Smart Village

Remote monitoring of drinking water consumption in a rural environment

Peter Vittali

January 26, 2024
v1.0

ABSTRACT

Remote monitoring of drinking water in the context of this project means that the amount of drinking water used by a group of people is constantly measured in terms of liters per minutes. The objective is to detect abnormal consumption which could be caused by a pipe leak or by some sort of exceptional usage by one or more subscribers. In both cases, the origins of this excess consumption must be analyzed as soon as possible. The monitoring system must immediately inform the person in charge of the water distribution system, preferably by SMS and/or email. In addition, the monitoring system should provide the data to build usage profiles. These profiles would show the consumption of drinking water over the course of a day, during summer and winter and during holidays. The latter is important because the village involved in this project has considerable touristic activity.

Challenges arise from the lack of energy and of reliable network infrastructure at the point of metering. The energy powering the system must thus be produced off-grid. We have chosen small solar panels. Data communications rely on GSM, whose coverage turned out to be intermittent in the location where the system operates.

CREDITS

L'équipe PGSSE des communes de Le Puid, Le Vermont et de Saint-Stael is a small team of volunteers in north-eastern France. PGSSE (*Plan de gestion de la sécurité sanitaire des eaux*) is a policy signed in law by the French government to ensure the quality and availability of drinking water on the entire french territory. The PGSSE is based on well-known international standards like IOS9001. All communities with more than 50 inhabitants must be certified according to the rules defined in this standard.

The team is currently comprised of the following individuals:

Francois Boulet, Gilles Droin, Jacotte Kinsk, Patrick Lorin, Jean-Gilles Naivin, Patrice Omurbek, Peter Vittali.

Gilles Droin has laid the groundwork for this project by founding the association *L'Atelier d'Acccompagnement Numérique*. This organization helps small, rural, communities to become more independent with respect to theirs purchase and usage of IT equipment. In particular, the transition to the Linux operating system is encouraged. It is an initiative to deploy open source technology in rural communities.

Many thanks also to Régine Chinouilh, major of Le Puid / Vosges, who tolerated many failed experiments and who was willing to deal with complaints from locals who didn't understand why we were degrading their beautiful environment with solar panels, plastic conduits and cables.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

A rural village in north-eastern France is supplying roughly 100 inhabitants with drinking water from local sources. These sources have been built throughout the 70ies and have so far never run dry. In recent years, however, signs of reduced flow have been observed and a couple of actions have been taken to prevent outright interruptions of the water supply to the village.

1.1 Motivation

Firstly, leaks in the water distribution network have been repaired. Secondly, a system was designed to detect new leaks right as they occur, rather than at some point later when supply issues become apparent. This has several important advantages. It is sometimes easier to locate the exact position of the leak when the advent of the leak can be time correlated with recent events in the area, for example ongoing construction work may have damaged a conduit. The water quality might suffer because untreated groundwater can infiltrate the water distribution network through the defect conduit. And lastly, initiating repairs only when absolutely necessary means that these repairs are then started during summer or at the end of the summer when upstream sources run low and consumption is at it highest. This period of time is not ideal for repair work because users are more impacted and local repair services will be strained because many other villages in the area will experience exactly the same problems. This situation is further aggravated by the fact that the current water conduits are vulnerable to small dislocations of the surrounding earth mass which increases after periods of drought due to the desiccation of the soil throughout the summer.

Preventive repairs are therefore mandatory to maintain a reliable water distribution infrastructure.

Another aspect of drinking water management is consumption management. Currently, subscribers pay a yearly lump sum and there are no individual water meter inside each subscriber home. This situation is certain to change, and it might be interesting to understand how this change influences consumption patterns. Consumption patterns are also important for conduit dimensioning. The required peek throughput is an important parameter when dimensioning the diameter of water pipes and the size of valves.

1.2 Challenges in a rural setting

I hope to have made clear, why flow monitoring in the drinking water system described above is necessary. The question is then: why not simply purchasing a smart-meter or even simpler, entering a service agreement with a resource management company that collects the data and makes it available in a cloud environment. It turns out that while this kind of facility management is increasingly common in cities, a concept also referred to as *Smart City*, it is not commonly used in the rural setting that I am referring to here. There are several reasons for this:

- Products are not geared towards small communities: Big players in the area of smart meters tend to offer products for industrial applications which do not cater very well to the requirements in our rural context, in particular with respect to cost.
- Lack of extensibility: Commercial products in the sector are not open-source, they might be protected by patents and even the collected data might be owned by the service provider, not the client. The market is dominated by few players with wide strategical moat and little competition. This makes it difficult to extend the system. For example, when metering the water flow it might be tempting to also meter water quality, temperature, rainfall and other environmental parameters of merit. I believe that the open-source/DIY/maker community has an important role to play.

CHAPTER 2

EXISTENT INFRASTRUCTURE

Before discussing solutions to these challenges, we must understand the current situation and constraints to which our remote metering system must comply:

- Absence of electricity at the site of deployment: we must produce the electricity required to run the system with some off-grid technology, for example solar panels. During certain weather conditions like snowfall, solar power might be unavailable for a longer period than battery power can last and the system must be able to hibernate and restart without manual intervention.
- Absence of reliable wireless infrastructure: The collected data must be transmitted to a location where it can be stored and analyzed. Network coverage at the site of deployment turns out to be week and unreliable. The system must be able to cope with the inability to transfer measurements at a certain of point of time and adopt some strategy of retrial without introducing measurement errors.
- how to interface with the current water meter ?

Let's start with the last point and have a closer look at what we have at the deployment site.

2.1 Existent water meter

The existent water meter installed upstream of the distribution network is shown in Fig 3.1. All drinking water headed towards the village passes through this meter, thus the totalizer on top of the device shows to total consumption with a resolution of 1 l. The device is sealed and the small reflective, revolving disc segment shown in Fig 2.1c is made of non-magnetic material to avoid tampering, emphasized within the red-bordered ellipse on the photo.

The manufacturer offers three different communication facilities:

1. walk-by and drive by systems
2. pulse output
3. radio frequency LoRaWAN and Sigfox networks

Option 1 is not practical because of the requirement to automatically detect possible leaks. This must be done on a permanent basis without human intervention. The second option would require some investment to change or unblock the current configuration. This is certainly possible, and it would have been the ideal option for this use-case. However, there exist many older meters in the area without that feature and I believe that it is interesting to develop techniques that can be used in conjunction with older analog devices. Finally, the last options, LORaWAN and Sigfox both require additional network infrastructure (and electricity) to connect to the Internet. When we

(a) WOLTEX M water meter (b) totalizer (c) totalizer detail

Figure 2.1: The existent water meter

started this project, LORaWAN was relatively new and the realistic range of radio coverage in a mountainous and forested environment was unclear. Also, LoRaWAN - internet gateways were relatively expensive and complicated to configure. Sigfox is a proprietary network owned by UnaBiz and for reasons already mentioned, we strongly prefer open-source solutions.

We therefore choose to interface with the current meter via opto-coupling. In fact the totalizer shown in Fig 2.1c can be used to trigger an optical sensor, for example an optical switch.

We can now propose a system that fits into the current environment.

CHAPTER 3

REMOTE WATER METERING SYSTEM OVERVIEW

The proposed system consists of software and hardware components shown in 3.3.

3.1 Opto-coupling with the existing meter

We decided to glue an opto-electronical device right on top of the revolving disc segment. An infrared is reflected from the disc as it passes underneath the sensor. One revolution equals 10 l. The details on how this signal is translated into a digitized pulse stream is discussed later in this document.

3.2 Off-grid electricity production

Since the existent water meter is installed inside a water gully with no electricity outlet in the vicinity, the energy required to operate the system must be produced by the installation. We explore two different approaches:

1. a small solar panel capable of delivering 10 W peak power.
2. a microturbine connected to a nearby reservoir overflow outlet capable of delivering approx. 8 V @ 8 mA.

(a) solar panel (10 W) (b) consumer microturbine (60 mW)

Figure 3.1: Inexpensive off-grid power production

Currently, we use two solar panels oriented south and west, respectively. The microturbine requires a different circuit in order to charge a LiPo battery. We will discuss this circuit in a different article. Also, note that the depicted model is rather a toy and most likely unsuited for an application in production. We also noted that the additional wiring and electrical trench provision are probably too high a cost in our use case. But we do think, that it would be interesting to gather more data and experience with this kind of inexpensive consumer products because in some situations - and for some village budgets - this might be the only choice.

The two solar panels are located in a rather shady forest area at a distance of 15 m from the gully housing the water meter and the sensor hardware.

Figure 3.2: Solar panel configuration

3.3 MQTT

MQTT is a widely used protocol to connect embedded systems - things - to the internet. It is ideally suited (but not limited to) for applications where relatively low data volumes must be transmitted in regular time intervals, like for example temperature sensors. In our case, we need to transmit readings from the water meter as often as the available amount of energy allows it. Currently, this amounts to one transmission per day, but there is potential for more frequent transmissions, as we will discuss later in this paper. Recall, that we want to react rapidly to potential leaks.

MQTT is touted as being “lightweight”:

> MQTT is an OASIS standard messaging protocol for the Internet of Things (IoT). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint and minimal network bandwidth.

Note that the cloud based broker we use is not free of charge. Currently, we occur between 5 and 10 USD in transmission costs per month. This is not negligible and points to the necessity to replace the GSM network layer with a locally provided infrastructure, wherever possible. One way to do so would be a LoRaWAN-type protocol based on the 867 Mhz band. Another option is to use a service like ngrok to securely expose a locally hosted MQTT broker to the Internet.

3.4 Data retrieval and storage

After the meter readings have been uploaded to the MQTT broker, we need to retrieve it to a local workstation or server for storage, display and further analysis. We have developed a Java based application that runs as a Linux daemon on an always-on small form factor PC in the office of a local resource manager. This software connects to the MQTT broker, downloads the latest readings and stores them to a local database. User relevant data is also stored to a cloud-based database, from where the data can be displayed on a globally accessible internet website dedicated to the resource management of a group of communities. More comprehensive data is further published to a local MQTT broker. We use Home Assistant as a highly configurable admin dashboard. The daemon software also monitors consumption thresholds. Unusually high average readings potentially implying a leak will trigger an SMS alert and an email alert.

The design and implementation of this software will be discussed in a different paper.

In this chapter we will highlight some elements shown Fig 3.3, specifically those in the Embedded System part.

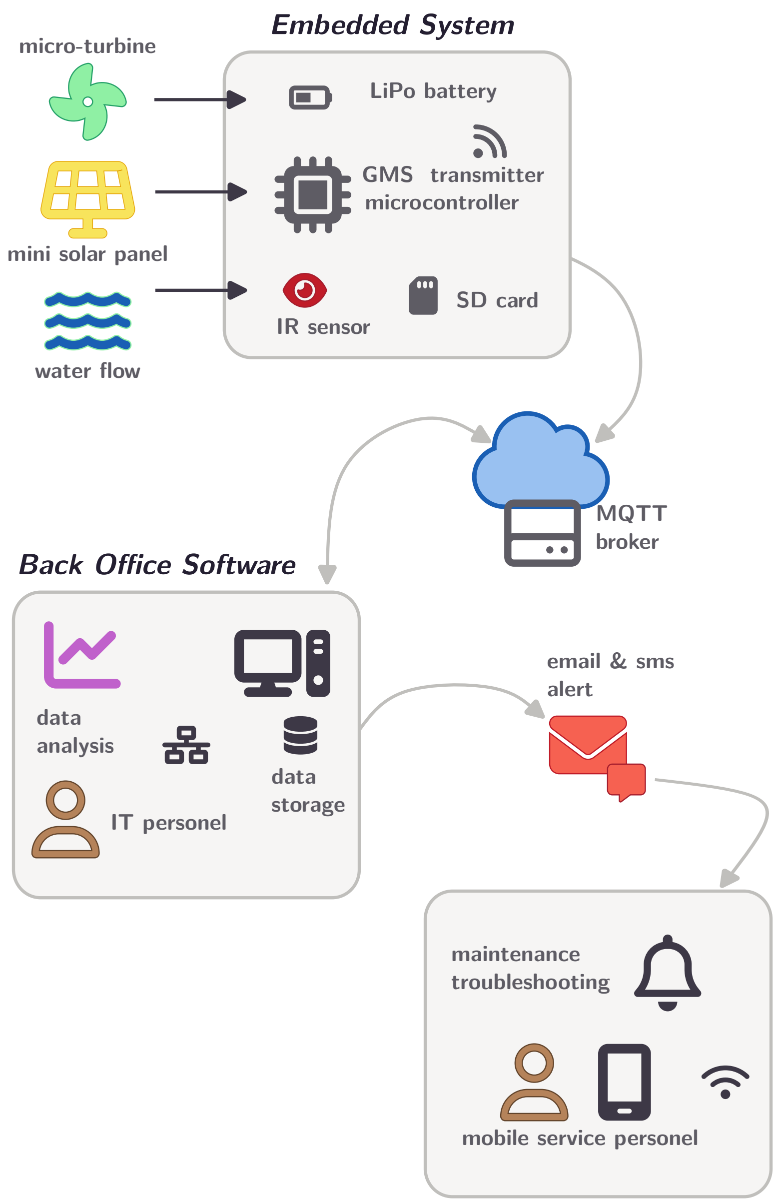


Figure 3.3: Remote Water Meter system overview

CHAPTER 4

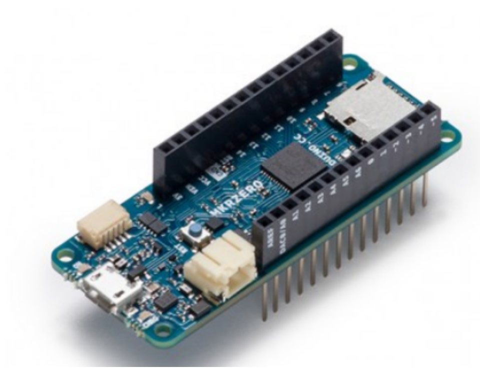
HARDWARE

The hardware comprising the embedded system in 3.3 is based on a master-slave configuration.

