transmission. To ensure functionality, enough delay time (2 s) between these two events was inserted. With a fully synchronized transmitter, it should be possible to reduce the time, in which the microcontroller is in wakeup mode.

Chapter 6

Conclusion

Over the course of this thesis, a working prototype of an occupancy schedule was developed. As components served several development kits and simple prints. With a self-holding circuit, it was possible to turn off parts of the receiver completely and switch them back on with the a wakeup interrupt. This central part makes it possible to use more power consuming hardware, since it only is connected to the power supply for intervals of 5-6 seconds. Using a low power wakeup receiver turned out to be the right approach, since the power consumption during standby mode could be reduced to a minimum of $4.5\,\mu$W that way. This power loss is easily compensated by the solar cells which deliver $485.52\,\mu$W given an illuminance of 200 lux. If no light reaches the solar cells and no wakeups occur, it is assumed that the prototype stays functional for over one year, under the condition that the super capacitor was charged fully beforehand.

Future work Following improvements on the prototype have to be made, before developing a complete layout of the receiver:

- Developing a power management which protects the super capacitor from over discharging
- Changing the microcontroller to display driver connection from SPI to a parallel bus (for ex. I80)
- Synchronising of the refresh sequence on transmitter end
- Measure the leakage of the super cap to better estimate long term functionality

These adjustments should ensure functionality and reduce power consumption of the prototype.

Further it may be useful, to extend the BLE-communication to a mesh network, or even change to a different protocol with more range.

References

[1] K. Amundson, *Electrophoretic Displays*. Springer International Publishing Switzerland, 2016.

[2] *Information technology — programming languages — c*, Standard, Geneva, CH, Mar. 2011.

[3] *AS3933 3D Low Frequency Wakeup Receiver*, Mar. 2015.

[4] *RFicient ULTRA-LOW POWER WAKE-UP RECEIVER DATASHEET (preliminary)*, 2019.

[5] *MSP430FR698x(1), MSP430FR598x(1) Mixed-Signal Microcontrollers*, Aug. 2018.

[6] *STM32L4R5xx STM32L4R7xx STM32L4R9xx*, Apr. 2018.

[7] (). 9.7inch e-Paper HAT, [Online]. Available: `https://www.waveshare.com/wiki/9.7inch_e-Paper_HAT`.

[8] *nRF52480 Product Brief*, 2017.

[9] *Amorphous Silicon Solar Cells*, Aug. 2019.

[10] *Ultralow Power Boost Regulator with MPPT and Charge Management*, Feb. 2017.

[11] *EVAL-ADP5090 User Guide*, 2014.

[12] *Cylinder Type Lithium Ion Capacitors*, Aug. 2019.

[13] *nRF52840 Objective Product Specification v0.5*, Dec. 2016.

[14] *IT8951 Programming Guide (Host I90 Command Based)*, Jan. 2019.

List of Figures

List of Tables

Statement of Plagiarism

We declare that, apart from properly referenced quotations, this report is our own work and contains no plagiarism; it has not been submitted previously for any other assessed unit on this or other degree courses.

Place	**Date**
Rapperswil	December 19, 2019

Signatures

Cedric Renda

Manuel Tischhauser

Appendix A

Requirements

A.1 Assignment

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Low power wakeup receiver

Semesterarbeit für Manuel Tischhauser und Cédric Renda

Herbst 2019

1. Einführung

Im Gebäudemanagement ist es üblich, Belegungspläne an den Eingängen der Räume anzubringen. Oftmals sind diese in Papierformat und müssen bei einer Änderung von Hand gewechselt werden. Mit dieser Methode werden kurzfristige Belegungen nicht aufgezeigt. Dies könnte man umgehen, wenn man mit Displays arbeitet die Wireless aktualisiert werden können. Dabei stellt sich allerdings das Problem, dass man entweder Kabel für die Netzeinspeisung verlegen muss oder Batterien verwendet, die regelmässig ersetzt werden müssen. Im Idealfall entfällt die Speisung komplett.

2. Aufgabenstellung

Zu einem Empfängermodul soll eine bidirektionale low power Kommunikationsstrecke aufgebaut werden. Der Empfänger soll durch Energy-harvesting Massnahmen möglichst passiv betrieben werden können. Dieser enthält einen Ultra Low Power Wake-up Receiver und eine Anzeige mit dem die empfangenen Daten auf entsprechende Weise dargestellt werden. Ein System kann aus mehreren Empfängern bestehen, welche unabhängig voneinander vom Sender selektiert werden können.