4.1 Theory of operation

Master-Slave microcontroller configuration

We use a pair of 32-bit microcontroller to capture and transmit data collected from the water meter. The reason for this configuration stems from the fact that we were unable to completely switch off the GSM module on the Arduino MKR GSM 1400 development kit (DK) which resulted in more energy consumption than we can meet with the off-grid solution we have described in the last section. Thus, the master controls the infrared sensor, digitizes the sensor signal and accumulates a counter which it transmits to the slave. The slave transmits the data during selected time slots. The master controls power to the slave and the latter is completely switched off outside of transmission intervals. We will discuss further details of the pro and cons of this configuration later in this article.



(a) Master: Arduino MKR Zero DK

(b) Slave: Arduino MKR GMS 1400 DK

Figure 4.1: Master slave embedded system configuration

Master-Slave handshake operation

The master increments a counter on each high-low transition of the phototransistor's collector voltage. This value is accumulated during 24 hours and then uploaded to a server that is accessible via the Internet. Since the upload is done by the slave, the counter value must be transferred to the slave (via I2C). Once this value has been successfully transferred, the slave sends a success code to the master and the counter is updated with the number of increments that occurred between the request to the slave and the reply from the slave.

In case the data could not be transferred within in a given budget of attempts, the slave sends a failure code. In either case the slave is shutdown by the master after the reply has been received. In case of failure, the master will not reset the counter and start another transmission attempt later. The slave might also get stuck while attempting to transmit data. In this case, the slave firmware makes no progress and no reply is sent to the master. To avoid a deadlock, the master will simply switch of the slave after a given delay and consider the transmission attempt as failed.

The transmission is done via the I2C bus. Once the slave is powered on, it will connect its I2C interface with the one of the master controller via analog switches. This allows data to be exchanged in both directions. The analog switches provide I2C bus isolation. Since the slave controller can be powered of while the master controller is powered on, the former would be back-powered via the I2C IO pins through the internal ESD diode. This situation can cause malfunctions and violates the Absolute Maximum Ratings of the SAMD21 microcontroller hosted on the Arduino DKs. This situation must therefore be avoided.

4.2 Module structure

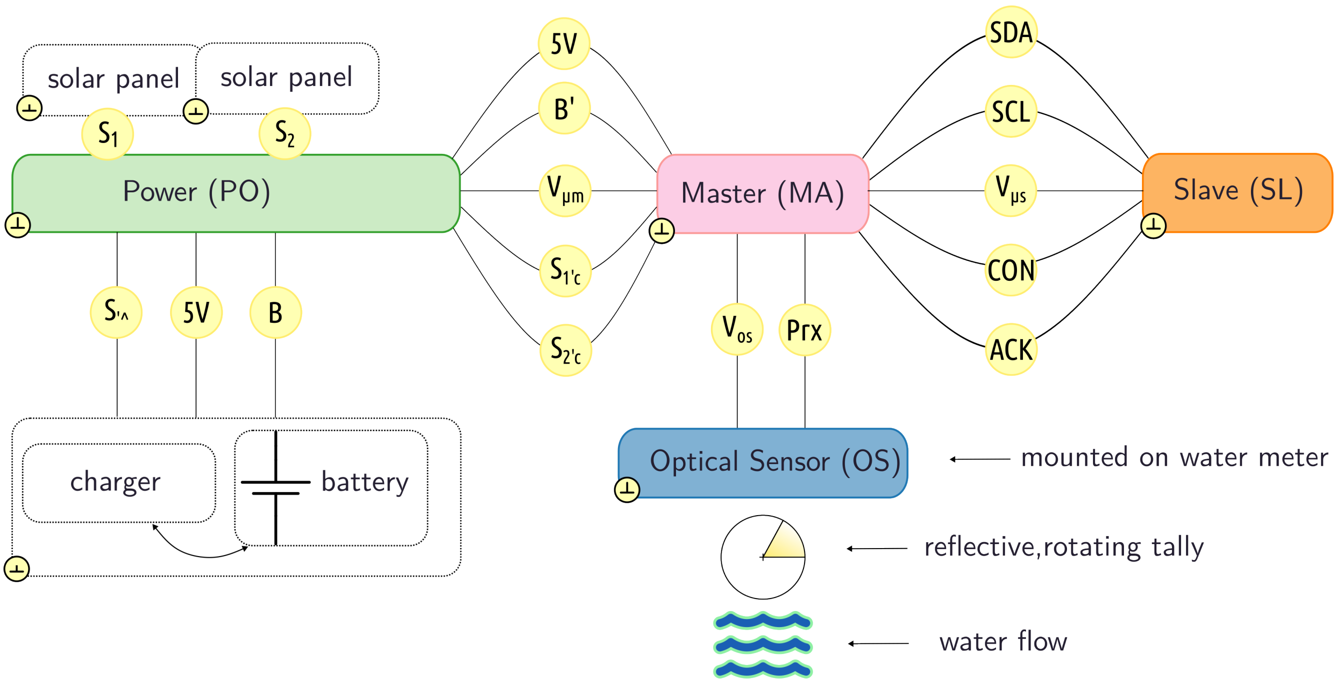
In the following section, we present the embedded system hardware as a set of modules. Modules can be composed of hardware components like resistors or integrated circuits (IC). A module can also contain other modules in a nested, tree-like structure.

Modules are interconnected with nets. Nets are labeled, for example *5V* or *B*. A net is labeled in the module where it first occurs in a top-down manner. It can then been referred to in a child module by prepending a dot to the label. For example, given that net B has been introduced in the top-level module, it will be referred to as .B in any child module. This notation is inspired from object-oriented programming. A net has also an Id and a Rank. The former is a sequential number starting at 1 that is local to each module. Rank is a hint on how important the net is when it comes to PCB layout. For example, nets carrying high-speed signals or supply power are usually higher ranked than a net that controls a user LED.

4.3 Remote Water Meter (RWM) system

The RWM system is composed of four top-level modules:

1. Master Module (MA): water flow measurement and SL power management.
2. Slave Module (SL): data transmission via GSM.
3. Power Module (PO): power source aggregation, signal conditioning and MA power management.
4. Optical Sensor (OS): signal conditioning of water flow indicator (tally).

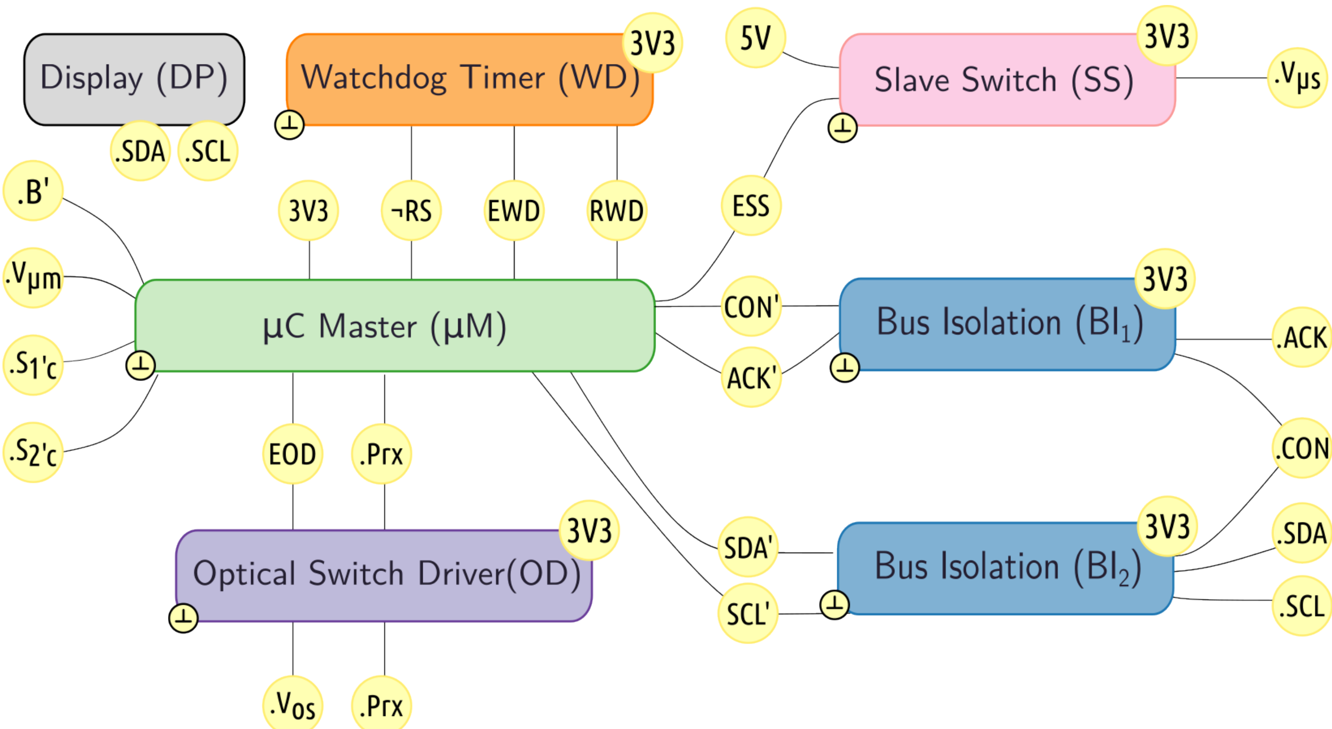


Id	Rank	Name	Net description
1	5	S_1	solar panel 1, positive output voltage
2	5	S_2	solar panel 2, positive output voltage
3	5	$S_{1'}$	V_{s1} protected against overvoltage
4	5	$S_{2'}$	V_{s1} protected against overvoltage
17	5	$S_{'\vee}$	$S_{1'}$ OR $S_{2'}$, OR-ing of dual solar power sources
5	2	5V	USB compatible charger output
6	1	B	battery voltage
6	1	B'	scaled down battery voltage for 3.3 V A/D conversion
7	3	$V_{\mu M}$	supply voltage for master microcontroller (μM) controlled by the Power Module (PO).
8	6	$S_{1'c}$	scaled down $V_{s1'}$ for 3.3 V A/D conversion
9	6	$S_{2'c}$	scaled down $V_{s2'}$ for 3.3 V A/D conversion
10	2	`SDA`	slave: I2C data
11	2	`SCL`	slave: I2C clock
12	2	$V_{\mu S}$	supply voltage for slave microcontroller
13	2	`CON`	slave: are slave I2C bus and master I2C connected ?
14	2	`ACK`	slave: is slave ready to receive I2C data from master ?
15	2	V_{OS}	supply voltage for Optical Switch (OS) module controlled by μM.
16	2	Prx	optical sensor output voltage (position of water meter tally) for A/D conversion by μC.

4.4 Master Module (MA)

The MA module is composed of six modules:

1. Display (DP): present process info to the user.
2. Watchdog Timer (WD): supervise program flow of μM.
3. Slave Switch (SS) : control power supply to Slave Module (SL).
4. μC Master (μM): run the water flow measurement firmware.
5. Bus Isolation (BI_1, BI_2): prevent back-powering via GPIO and I2C when SL is switched off.
6. Opto Switch Driver (OD): control power supply to the Optical Sensor Module (OS).



Id	Rank	Name	Net description
1	1	3.3V	voltage provided by μM
3	4	`ESS`	enable SS
4	4	`CON'`	slave: are slave I2C bus and master I2C connected ?
5	4	`ACK'`	slave: is slave ready to receive I2C data from master ?
6	4	`EOD`	enable OD
7	4	`¬RS`	watchdog timer reset due to a software or hardware defect
8	4	`EWD`	enable watchdog timer at the end of the startup code
9	4	`RWD`	reset ("kick") watchdog time to prevent a reset of U_1
10	4	`SDA`	master I2C data
11	4	`SCL`	master I2C clock
-			

Table 4.1: MA - Netlist

4.4.1 Display Module (DP)

The Display Module (DP) presents information on the ongoing metering process. This includes:

- timestamp: current date and time
- process state: accumulating, transmitting, requesting internet time, ...
- power: battery voltage, solar panel voltage
- SD-Card reader: current filename

Requirements

We noticed the importance of providing comprehensive diagnostic information at the location of deployment. The hardware is located inside a street gully which is exposed to humidity, dirt and mosquitos. When inspecting the system, the technician must be able to diagnose a potential problem at a glance. He must be able to quickly decide if the unit must be dismounted and sent to the lab. The display must therefore be clearly visible from every angle. This excludes LCD displays. Given the absence of grid-connection, low power consumption is mandatory. This excludes LED displays.

Implementation

We use a small, inexpensive OLED with very low power consumption of only 30 nA.

Id	Net	Pin mapping Nb.	Name	Type	Function
U_1	.3V3	4	`VCC`	⇀	
U_1	.GND	5	`GND`	⇀	
U_1	.SCL	6	`SCL`	↼	
U_1	.SDA	7	`SDA`	⇋	

Id	BOM Item	Order Code	Package	Rationale
U_1	*WPI438*	WPI438 / 59	DIL (4)	I2C interface

4.4.2 Watchdog Module (WD)

The Watchdog Module (WD) module prevents unexpected or intermittent software or hardware failures from stalling the microcontroller indefinitely. The module expects a timer reset signal from the microcontroller withing a given period. In the absence of this event, the watchdog circuit will reset the μC.