3. Ablauf

Zu Beginn der Arbeit sind ein Projektplan und ein Pflichtenheft zu erstellen, welche in den ersten Wochen dem Betreuer abgegeben werden müssen. Ein Vorschlag des Pflichtenheftes befindet sich im Arbeits-platzordner oder als *.docx File auf dem Public Server (\\hsr.ch\root\auw\sge\labors\Mk\pub_for_students). Planen Sie total 240 Arbeitsstunden (8 ECTS * 30 h/ECTS) ein. Die Arbeiten sollen innerhalb der Gruppe geeignet aufgeteilt werden; die Aufteilung ist im Bericht entsprechend festzuhalten, genauso wie ein Vergleich des geplanten und des effektiv durchgeführten Projektplans. Weitere Einzelheiten werden an den wöchentlichen Besprechungen festgelegt. Die Arbeiten sollen möglichst selbständig durchgeführt werden. Die Kriterien der Beurteilung und Notengebung sind im unten erwähnten Leitfaden zu finden.

4. Laborjournal

Während der Arbeit ist ein persönliches (d.h. pro Person eines), gebundenes, handschriftliches und datiertes Laborjournal zu führen. Darin werden alle Tätigkeiten betreffend Dauer und Resultate eingetragen. Ebenfalls soll darin ein Protokoll geführt werden von den wöchentlichen Treffen. Das Laborjournal wird am Ende der Arbeit abgegeben und wird mitbenotet.

5. Bericht

Über die Arbeit ist ein Bericht zu verfassen, dessen Textteil maximal 60 Seiten umfassen und eine Dateigrösse von 5MB nicht überschreiten soll. Im Bericht sollen alle gemachten Überlegungen, Abklärungen, Berechnungen und Untersuchungen detailliert (in Text und Bild) dokumentiert werden. Der Bericht muss gut leserlich geschrieben und übersichtlich gegliedert sein. Weitere Richtlinien, wie ein Bericht aufgebaut sein kann, und weitere nützliche Informationen findet man im Leitfaden, welcher in gedruckter Version im Arbeitsplatzordner und auf dem Public Server abgelegt ist.
Des Weiteren muss im Bericht unbedingt eine unterschriebene Nicht-Plagiatserklärung enthalten sein, ein Beispiel dieser Erklärung befindet sich auf dem Public Server.
Der Bericht ist in 1 Papier-Exemplar abzugeben, mit einer beiliegenden CD-ROM, auf der alle anfallenden Daten, wie auch der Bericht selbst (im PDF-Format) gespeichert sind.

Writing in English is highly encouraged.

6. Termine

Beginn der Arbeit:
Abgabe des Berichts:
Mündliche Präsentation:

7. Organisatorisches

Betreuung der Arbeit:
Betreuung des Labors:
Arbeitsplatz:
Industriepartner:
Besprechungen: wöchentlich, nach Vereinbarung, an der HSR
Examinator: Prof. Dr. Heinz Mathis, hmathis@hsr.ch

Rapperswil, [Datum]

Viel Erfolg wünscht Ihnen

Heinz Mathis,
Dozent Mobilkommunikation

A.2 Requirement Specification

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

Pflichtenheft

Projekt: Low power wakeup receiver

Version 0.1

Cédric Renda, Manuel Tischhauser

Name	**Datum**	**Unterschrift**
Prof. Dr. Heinz Mathis		
Selina Malacarne		
Cédric Renda		
Manuel Tischhauser		

27. September 2019

Inhaltsverzeichnis

1 Einleitung

Im Gebäudemanagement ist es üblich, Belegungspläne an den Eingängen der Räume anzubringen. Oftmals sind diese in Papierformat und müssen bei einer Änderung von Hand gewechselt werden. Mit dieser Methode werden kurzfristige Belegungen nicht aufgezeigt. Dies könnte man umgehen, wenn man mit Displays arbeitet, die über eine drahtlose Schnittstelle aktualisiert werden können. Dabei stellt sich allerdings das Problem, dass man entweder Kabel für die Netzeinspeisung verlegen muss, oder Batterien verwendet, die regelmässig ersetzt werden müssen. Im Idealfall entfällt die Speisung komplett.

2 Auftrag

Im Rahmen dieser Semesterarbeit soll eine Lösung zur oben beschriebenen Problematik ausgearbeitet werden. Der Fokus liegt auf der Erstellung eines autarken Anzeigesystems welches über eine drahtlose Schnittstelle bedient werden kann. Die folgende Punkte sollen dabei abgearbeitet werden:

- Recherche bezüglich Schnittstelle und Energy Harvesting.
- Vor- und Nachteile bestehender Technologien abwägen und geeignete Hardware wählen.
- Erstellen eines lauffähigen Prototypen.