Requirements

The application runs only every second or so and the μC is in power-saving mode most of the time. There are no strict timing requirements but unnecessary watchdog resets must be avoided. This means that the watchdog circuit must be configurable for timeout periods of several seconds. This requirement excludes many available watchdog ICs.

1. supply voltage 3.3 V.
2. low power, low quiescent current.
3. timeout period of several seconds.

For my application, I choose a timeout period of roughly 8 s. This prevents any possible false resets due to variations in program execution time that might be introduced by future software updates.

Implementation

Id	Desc	Order Code	Package	Rationale
U_1	*TPS3431*	TPS3431SDRBR/681	VSON-8	wide range of timeout values[1]
R_p	100k	generic	0603	pull-up resistor[2]
C_c	100 nF, 16 V	generic	0603	value for ≈8 s timeout[3]
C_b	100 nF, 16 V	generic	0402	bypass cap, optional but recommended in datasheet

[1] and low quiescent current, active-low open drain output suitable for Arduino dev board.
[2] see *TPS3431*, 8.2.2.1 Calculating WDO Pullup Resistor Design 1.
[3] see *TPS3431*, table 8-3. [b]

Table 4.2: WD Module - BOM

Id	Issue	Potential solutions
1	Make watchdog reset persistent	add a flip-flop[1]

[1] the state of the flip-flop could then be transmitted to alert a human supervisor that the circuit has malfunctioned.

Table 4.3: WD - issues

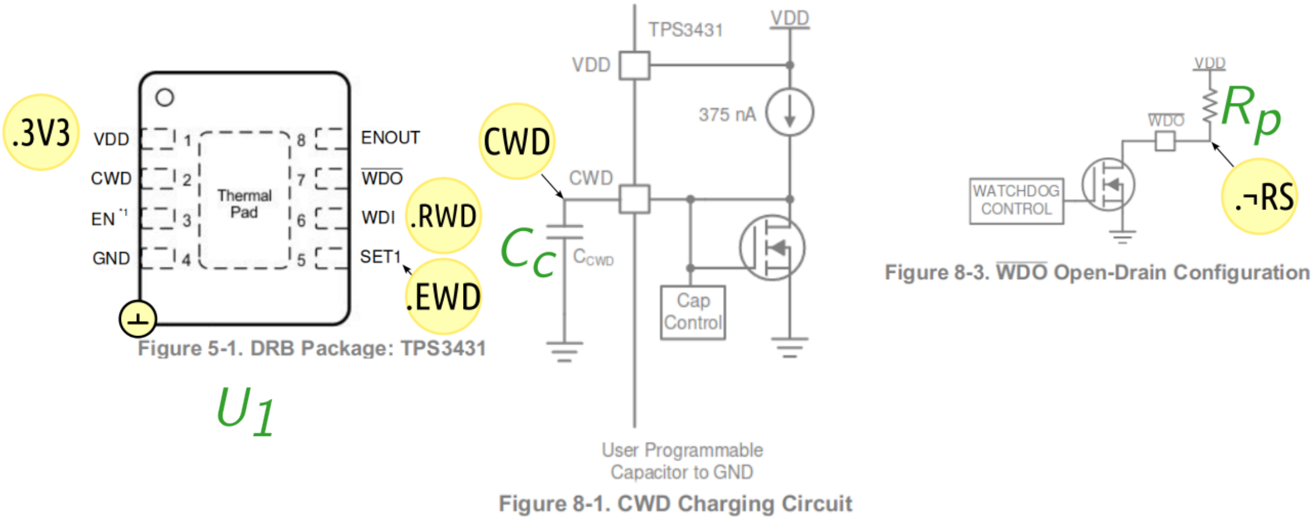


Figure 4.2: WD - schematic, from datasheet *TPS3431*

		pin			
Id	Net	Nb.	Name	Type	Function
U_1	.3V3	1	`VDD`	←	power supply
U_1	CWD	2	`CWD`	⇜	The timeout period is configured with C_c
U_1	⊥	4	`GND`	⊥	
U_1	.EWD	5	`SET1`	↼	enable timer
U_1	.RWD	6	`WDI`	↼	reset timer[1]
U_1	.¬RS	7	$\overline{\texttt{WDO}}$	⇀	pin pulled down if timer expired
U_1		3,8	`EN`, `ENOUT`		can be left floating for this application
C_c	CWD	1	`1`		
C_c	⊥	2	`2`	⊥	
R_p	.¬RS	1	`1`		open drain pullup
R_p	⊥	2	`2`	⊥	

[1] pulled down by μM in regular intervals < 8 s during normal operation.

Table 4.4: WD - Pin mapping

4.4.3 Slave Switch Module (SS)

The Slave Switch Module (SS) controls the power supply to the Slave Module (SL). As explained previously in section 4.1, the slave is powered only during transmission in order to reduce energy consumption mainly due to the relatively high standby current of the radio module.

Requirements

The load switch must be able to switch the maximum current for both master and slave. The maximum current is determined by the GMS module on the slave. We expect the maximum to not exceed 1 A.

Implementation

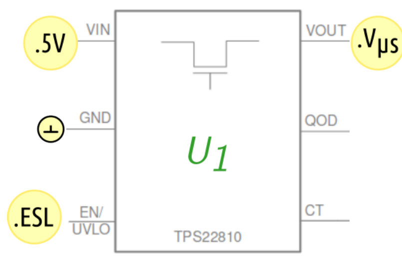


Figure 4.3: SS - schematic, based on datasheet *TPS22810*

		Pin mapping			
Id	Net	Nb.	Name	Type	Function
U_1	.5V	1	VIN	↼	input
U_1	⊥	2	`GND`	⊥	
U_1	.ESL	3	`EN/UVLO`	⇀	switch enable
U_1		4	`CT`		left floating[1]
U_1		5	`QOD`		left floating[2]
U_1	.V$_{\mu S}$	6	`VOUT`	⇀	output

[1] the Arduino is not a high current load for this switch.
[2] we don't care how fast the charge at the output decreases.

Id	BOM Item	Order Code	FF	Rationale
U_1	*TPS22810*	TPS22810DBVR/710	SOT-23/6	low R_{ON}[1], input voltage range down to 2.7 V, low quiescent current.

[1] The *Arduino MKR Zero* requires a 5 V power supply which is regulated to 3.3 V on the board. The Arduino board uses the *AP7115* voltage regulator. According to the datasheet, the dropout voltage is 200 mV. Therefore, the voltage supplied to the Arduino $V_{\mu M}$ must be greater than 3.5 V.

Table 4.5: SS - BOM

4.4.4 μMaster Module (μM)

μM is a microcontroller that runs software to acquire, accumulate and request transmission of water flow measurements.

Requirements

Almost any microcontroller is able to accomplish this task. Physical space is not critical, therefore we use a readily available low cost Arduino Development Kit (DK) DK, rather than designing a custom board. Given the absence of more specific requirements, I choose the *Arduino MKR Zero* .

Implementation

Id	BOM Item	Order Code	Package	Rationale
U_1	*Arduino MKR Zero*		DIL (28)	availability, ease of use

Table 4.6: μM - BOM

Id	Issue	Potential solutions
1	U_1 has relatively high power consumption.[1]	low power μC (MSP430FR2311IPW16) and custom charger circuit.
2	U_1 requires a stable 5 V power supply.[2]	see issue 1

[1] This is due to the microcontroller itself (SAM D21) and to the peripheral components (battery charger).
[2] Given that a 3.7 V LiPo battery is used, this requires a boost conversion which in turn implies losses.

Table 4.7: μM - Issues

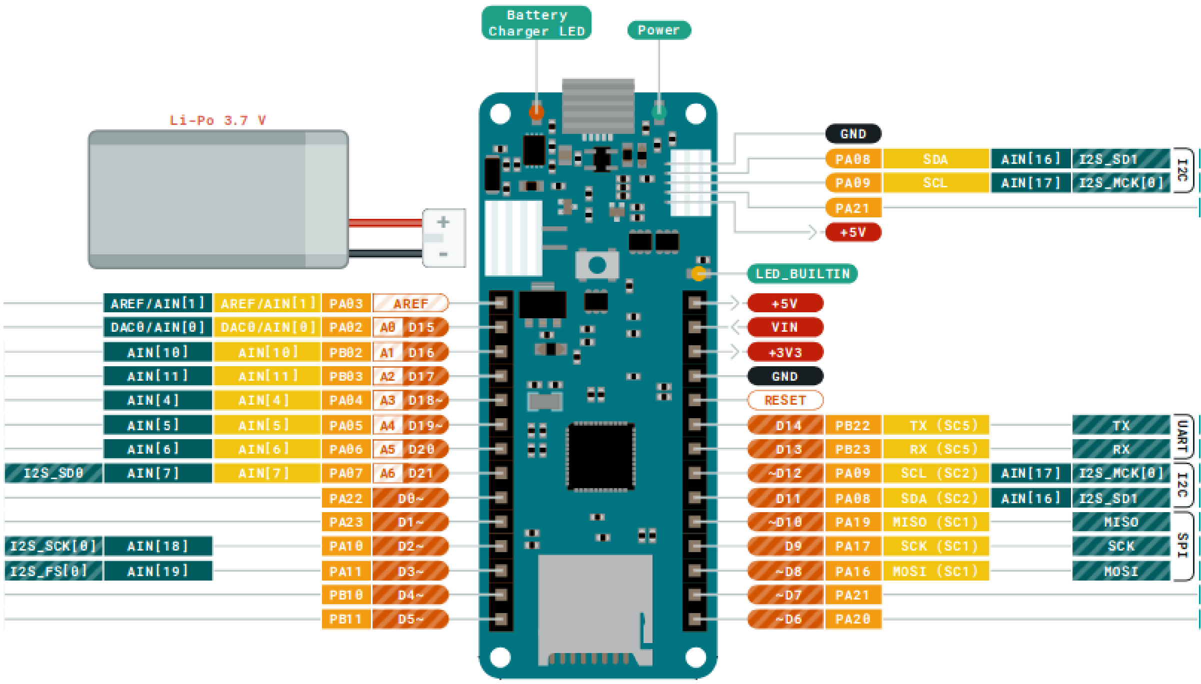


Figure 4.4: μM - pin out *Arduino MKR Zero*

		Pin mapping			
Id	Net	Nb.	Name	Type	Function
U_1	.Prx	2	`A0`	⇜	A/D conversion of OS output voltage
U_1	ESS	9	`D0`	⇀	enable SS
U_1	ACK'	10	`D1`	↽	.ACK connected via BI_1
U_1	EOD	11	`D2`	⇀	enable OD
U_1	EWD	12	`D3`	⇀	enable WD
U_1	RWD	13	`D4`	⇀	reset WD timer
U_1	¬.PB_1	14	`D5`	↽↾	user push button
U_1	.B'	16	`A1`	⇜	A/D conversion of scaled down battery voltage (<3.3 V)
U_1	.$S_{1'c}$	17	`A2`	⇜	A/D conversion of scaled down solar voltage (<3.3 V)
U_1	.$S_{2'c}$	18	`A3`	⇜	A/D conversion of scaled down solar voltage (<3.3 V)
U_1	.SDA	20	`SDA`	⇋	.SDA connected via BI_2
U_1	.SCL	21	`SCL`	⇋	.SCL connected via BI_2
U_1	¬RS	24	`RESET`	↽	pulled down by WD timer or user button in case of timeout
U_1	⊥	25	`GND`	⊥	
U_1	3V3	26	`VCC`	→	regulated output voltage[1]
U_1	.$V_{\mu M}$	27	`VIN`	←	unregulated input voltage > 3.5 V [2]

[1] [...] This pin outputs 3.3V through the on-board voltage regulator. This voltage is the same regardless the power source used (USB, Vin and Battery) (*Arduino MKR Zero*).

[2] [...] This pin can be used to power the board with a regulated 5V source. If the power is fed through this pin, the USB power source is disconnected. This is the only way you can supply 5 V (range is 5V to maximum 6V) to the board not using USB. (*Arduino MKR Zero*).

4.4.5 Bus Isolation Module (BI)

The Bus Isolation Module (BI) allows the slave to interconnect its GPIO pins with equivalent pins of the μM module. These pins are not designed to support voltages > 0 when the microcontroller is powered off. Since the slave microcontroller is powered on only during the transmission phase, the pins configured for I2C would effectively be back-powered via the internal ESD protection diodes.

Specification

The switch must have the following characteristics:

- low power: this excludes mechanical switches like reed relays
- bidirectional
- 3.3V logic compatible

Implementation

Id	BOM item	Order Code	FF	Rationale
U_1	*TS5A23157*	TPD3S014-Q1/176	VSSOP/10	low Ron

Table 4.8: BI - BOM

Id	Issue	Potential solutions
1	$U_{1,2}$ can not be back-powered.[1]	TMUX1574RSVR (TI)

[1] Usually, the slave is not powered on when the master is powered off because the load switch for the slave power supply requires a logic H from the master to be on. However, when flashing (programming) the slave, the master could be switched off and in this case the supply voltage of $U_{1,2}$ if 0 V.

Table 4.9: μM - Issues

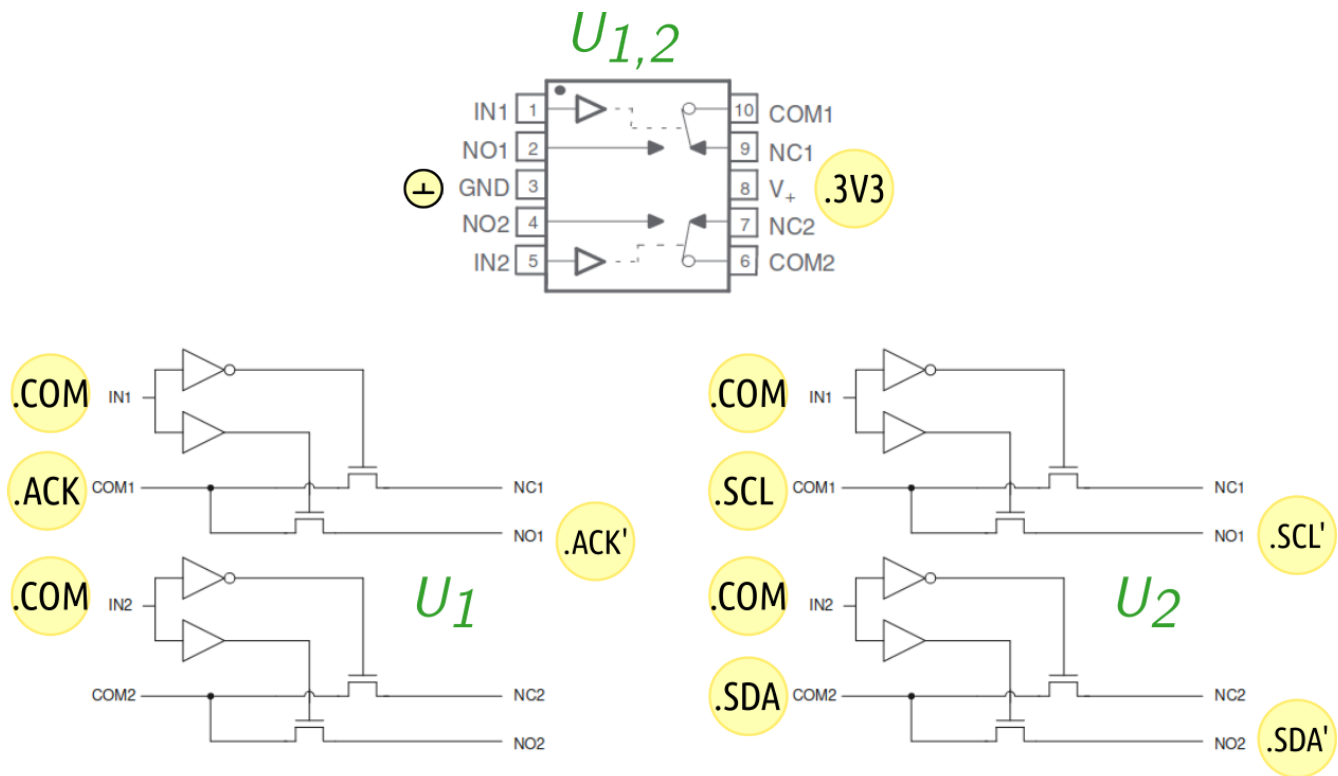


Figure 4.5: BI - schematic, based on datasheet *TS5A23157*

		Pin mapping			
Id	Net	Nb.	Name	Type	Function
U_1	.COM	1	`IN1`	$\leftharpoonup$	switch 1 enable, driven by μS
U_1	.ACK'	2	`NO1`	$\rightharpoonup$	normally open, switch 1 output
U_1	$\perp$	3	`GND`	$\perp$	$\perp$
U_1	.COM	5	`IN2`	$\leftharpoonup$	switch 2 enable, switch 2 not used
U_1	.3V3	8	`V+`	$\leftarrow$	power supply
U_1	.ACK	10	`COM1`	$\leftharpoonup$	switch 1 input
U_2	.COM	1	`IN1`	$\leftharpoonup$	switch 1 enable, driven by μS
U_2	.SCL'	2	`NO1`	$\rightharpoonup$	normally open, switch 1 output
U_2	$\perp$	3	`GND`	$\perp$	$\perp$
U_2	.SDA'	4	`NO2`	$\rightharpoonup$	$\perp$
U_2	.COM	5	`IN2`	$\leftharpoonup$	switch 2 enable, switch 2 not used
U_2	.SDA	6	`COM2`	$\leftharpoonup$	switch 2 enable, switch 2 not used
U_2	.3V3	8	`V+`	$\leftarrow$	power supply
U_2	.SCL	10	`COM1`	$\leftharpoonup$	switch 1 input

4.4.6 Opto Switch Driver Module (OD)

The Opto Switch Driver Module (OD) is an interface between module μM and the Opto Switch Module (OS).

Requirements

Experiments have shown that at least 50 mA are required to obtain clearly separated voltage levels at the output of OS (.Prx).

Implementation

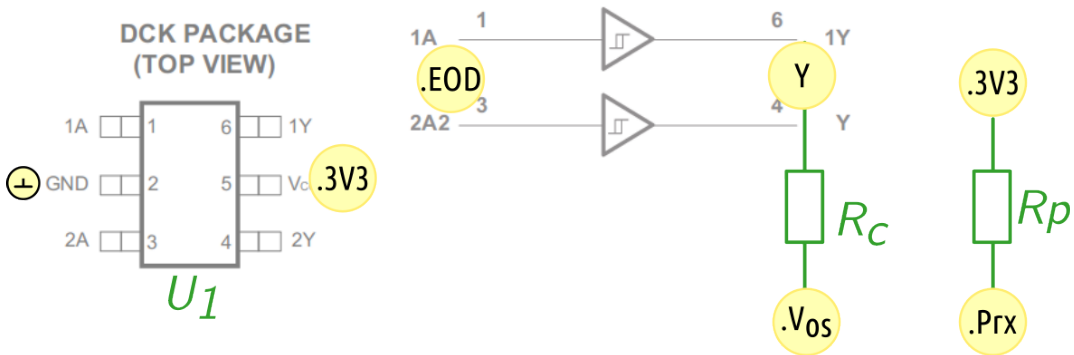


Figure 4.6: OD - schematic, based on datasheet *SN74LVC2G17*

		Pin mapping			
Id	Net	Nb.	Name	Type	Function
U_1	.EOD	1	`1A`	↼	buffer input controlled by μC
U_1	.EOD	3	`2A`	↼	buffer input controlled by μC
U_1	⊥	2	`GND`	⊥	
U_1	Y	4	`1Y`	⇀	buffer output
U_1	.3V3	5	`VCC`	←	power supply
U_1	Y	6	`2Y`	⇀	buffer output
R_c	Y	1	`1`		
R_c	$.V_{OS}$	2	`2`		anode voltage of photodiode
R_p	.3V3	1	`1`		upper rail for pullup resistor
R_p	.Prx	2	`2`		collector voltage of phototransistor

Id	Issue	Potential solution
1	power consumption	current source[1]

[1] The current implementation is not ideal. It would be better to use a (programmable) current source to avoid burning power in the current limiting resistor. In addition, the optimum output current remains to be determined.

Table 4.10: OD - issues

Id	Desc	Order Code	FF	Rationale
U_1	*SN74LVC2G17*	SN74LVC2G17DCKR/473	SC70/6	high output power, parallel output[1]
R_p	1 kΩ	generic	0603	pull-up resistor[2]
R_c	20 Ω	generic	0603	current limit resistor [3]
C_b	100 nF, 16 V	generic	0402	bypass cap

[1] for a total rated maximum of 48 mA, low power consumption, the Smitt-Trigger inputs are not really necessary for this application.

[2] We should explain why we choose this value. But as mentioned in Issue 1, we are likely to replace the current limit resistor with a current source in the near future.

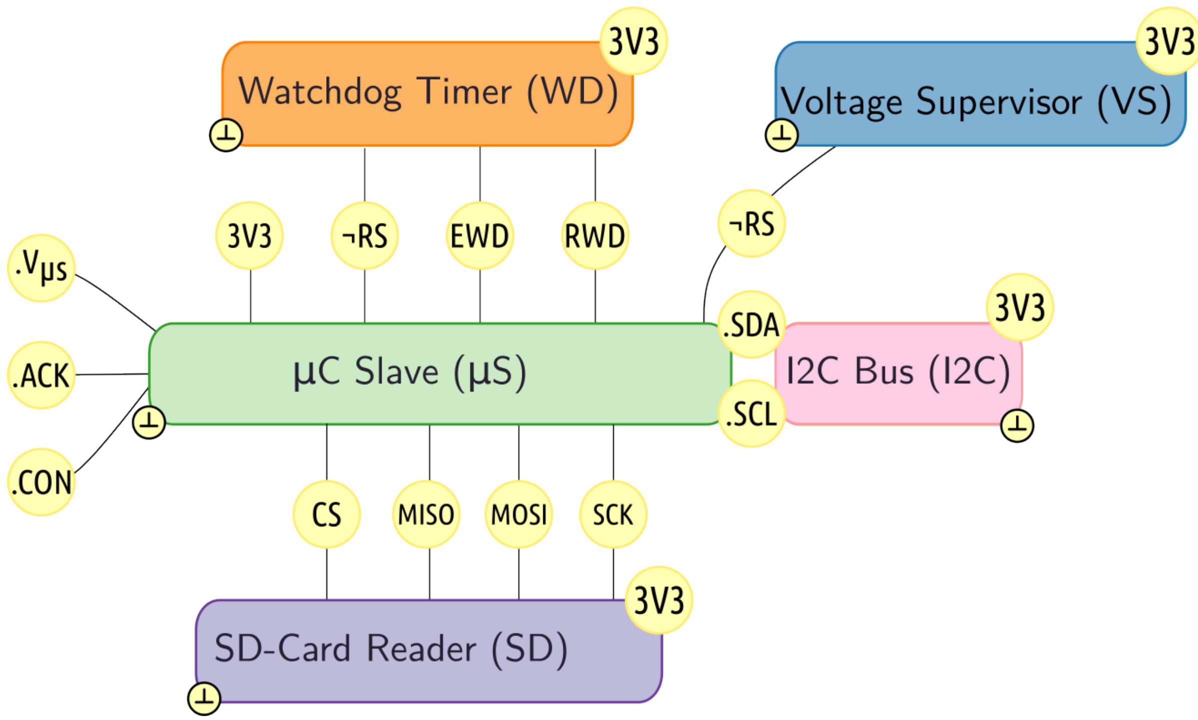
[3] idem

Table 4.11: OD - BOM

4.5 Slave Module (SL)

The SL module is composed of five modules:

1. μC Slave (μS): transmit measurement data via GSM.
2. Watchdog Timer (WD): supervise program flow of μS.
3. Voltage Supervisor (SS) : supervise supply voltage of μS.
4. I2C Bus (I2C): implement I2C bus requirements (pullup resistors).
5. SD-Card Reader (SD): log process data to SD-card.



Id	Rank	Name	Net description
1	1	3.3V	regulated voltage provided by μS
2	4	`ESS`	enable SS
3	4	`CON`	H: μS I2C bus and μM I2C bus are connected
4	4	`ACK`	H: μS is ready to receive I2C data from μM
5	4	`¬RS`	reset by watchdog timer, voltage supervisor or on-board user button
6	4	`EWD`	enable watchdog timer at the end of the startup code
7	4	`RWD`	reset ("kick") watchdog time to prevent a reset of U_1
8	4	`SDA`	I2C data
9	4	`SCL`	I2C clock
10	4	`CS`	SPI, Chip select
10	4	`MISO`	SPI, MISO
10	4	`MOSI`	SPI, MOSI
10	4	`SCK`	SPI, SCK

4.5.1 μSlave Module (μS)

μS is a microcontroller that receives measurement data via the I2C bus and uploads this data to a public mqtt broker via GSM (GPRS).

Requirements

Originally, I planed to use only the *MKR GSM* DK to realize the entire application. It turned out however, that I was unable to put the u-blox GSM module hosted on the DK into low power mode. While I was able to obtain some power consumption reduction via the Arduino GSM library, this was by far not enough for a low power application. I decided therefore to split the functionality across two DKs: one who does the actuals water flow measurement - the master - and another separate GSM enabled module - the slave - to perform actual data transmission. The master would then control the slave power supply such that the slave and the radio module would only draw current during the relatively short period required for data transmission. Hence, the requirements for μS are:

1. bidirectional data flow between master and slave.
2. GSM/GPRS compatible modem.

The *MKR GSM* provides an I2C interface and fulfills both requirements. Another solution would have been to simply find a UART-compatible radio module (without additional microcontroller). While this would have resulted in a simpler and more compact circuit, I would have had to adapt the Arduino GSM library or write one from scratch.