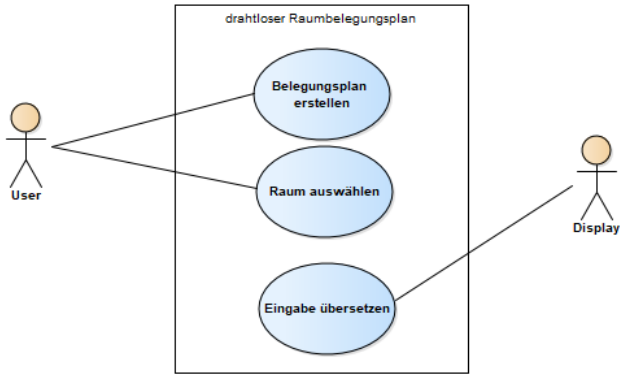


Abbildung 2.1: *Use-Case Diagramm*

Der Prototyp richtet sich nach dem Use-Case Diagramm in Abbildung 2.1.

3 Produktanforderungen

Das Empfängermodul ist autark und kann auf einem Display Stundenpläne anzeigen. Diese werden über eine drahtlose, bidirektionale Schnittstelle gesendet. Der Sender wird mit einem Computer bedient.

Besteht das System aus mehreren Empfängern, so kann das Sendemodul diese unabhängig voneinander selektieren.

3.1 Hardware

Sender

- Schnittelle zum Computer
- Sendemodul

Empfänger

- Mikrocontroller oder Vergleichbares (Prozessor, Speicher, usw.)
- E-Paper-Display
- Energy-Harvesting-Einheit
- Energiespeicher
- Empfangsmodul