Implementation

Id	BOM Item	Order Code	Package	Rationale
U_1	*MKR GSM*		DIL (28)	availability, ease of use

Table 4.12: μS - BOM

Id	Issue	Potential solution
1	U_1: GMS 3 only, chipset obsolete	select a GSM LTE or LTE-M generation module

Table 4.13: μS - Issues

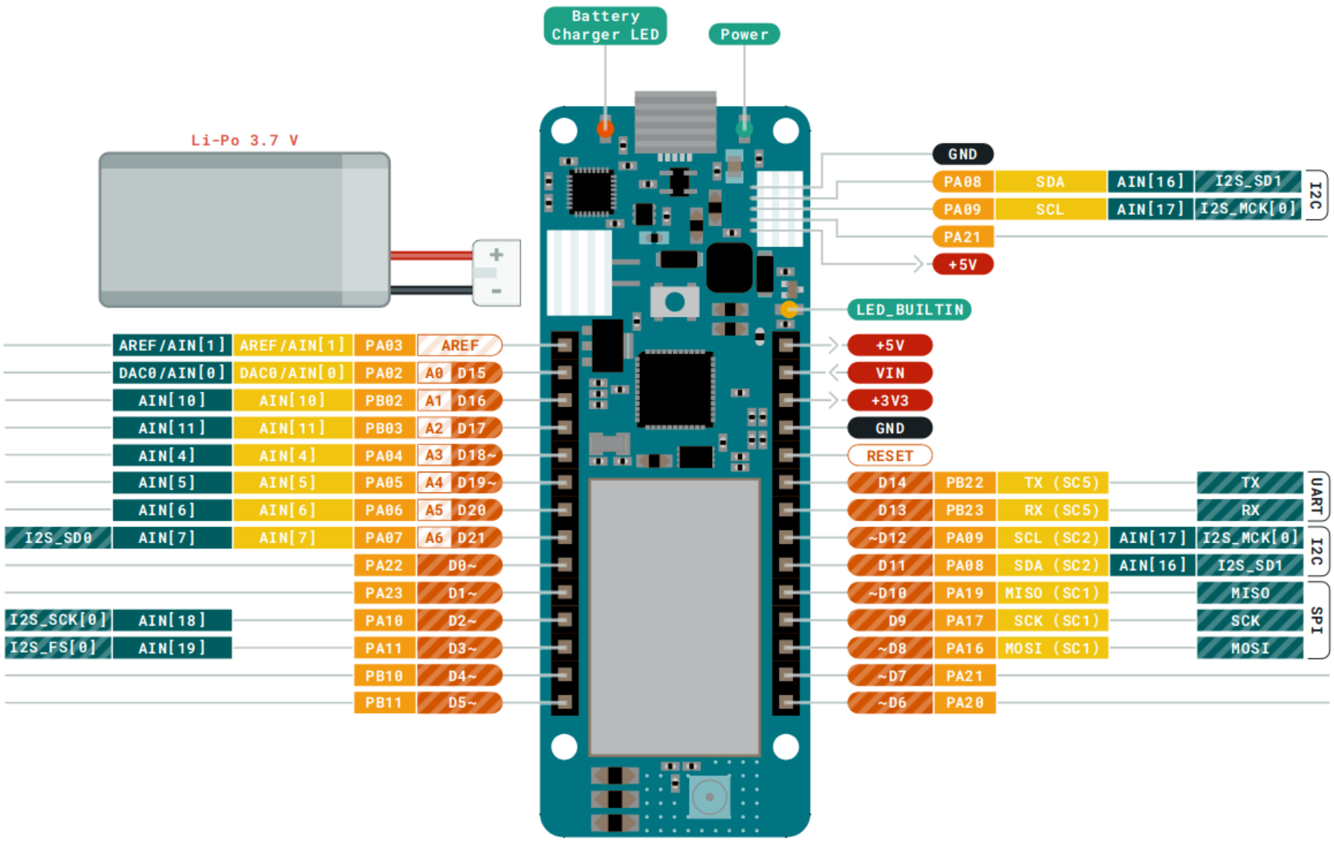


Figure 4.7: μS - pin out *MKR GSM*

		Pin mapping			
Id	Net	Nb.	Name	Type	Function
U_1	.CON	9	`A0`	$\rightharpoonup$	high: connect I2C bus to master (bypass the isolation barrier)
U_1	.ACK	10	`D0`	$\rightharpoonup$	high: signal to master that I am ready to receive data
U_1	.CS	11	`D1`	$\rightharpoonup$	
U_1	EWD	12	`D5`	$\rightharpoonup$	enable WD
U_1	RWD	13	`D5`	$\rightharpoonup$	reset WD
U_1	.MISO	17	`D2`	$\leftharpoondown$	
U_1	.SCK	18	`D3`	$\rightharpoonup$	
U_1	.MOSI	19	`D4`	$\rightharpoonup$	
U_1	.SDA	20	`SDA`	$\leftrightharpoons$	
U_1	.SCL	21	`SCL`	$\rightharpoonup$	
U_1	¬RS	24	`RESET`	$\leftharpoondown$	pulled down by WD timer or user button in case of timeout
U_1	$\perp$	25	`GND`	$\perp$	
U_1	.3V3	26	`VCC`	$\rightarrow$	regulated output voltage[1]
U_1	.V$_{\mu S}$	27	`VIN`	$\leftarrow$	input voltage controlled by μM

4.5.2 Watchdog Module (WD)

This module is identical to the one discussed in 4.4.2.

4.5.3 I2C Module (I2C)

This module consists simply of two pullup resistors as required by the I2C interface specification. I choose to dedicate a separate module to this simple circuit for a couple of reaons:

- There is some discussion on how these resistors should be choosen, dependent on the bus speed (TODO).
- There is a lot of documentation available for the I2C bus and I wanted a nice home for this material.
- I plan to factor out this module into a standalone library that can be shared amongst many projects.

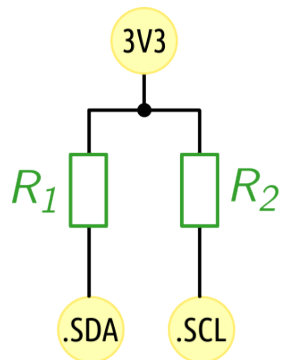


Figure 4.8: I2C pullup resistors

Id	BOM Item	Order Code	Package	Rationale
R_1	10k	generic	0603	commonly suggested value
R_2	10k	generic	0603	commonly suggested value

Table 4.14: I2C - BOM

4.5.4 SD-Card Reader Module (SD)

The SD-Card Reader Module (SD) allows to write debug data to a standard SD-card. All the information listed in 4.4.1 is made persistent for further analysis. Doing so via GSM transmission would be too costly both in terms of energy consumption and in service usage fees.

Requirements

SD must offer a SPI or UART interface because the I2C bus is already used for master-slave communication. The operating voltage must be either 3.3 V or 5 V.

Implementation

We use an Arduino-compatible MicroSD Card shield available on various online retailers.

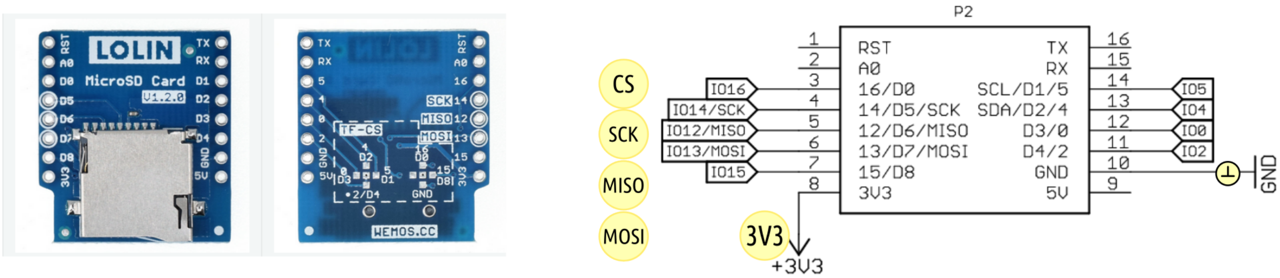


Figure 4.9: SD - schematic

		Pin mapping			
Id	Net	Nb.	Name	Type	Function
U_1	.SCK	4	D5	⇀	
U_1	.MISO	5	D6	⇀	
U_1	.MOSI	6	D7	↽	
U_1	.CS	7	D8	↽	
U_1	.3V3	8	D8	←	
U_1	⊥	10	GND	⊥	

Table 4.15: SD - pin mapping

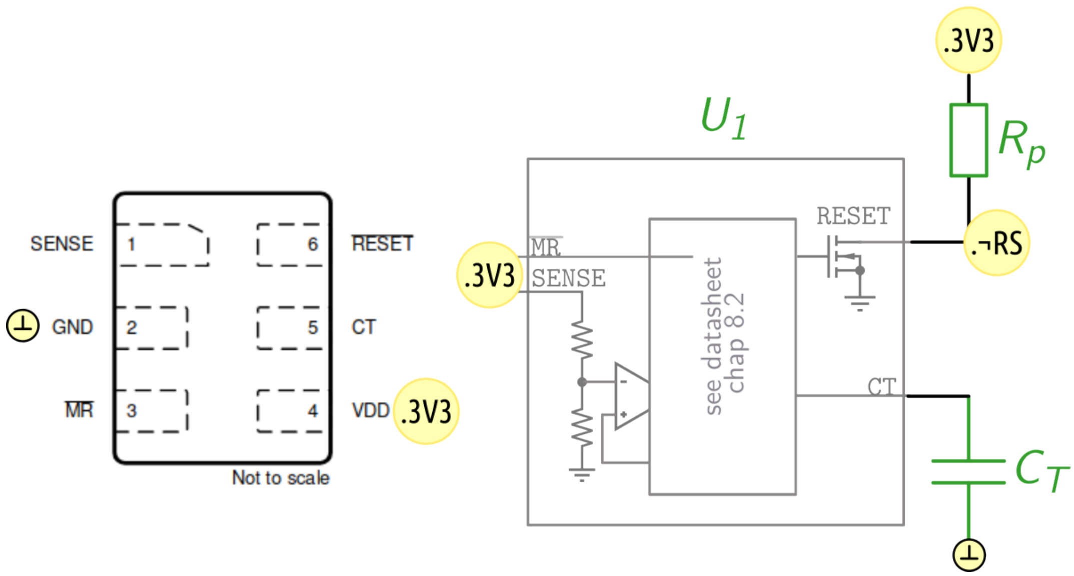
4.5.5 Voltage Supervisor (VS)

The Voltage Supervisor Module (VS) keeps the μS in reset as long as the supply voltage undershoots or overshoots the recommended operating conditions. When the master switches on the slave power supply, a certain voltage drop is to be expected due to inrush current. This drop increases as the battery charge decreases. Furthermore, the drop depends on the ambient temperature. VS ensures that μS does not attempt to boot until the supply voltage has recovered. Failing to do so could lead to undefined behavior.

Requirements

VS must be suitable for 3.3 V systems.

Implementation



Id	Net	Pin mapping			
		Nb.	Name	Type	Function
U_1	.3V3	1	`SENSE`	⇜	
U_1	⊥	2	`GND`	⊥	
U_1	¬MR	3	$\overline{\texttt{MR}}$	↼	manual reset
U_1	.3V3	4	`VDD`	←	
U_1	CT	5	`CT`	⇜	adjustable reset delay time
U_1	¬RS	6	$\overline{\texttt{RESET}}$	↼	reset output open drain

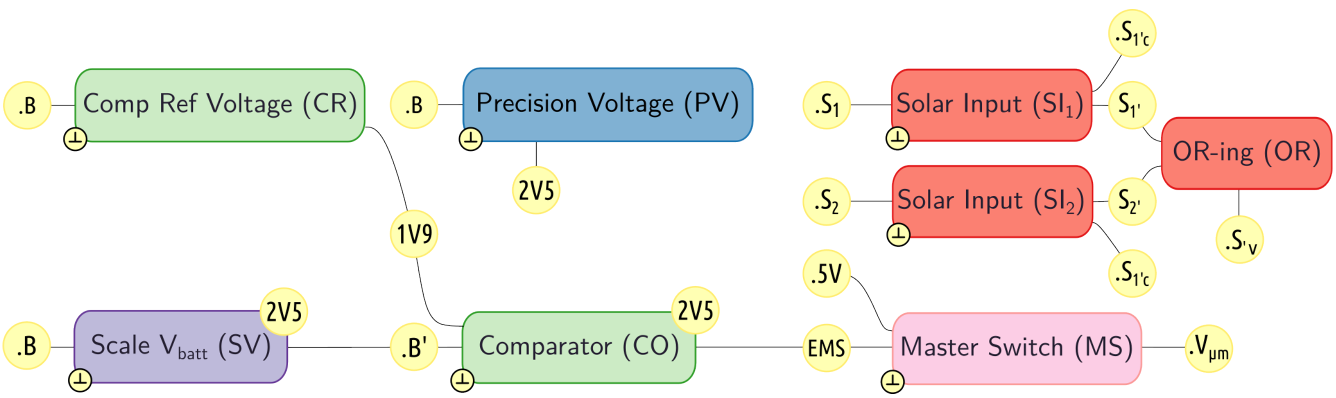
Id	BOM Item	Order Code	FF	Rationale
U_1	*TPS3890*	TPS389033DSER / 235	WSON/6	
R_p	10 kΩ	generic	0603	
C_t	100 nF	generic	0603	1 s delay

Table 4.16: VS - BOM

4.6 Power distribution and monitoring Module (PO)

The PO module is composed of eight modules:

1. Comp Ref Voltage (CR): provide regulated 1.9 V supply rail.
2. Precision Voltage (PV): provide regulated 2.5 V supply rail.
3. Solar Input (SI_1): filter and scale solar panel output.
4. Solar Input (SI_2): filter and scale solar panel output.
5. OR-ing (OR): combine outputs from ($SI_{1,2}$).
6. Scale V_{batt} (SV): scale down battery voltage suitable for a 2.5 V supply rail.
7. Comparator (CO): Schmitt-Trigger with hysteresis.
8. Master Switch (MS): control power supply of μM.



Id	Rank	Name	Net description
1	1	1V9	reference voltage for comparator threshold
2	4	2V5	supply voltage for low voltage circuits[1]
3	4	S_1	ESD protected voltage from solar panel 1
4	4	S_2	ESD protected voltage from solar panel 2

[1] This voltage must be lower than the minimum battery voltage.

Table 4.17: PO - NetList

4.6.1 Comp Ref Voltage Module (CR)

The Comp Ref Voltage Module (CR) provides a regulated 1.9 V supply rail for CO. As we shall see later, this voltage sets the amount of hysteresis applied to the switching threshold of the comparator.

Requirements

The reference voltage is determined by the computation shown in 4.6.6.

Implementation

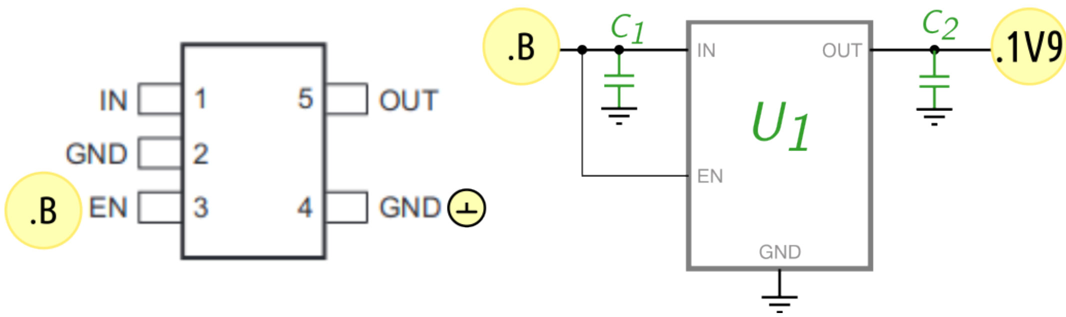


Figure 4.10: CR - schematic, from datasheet *TPS783xx*

		pin			
Id	Net	Nb.	Name	Type	Function
U_1	.B	1	`IN`	←	input
U_1	⊥	2	`GND`	⊥	
U_1	EN	3	`EN`	↼	enable output
U_1	⊥	4	`GND`	⊥	
U_1	.1V9	5	`OUT`	→	output

Table 4.18: WD - Pin mapping

Id	Desc	Order Code	Package	Rationale
U_1	*TPS783xx*	TPS78319DDCR/922	SOT-23-THIN-5	
C_1	1 µF, 16 V	generic	0603	
C_2	1 µF, 16 V	generic	0603	

Id	Issue	Potential solutions
1	The choice of 1.9V is not ideal, the threshold is too high.	move the comparison function to the μC
1		use a scaling module like SV for more control

Table 4.19: CR - issues

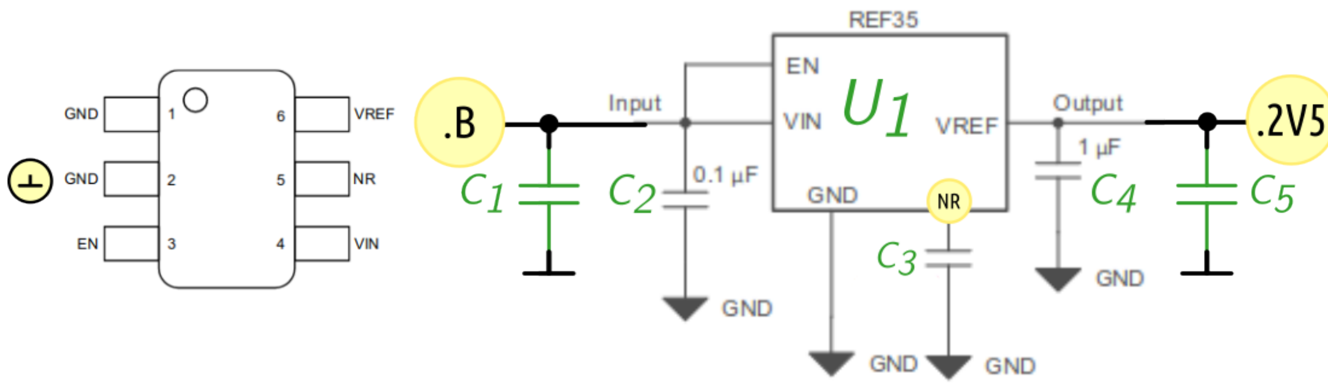
4.6.2 Precision Voltage Module (PV)

The Precision Voltage Module (PV) provides a regulated 2.5 V supply rail for SV and CO.

Requirements

The precision of PV has an impact on the upper threshold of CO. Given that the ambient temperature varies between 0 °C and 30 °C the temperature drift should be small.

Implementation



		pin			
Id	Net	Nb.	Name	Type	Function
U_1	⊥	1	`GND`	⊥	
U_1	⊥	2	`GND`	⊥	
U_1	EN	3	`EN`	↼	enable output
U_1	.B	4	`VIN`	←	input
U_1	NR	5	`NR`	↭	noise reduction
U_1	.2V5	6	`VREF`	→	output
C_1	.B	1	1		
C_1	⊥	2	2	⊥	
C_2	.B	1	1		
C_2	⊥	2	2	⊥	
C_3	NR	1	1		
C_3	⊥	2	2	⊥	
C_4	.2V5	1	1		
C_4	⊥	2	2	⊥	
C_5	.2V5	1	1		
C_5	⊥	2	2	⊥	

Id	Desc	Order Code	Package	Rationale
U_1	TI, *REF35*	REF35250QDBVR/921	SOT-23-6	
C_1	1 µF, 16 V	generic	0603	
C_2	100 nF, 16 V	generic	0603	
C_3	100 nF, 16 V	generic	0603	
C_4	1 µF, 16 V	generic	1206	
C_5	10 nF, 16 V	generic	0603	

Table 4.20: PV - BOM

Id	Issue	Potential solutions
1	The choice of 1.9V is not ideal, the threshold is too high	move the compairison function to the μC

Table 4.21: CR - issues

4.6.3 Solar Input Module (SI)

The Solar Input Module (SI) filters and scales the output voltage of the solar panels. Given the physical distance between panel and SI of approx. 20 m, the inputs are susceptible to overvoltage due to atmospheric disturbances. The solar panel output voltage should also be digitized by μM.

Requirements

The voltage applied to the analog input of μM must not exceed 3.3 V.

Implementation

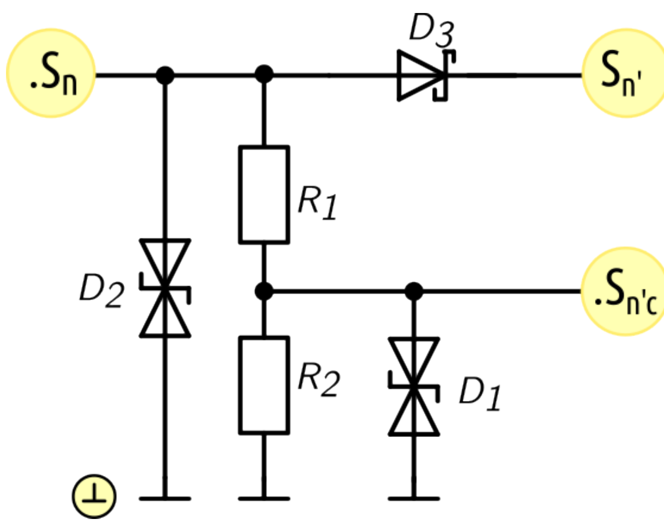


Figure 4.11: SI - schematic, n = 1,2

		pin			
Id	Net	Nb.	Name	Type	Function
D_1	.Sn'c	1	1	⊥	
D_1	⊥	2	2	⊥	
D_2	.Sn	1	1	⊥	
D_2	⊥	2	2	⊥	
D_3	.Sn	1	A		
D_3	.Sn'	2	C		
R_1	.Sn	1	1	←	
R_1	.Sn'c	2	2	↭	
R_2	.Sn'c	1	1	→	
R_2	⊥	2	2		

Id	Desc	Order Code	Package	Rationale
D_1	*CDSOD323*	CDSOD323-T03SC/820	SOD-323	
D_2	*CDSOD323*	CDSOD323-T36SC/945	SOD-323	
D_3	*CDBA340L*	CDBA340L-G/618	DO-214AC	
R_1	$180\,k\Omega$	generic	0603	
R_2	$20\,k\Omega$	generic	0603	

Id	Issue	Potential solutions
1	reverse current of D_3 is too high	make further tests

Table 4.22: SI - issues

4.6.4 OR-ing Module (OR)

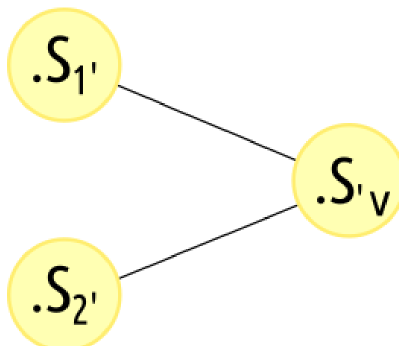
The OR-ing Module (OR) combines the outputs of SI_1 and SI_2.

Requirements

The contribution of both panels should be summed.

Implementation

$S_{1'}$ and .$S_{2'}$ are connected.



Id	Issue	Potential solutions
1	only one panels contributes at a given time	separate charger units

Table 4.23: OR - issues

LiPo Charger

Net .S'$_V$ is connected to `SOLAR IN +` of the charger shown in Fig.4.12.

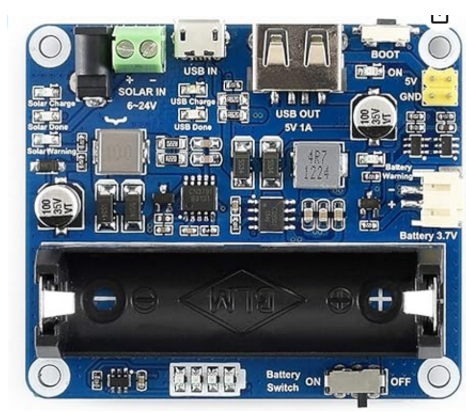


Figure 4.12: Waveshare solar power LiPo charger

4.6.5 Scale V_{batt} Module (SV)

The Scale V_{batt} Module (SV) scales the battery voltage to make it compatible with the operation voltage of μM (3.3 V).

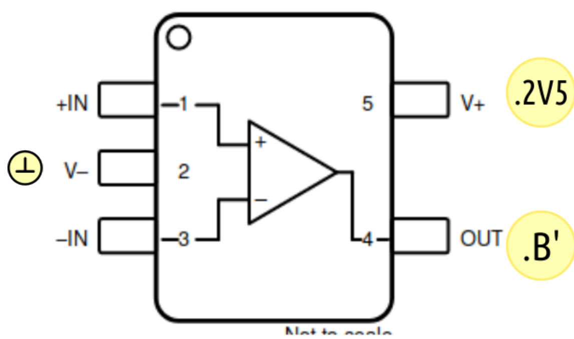
Requirements

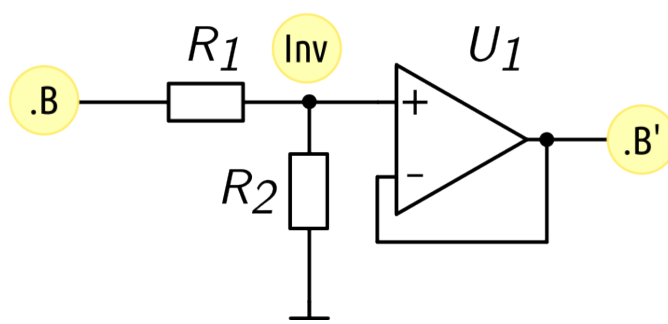
The battery voltage `.B` must be scaled such that the maximum value is close to the upper input voltage limit of SV. We suppose that the maximum voltage of a one-cell LiPo battery does not exceed 4.1 V. Let us recall that SV must operate from 2.5 V which is lower than the minimum battery voltage. There is some debate on how low LiPo batteries can be discharged. Some argue that 2.5 V can be set a lower threshold. We have decided to not got beyond 3 V until we have carried out our own measurements with our batteries.

Implementation

We choose U_1 with an input common mode range of ±100 mV beyond rail. That means we can map the maximum battery voltage to the supply rail of the op amp (2.5 V) while still keeping some margin.

The scale coefficient is thus $k = \frac{2.5V}{4.1V} = 0.61$. I choose $R_1 = 129\,\text{k}\Omega, R_1 = 200\,\text{k}\Omega, k = 0.645$. The maximum voltage at the non-inverting input of the buffer is therefore $V_{max} = 0.645 \cdot 4.1\,\text{V} = 2.64\,\text{V}$.





		pin			
Id	Net	Nb.	Name	Type	Function
U_1	Inv	1	`+IN`	⇜	input
U_1	⊥	2	`V-`	⊥	
U_1	.B’	3	`-IN`	⇜	inverting input
U_1	.B’	4	`OUT`	⇝	output
U_1	.2V5	5	`V+`	←	power supply
R_1	.B	1	`OUT`		
R_1	Inv	2	`OUT`		
R_2	Inv	1	`OUT`		
R_2	⊥	2	`OUT`		

Id	Desc	Order Code	Package	Note
U_1	TI, *OPAx391*	OPA391DCKR/931	SC70-5	
R_1	129 kΩ	RN73H2ATTD1293B25/52	0603	*RN73H*, 0.1 %
R_2	200 kΩ	RN73H1JTTD5693B50/59	0603	*RN73* , 0.1 %
C_b	100 nF, 16 V	generic	0402	bypass cap

Table 4.24: SV - BOM

4.6.6 Comparator Module (CO)

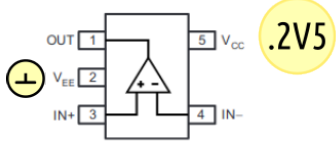
The Comparator Module (CO) maps the scaled battery voltage to a digital signal. A logical H applies power to μM.