3.2 Software

Sender

- Treiber für Sendemodul

Empfänger

- Firmware für Mikrocontroller

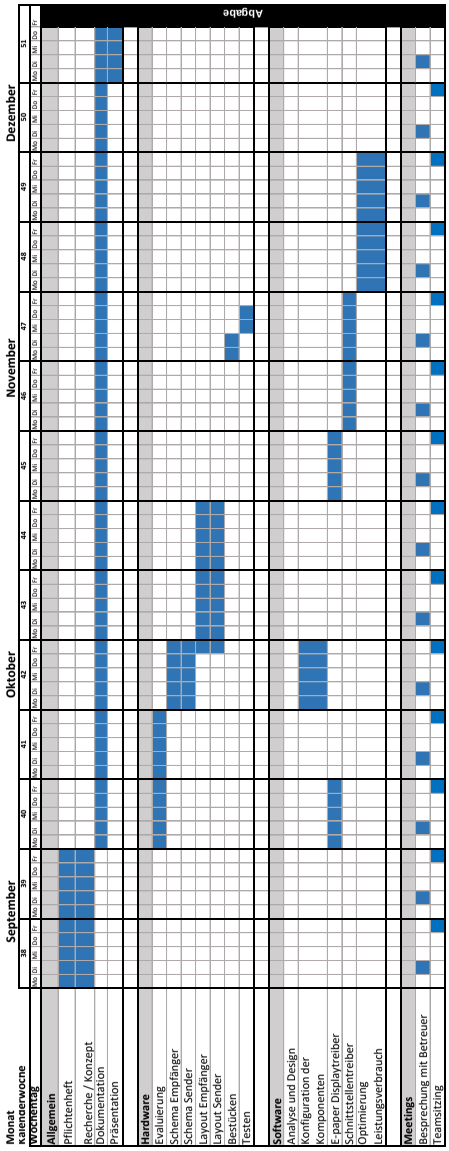
3.3 Varianten/Optionen

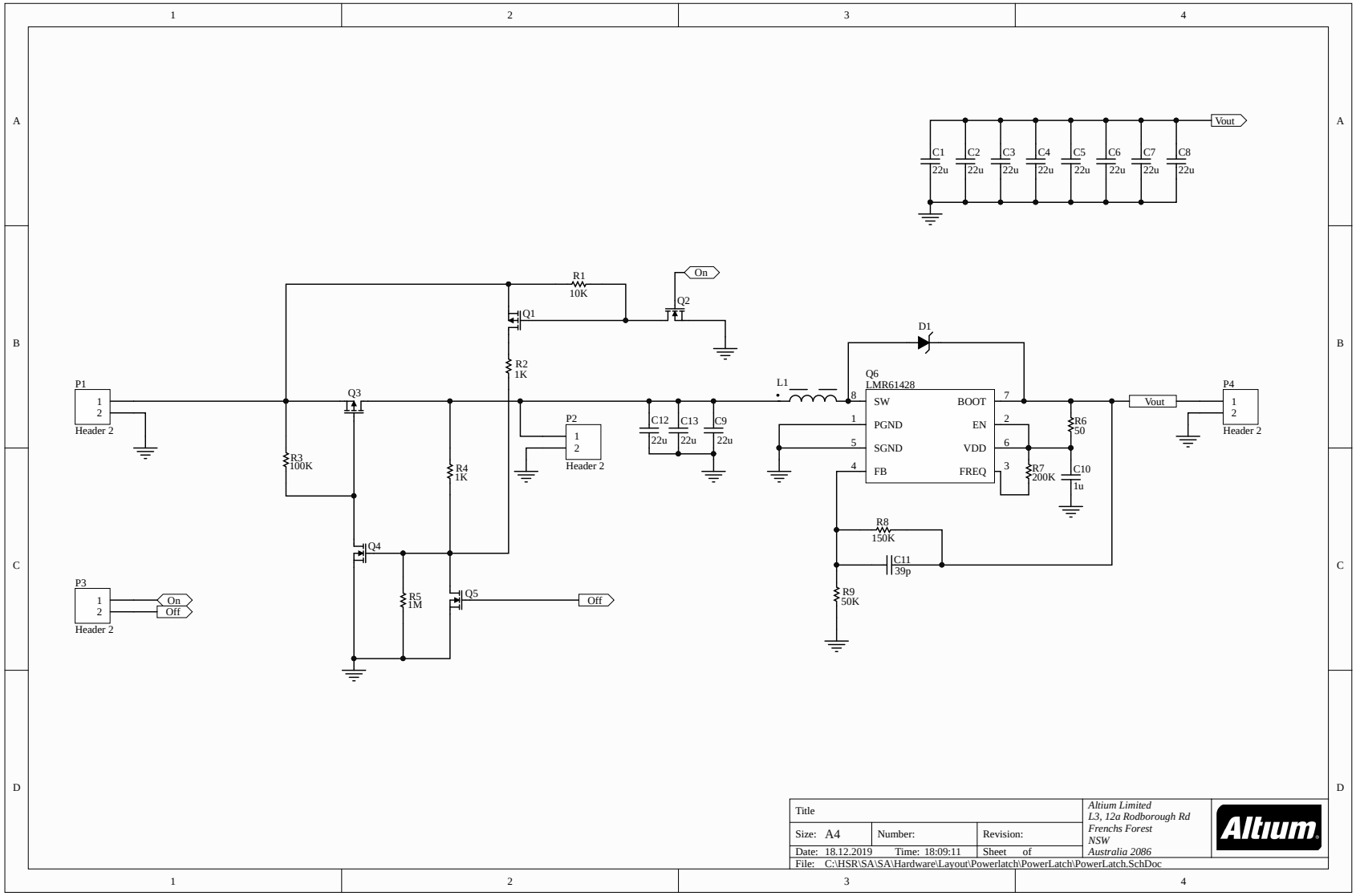
Ist der Prototyp funktionsfähig, soll zu einem späteren Zeitpunkt auch möglich sein, verschiedene Bildschirmgrössen zu verwenden, wobei sich auch Anzeige nicht nur auf Raumbelegungspläne beschränkt. Deshalb soll das System und insbesondere die Software so flexibel wie möglich entwickelt werden.

3.4 Dokumentation

Die Dokumentation beinhaltet sämtliche Überlegungen, Abklärungen, Berechnungen und Untersuchungen, welche im Laufe der Semesterarbeit gemacht wurden.

4 Zeitplan


Monat
Kalenderwoche
Wochentag
September
Oktober
November
Dezember
Abgabe
Allgemein
Pflichtenheft
Recherche / Konzept
Dokumentation
Präsentation
Hardware
Evaluierung
Schema Empfänger
Schema Sender
Layout Empfänger
Layout Sender
Bestücken
Testen
Software
Analyse und Design
Konfiguration der Komponenten
E-paper Displaytreiber
Schnittstellentreiber
Optimierung Leistungsverbrauch
Meetings
Besprechung mit Betreuer
Teamsitzing


1
2
3
4
A
B
C
D
Vout
C1 22u
C2 22u
C3 22u
C4 22u
C5 22u
C6 22u
C7 22u
C8 22u
On
R1 10K
Q1
Q2
D1
R2 1K
P1
1
2
Header 2
Q3
L1
Q6
LMR61428
SW
PGND
SGND
FB
BOOT
EN
VDD
FREQ
P4
Header 2
P2
Header 2
C12 22u
C13 22u
C9 22u
R6 50
R3 100K
R4 1K
R7 200K
C10 1u
R8 150K
C11 39p
Q4
P3
On
Off
Header 2
R5 1M
Q5
Off
R9 50K
Title
Size: A4
Number:
Revision:
Date: 18.12.2019
Time: 18:09:11
Sheet of
File: C:\HSR\SA\SA\Hardware\Layout\Powerlatch\PowerLatch\PowerLatch.SchDoc
Altium Limited
L3, 12a Rodborough Rd
Frenchs Forest
NSW
Australia 2086
Altium