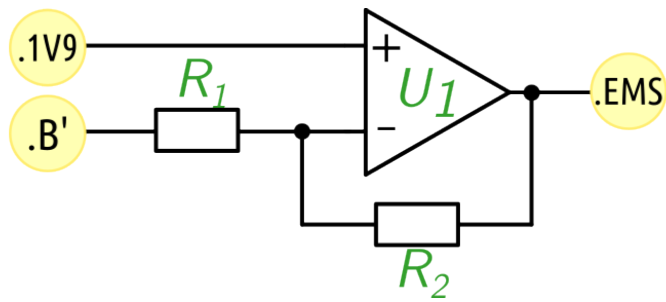
Requirements

We choose the lower threshold of the comparator to be equivalent to a battery voltage of 3 V. Since we scaled this voltage by a factor of 0.645 (see 4.6.5), the lower threshold for the comparator is then set to $V_L = 0.645 \cdot 3.0\,\text{V} = 1.94\,\text{V}$.

We choose the upper threshold of the comparator to be equivalent to a battery voltage of 3.9 V. The upper threshold for the comparator is then set to $V_H = 0.645 \cdot 3.9\,\text{V} = 2.52\,\text{V}$.

Implementation





		pin			
Id	Net	Nb.	Name	Type	Function
U_1	.B	1	`IN`	←	input
U_1	⊥	2	`GND`	⊥	
U_1	EN	3	`EN`	↼	enable output
U_1	⊥	4	`GND`	⊥	
U_1	.1V9	5	`OUT`	→	output

Table 4.25: WD - Pin mapping

Id	Desc	Order Code	Package	Note
U_1	*TLV703x*	TLV7031DCKT/923	SC70-5	
R_1	129 kΩ	RN73H2ATTD1293B25/52	0603	*RN73H*, 0.1 %
R_2	569 kΩ	RN73H1JTTD5693B50/55	0603	*RN73H* , 0.1 %
C_b	100 nF, 16 V	generic	0402	bypass cap

Configuring thresholds for the non-inverting comparator

U_1 is a non-inverting comparator with hysteresis. R_1 and R_2 together with the bias voltage 1.9 V select the lower and upper tripping voltages. The computation of thresholds for the non-inverting comparator configuration with hysteresis is shown in several documents produced by TI:

1. equation (4) in datasheet *TLV703x*: this equation seems to be incorrect.
2. TI application notes SBOA313A and TIDU020A.

I will use the standard approach to circuit analysis with Kirchhoff Voltage Law (KVL):

We want the hysteresis function shown in Fig 4.13 with lower voltage threshold V_L and higher voltage threshold V_H:

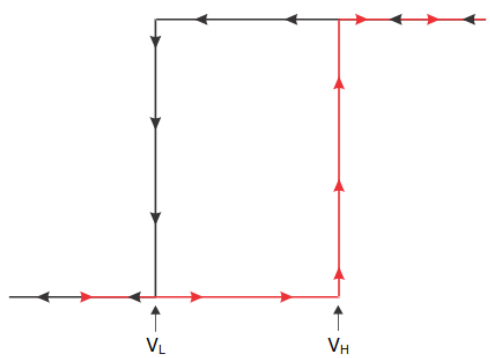
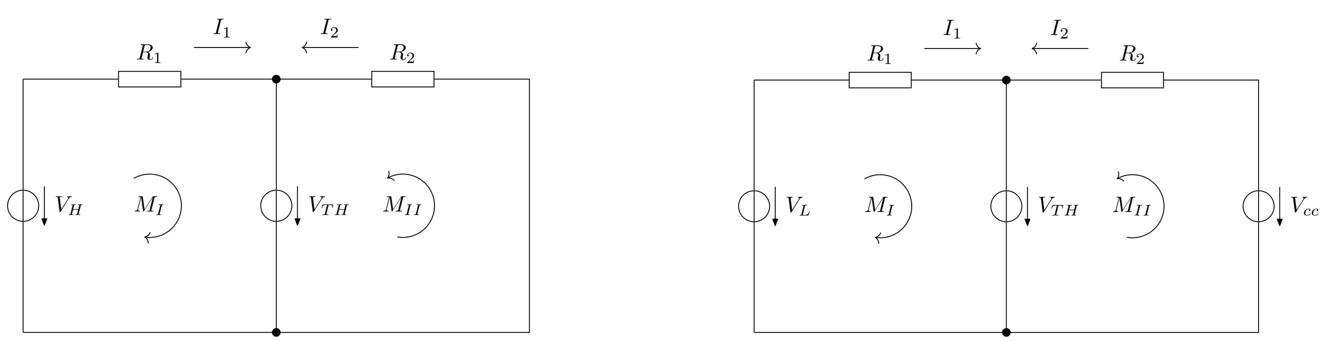


Figure 4.13: Comparator hysteresis

We search for V_H, the voltage where the comparator output will transition from low to high.

V_{TH} is the threshold voltage applied to the inverting input of the comparator. This is also known bias voltage.



(a) KK for low-high transition

(b) KK for high-low transition

Figure 4.14: Equivalent circuits for hysteresis equations

Applying KKL for Fig 4.14a gives us:

$$V_{TH} - V_H + R_1 I_1 = 0 \qquad \text{mesh } M_I$$
$$V_{TH} + R_2 I_2 = 0 \qquad \text{mesh } M_{II}$$
$$I_1 + I_2 = 0$$
$$\Rightarrow V_{TH} = V_H \frac{R_2}{R_1 + R_2}$$
$$\text{let } k = \frac{R_2}{R_1 + R_2}$$
$$\Rightarrow V_{TH} = V_H k \qquad (4.1)$$

Applying KKL for Fig 4.14b yields:

$$V_{TH} - V_L + R_1 I_1 = 0 \qquad \text{mesh } M_I$$
$$V_{TH} - V_{cc} + R_2 I_2 = 0 \qquad \text{mesh } M_{II}$$
$$I_1 + I_2 = 0$$
$$\Rightarrow -V_L + R_1 I_1 + V_{cc} - R_2 I_2 = 0$$
$$\Rightarrow -V_L + V_{cc} - I_2(R_1 + R_2) = 0$$
$$\Rightarrow I_2 = \frac{V_{cc} - V_L}{R_1 + R_2}$$
$$\text{let } k = \frac{R_2}{R_1 + R_2}$$
$$\Rightarrow V_{TH} = V_{cc}(1 - k) + V_L k \qquad (4.2)$$

With eq. 4.1 and 4.2 we must now select values for R_1, R_2 and V_{TH}. In practice, these values are not real numbers but must be chosen from a finite set of available components. In addition, at least three different circuit configurations for setting V_{TH} are possible:

- a basic voltage divider as show in (SBOA313A). The trade-off here is that larger resistors introduce more noise and smaller resistors increase power consumption. Since our application is subjected to large temperature variations, the resistors should not only feature tight tolerances (at least 0.1 %) but also a low temperature coefficient. Such resistors are expensive.
- a voltage divider followed by a low noise op amp in buffer configuration. This choice increases component count and cost.
- a voltage reference or voltage regulator with low temperature drift. This reduces component count (but not necessarily cost) and offers the best accuracy. The trade-off here is that only a small set of reference voltages is available as single component which means that eq. 4.1 and 4.2 can only be approximated.

We chose the last option since it minimizes component count. Selecting voltage regulator U_7 with $V_{TH} = 1.9\,\text{V}$, $R_1 = 129\,\text{k}\Omega$ and $R_2 = 569\,\text{k}\Omega$ approximates the desired tripping voltages $V_l = 3.7\,\text{V}$ and $V_h = 3.9\,\text{V}$ with acceptable accuracy.

This is verified with the help of a circuit simulator.

Simulation the power supervisor circuit

We use LTSpice 17.1.10. We simulate $V_L = 2.89\,\text{V}$ and $V_H = 3.84\,\text{V}$. Recall that the goal was $V_L = 3.0\,\text{V}$ and $V_H = 3.9\,\text{V}$. This deviation is acceptable for our application.

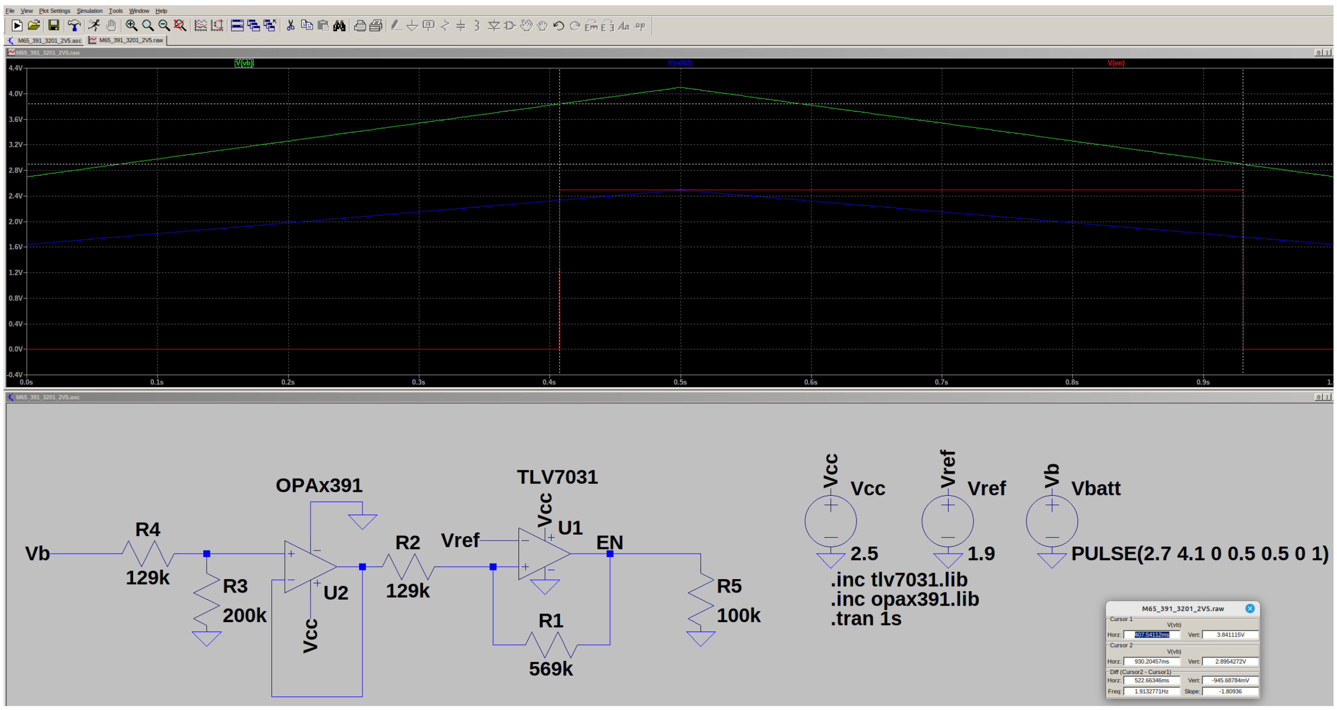


Figure 4.15: Power supervisor Spice simulation

4.6.7 Master Switch module (MS)

The Master Switch module (MS) is identical to 4.4.3 as far as the BOM is concerned.

Implementation

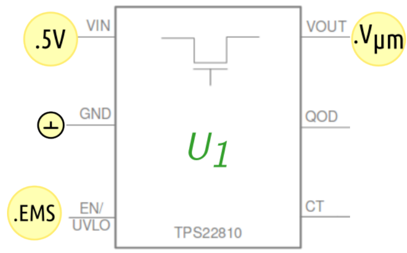


Figure 4.16: MS - schematic, see also 4.4.3

		Pin mapping			
Id	Net	Nb.	Name	Type	Function
U_1	.5V	1	VIN	↼	input
U_1	⊥	2	GND	⊥	
U_1	.EMS	3	EN/UVLO	⇀	switch enable
U_1	.$V_{\mu M}$	6	VOUT	⇀	output

4.7 Optical Sensor Module (OS)

The Optical Sensor Module (OS) transforms the visual activity of the analog water meter display into a voltage. A reflective tally disk is encapsulated in the meter housing and protected by glass. The number of rotations per time unit corresponds to the amount of water drawn downstream.

4.7.1 Requirements

The sensor acts as an optical proximity detector. The distance between the emitter and the reflective surface changes as the tally disc passes underneath the emitter beam. The receiver must be able to discern this change in distance. Furthermore, the sensor should consume as less power as possible while operating from 3.3 V or 5 V supply voltage.

4.7.2 Implementation

The optical switch comprises an infrared light (940 nm) emitting diode with a typical forward current of 50 mA and a phototransistor. In our application, the upper hatched area in Fig.4.17 corresponds to the revolving disc of the water meter totalizer.

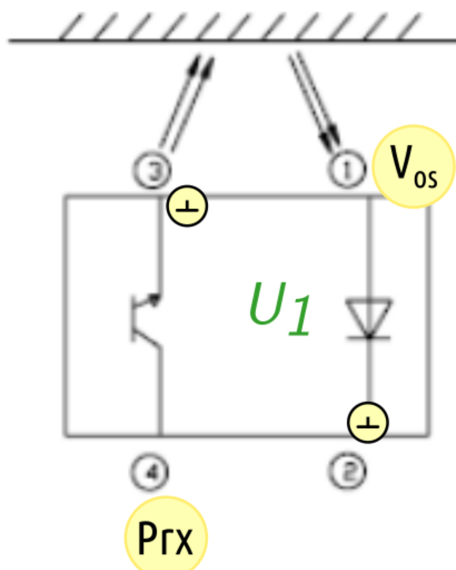


Figure 4.17: OS - schematic

		pin			
Id	Net	Nb.	Name	Type	Function
U_1	V_{OS}	1	1	←	IR diode anode
U_1	⊥	2	2	⊥	IR diode cathode
U_1	⊥	3	3		IR emitter
U_1	⊥	4	4	↼	IR collector

Id	Desc	Order Code	Package	Note
U_1	EVERLIGHT, *ITR9904*	ITR9904/230		

4.7.3 Opto Switch Drift compensation

The open collector output of the phototransistor, net `Prx`, represents the analog reading of the current position of the totalizer tally disc. This value will be "high" when the reflective surface of the revolving disk does not obstruct the infrared beam emitted by the diode in U_8. It will be "low" during the time when the infrared beam is reflected back to the phototransistor. When the flow of water is low, several seconds might pass until the disc has fully traversed the region covered by the sensor. The driver software in the master must account for this situation and filter out duplicate increments. The question is now which collector voltage is "high" and which is considered to be a "low".

We observed that the collector voltage of U_8 is subject to drift. And indeed, the datasheet of the ITR9904 mentions several sources of temperature dependent variables:

- forward current
- peek emission wavelength
- collector power dissipation
- collector dark current
- relative collector current

This drift can be either compensate in hardware or in software. The former is not obvious because the production system can not be easily reproduced in the lab. In addition, we suspect that the high humidity inside the gully contribute to the drift problem.

We therefore decided to implement a software compensation that is detailed in chapter 5.1.1.

CHAPTER 5

SOFTWARE

The software consists of several components:

1. firmware for μM and μS.
2. server (Linux daemon) for data retrieval and storage.
3. command line tools for post-processing and reporting.

5.1 Firmware

The firmware consists of two separate programs that are loaded (flashed) on the master and the slave microcontroller. Both programs share a set of libraries that provide common functions like logging and user I/O. Both subsystems write diagnostic data to a SD-card that can be used for debugging.

5.1.1 Master firmware

The master has the following tasks:

1. after startup, request the current internet time from the slave
2. then start incrementing the flow counter based on the voltage readings from the IR opto switch
3. at a given time - currently at 12pm - transmit the data to the slave for upload to the MQTT broker
4. while waiting for the upload to succeed, continue incrementing the flow counter
5. if the slave acknowledges a successful upload, reset the counter and continue
6. if the slave does not succeed, retry later and continue

Let us further detail these tasks.

At the end of the startup routine the following conditions hold:

- the Watchdog Module (WM) is enabled (.MA.WD.EWD = H).
- the Slave Module ((μS)) is powered of (.MA.SS.ESS = L).

The master then goes into power-saving sleep mode and wakes up every n seconds to perform the following actions:

- toggle MA.WD.¬RS to prevent a watchdog timeout and a subsequent reset.
- read the analog opacity level from module OS with the following sequence:
  .MA.OD.EOD = L, read analog input MA.OD.Prx, .MA.OD.EOD = H.
- read the digital input MA.BI.CON' to check if the slave is ready to receive commands via the I2C bus.

Internet Time

Accurate time is required for logging (timestamp) and to initiate the transmission request at a specific time. The time is updated only at startup, there will be some drift over time based on the stability of the microcontroller RTC. This is acceptable for this application.

Robust sensor reading

The voltage read from the IR opto switch oscillates between a maximum and minimum voltage. To increase robustness with respect to noise, a lower and higher threshold are chosen. Further, a hysteresis function must be applied to avoid erroneous transitions in between the maximum/minimum values. The question is now how to choose the lower and upper thresholds as well as the hysteresis. As explained in chapter 4.7.3 the sensor readings are subject to drift and it is therefore desirable to increase the current through the sensor to decrease the impact of drift by widening the gap between lower and higher threshold. This however, increases power consumption. From the perspective of power consumption, it would therefore be optimal to drive just enough current through the sensor to allow for unambiguous discrimination of upper and lower thresholds.

To mitigate the effect of drift, we use an averaging algorithm that constantly adjusts the upper and lower thresholds:

```
current_min ← high_rail                              ▷ upper rail, Vcc, supply voltage
currentmax ← low_rail                                ▷ lower rail, Gnd, 0 V
lower threshold breached ← false                     ▷ have we already seen the lower threshold ?
loop                                                 ▷ every second
    current ← A_0
    current_min ← min(current, current_min)          ▷ search for new min
    current_max ← max(current, current_max)          ▷ search for new max
    if n (mod N) = 0 then                            ▷ every Nth invocation
        update low                                   ▷ apply hysteresis
        update high                                  ▷ apply hysteresis
        currentmin ← high_rail                       ▷ reset
        currentmax ← low_rail                        ▷ reset
    if (current < low) (∧ ¬ lower threshold breached) then
        inc counter
        lower threshold breached ← true
    else if current > high then
        lower threshold breached ← false
```

Figure 5.1: Adaptive threshold algorithm

Power sensitive operation

Upload only if battery voltage is high enough.

Master Slave communication

The I2C bus is used for bidirectional communication with the help of the Arduino `Wire` library:

1. the master, during startup initializes the bus with a call to `Wire.begin()`.
2. the slave, during startup joins the bus with a free I2C address `Wire.begin(address)`.
3. the slave installs event handlers to receive data and to receive requests to send data.
4. the slave interconnects `SDA` and `SCL` between master and slave by pulling up D_0.

5. the master - and only the master - initiates the communication by either sending data with `Wire.write(data)` or requesting data with `Wire.requestFrom(address,nb of bytes)`.

5.1.2 Slave firmware

At specific points in time, the master transmits the meter reading to the slave for upload to a MQTT broker. Currently, the time interval is fixed (every 24 hours) but a more sophisticated strategy would be to make this interval dependent on the current charge level of the battery. That is in summer - where leaks are more disturbing and energy is plentiful - more frequent transmissions could be scheduled than in winter where the conditions and requirements are very different.

In our remote environment with relatively pour radio coverage we have found it challenging to upload data via General packet radio service (GPRS). Firstly, connection attempts frequently fail and no data is transmitted at all. Secondly, the Arduino `MQTT` library might return one of the following error codes despite the fact that the upload has been successful:

(This was with Qos = 1)

```
LWMQTT_NETWORK_FAILED_CONNECT = -3,
LWMQTT_NETWORK_TIMEOUT = -4,
LWMQTT_NETWORK_FAILED_READ = -5,
LWMQTT_MISSING_OR_WRONG_PACKET = -9,
```

And lastly, based on the return value provided by `MQTT`, the slave might think that the transmission was successful when in fact it was not.

5.1.3 Choosing MQTT parameters

For our purposes, the following parameters are important:

1. Qos (Quality of Service): we use `Qos = 0` because as explained before the handshake operation does not seem to work reliably, given the intermittence of the radio link.
2. Retained messages: we use `retained = true` to garantee that the server can connect and subscribe without blocking
3. Persistent Session/Queued Messages: we use `cleanSession = true` because we do not need to store subscription information or any other information across multiple TCP connections.

The most important MQTT feature for our application is Retained Messages. This feature guarantees that when the server connects, it will always see the last data uploaded by the slave. Retained Messages provide a start state. Note the difference between Retained Messages and Queued Messages.

Retained Messages

- work on a topic level
- newly-connected subscriber to a topic receive a message immediately

Queued Messages

- work in a client context
- the broker queues undelivered messages for a specific client

We do not need to queue messages, because we know exactly how many messages the slave will send in a given timeframe. Therefore, the sample interval of the server can be chosen such that messages are never lost.

To make the upload procedure reliable we chose the following approach:

```
success = false
for all n ∈ {1, . . . , 6} ∧ success do
    open GPRS connection
    open subscribe to topic water_meter                    ▷ subscribe before publish !
    publish meter data to topic water_meter
    received ← echo from subscription            ▷ this only happens when the upload was successful
    success = (sent == received)                      ▷ we are finished when the echo matches
if success then
    master ← success
else
    master ← failure
```

Figure 5.2: Slave Broker handshake operation

Once the master receives either `success` or `failure`, the slave is powered off.

A slight disadvantage of this handshake operation is that the message send from the broker to the slave gets lost. In this case, the slave will attempt another (paid) transmission despite the fact that the previous upload has been successful. Our strategy is therefore conservative which matches the requirements.

5.2 Server software

This component is available as a Linux daemon managed with the `systemctl` facility which is standard on current Linux desktop and server stations. The software is meant to be installed on an always-on box and should be configured via `systemctl` to start automatically on boot. The software is written in Java.

The daemon has the following tasks:

1. poll the MQTT broker for a new upload from the slave
2. compensate the reading for possible gaps in the upload sequence
3. store the data in a local database
4. upload the data to a cloud-based database such that data can be visualized in a static webpage
5. alert the supervisor in case of abnormal readings (leak)

5.2.1 Getting data from the broker

When using a publicly accessible broker, one must bear in mind that this doesn't come for free. While there are free solutions, we found them unsuitable for a production scenario. Since MQTT is a publish-subscribe framework, the obvious way to get data updates is to subscribe to the relevant topic. However, the broker has to do work to maintain the subscription connection and this translates in transactions that are billed.

We therefore do not entertain a permanent subscription with the broker in the same way as we would choose if that broker were deployed locally where we would not incur communication costs. Rather, we connect and subscribe only at specific time points. Since the slave sends the message with the retained flag set, the server is guaranteed to get an update instantaneously (after at least one upload has been performed by the slave). In other words, the `Runnable` passed to the `ScheduledService` will never block. This approach resembles more to polling than to publish/subscribe. As long as the polling interval is shorter than the upload interval, we are guaranteed not to lose any data. Here, we connect every 12 hours. Recall that the slave uploads every 24 hours.

Here is an extract from the server code:

```
1 Runnable r = () -> {
2                         var cloud = io.smq().get();
3                         Effect<Data_Arduino> e_inc = data -> {
4                                 cloud.disconnect();
5                                 pf.put(f_yi_sensor(io::y).apply(data));
6                         };
7                         cloud.sub(arduino_mqtt_topic, fd_broker, e_inc);
8                 };
9         ScheduledFuture<?> update_data = io.fses().apply(r);
```

Figure 5.3: Polling-style publish/subscribe operation

Comments :

1: create a `Runnable`. The code in curly brackets will not execute until passed to the scheduler in line 9.

2: open the connection with the MQTT broker. A `Supplier` is used so that we can pass in a test mock-server.

3-6: event handler that will run when the server receives a message on topic `arduino_mqtt_topic`. Note that we disconnect from the broker as soon as we receive the message.

7: subscribe to the broker and receive a message immediately (without blocking) because of the Retained Message mechanism explained earlier.

9: submit the `Runnable` created in 1. to the scheduler service (again obtained in a way that simplifies tests). The code will run at predefined intervals.

CHAPTER 6 CONCLUSIONS

We have presented a simple system to continuously monitor the volume of drinking water passing through an analog water meter upstream of a village located in a rural environment. The embedded part of the system is powered by a small off-grid photovoltaic installation and has to face intermittent and poor GSM signal coverage. We have designed the system as a set of encapsulated hardware modules, inspired by common software engineering techniques. We plan to improve many of these modules based on their performance during production. The data we accumulate during operation (battery voltage, solar production, optical sensor output voltage range) will be a valuable help.

We hope to encourage others to experiment with simple environmental sensors like the one discussed in this document.

BIBLIOGRAPHY

SN74LVC2G17 DualSchmitt-Trigger Buffer. 473. 2002. URL: https://www.ti.com/product/SN74LVC2G17?dcmp=dsproject&hqs=pf.

TS5A23157 Dual10- Ω SPDTAnalog Switch. 176. 2004. URL: https://www.ti.com/product/TS5A23157?dcmp=dsproject&hqs=pf.

CDBA340L-HF Low VF SMD Schottky Barrier Rectifiers. 618. 2009.

EVERLIGHT. *ITR9904 OPTO INTERRUPTER ITR*. 230. 2010.

TPS783xx 500-nA IQ,150-mA, Ultralow Quiescent Current Low-Dropout Linear Regulator. 922. 2014. URL: https://www.ti.com/product/TPS783?dcmp=dsproject&hqs=pf.

Arduino MKR Zero. 2016. URL: https://docs.arduino.cc/hardware/mkr-zero.

TPS22810, 2.7-18-V, 79-mΩ On-Resistance LoadSwitch WithThermal Protection. 710. Dec. 2016. URL: https://www.ti.com/product/TPS22810?dcmp=dsproject&hqs=pf.

TPS3890 LowQuiescent Current, 1%Accurate Supervisor withProgrammable Delay. 235. 2016.

Type RN73 Series. 59. 2016.

AP7115, 150mA LOW DROPOUT LINEAR REGULATOR WITH SHUTDOW. 2017. URL: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/911/AP7115_Sept2017_DS.pdf (visited on 11/02/2023).

TPS3431 Standard Programmable Watchdog Timer with Enable. 681. 2018. URL: https://www.ti.com/product/TPS3431?dcmp=dsproject&hqs=#tech-docs.

CDSOD323-TxxSC – TVS Diode Series. 820,945. 2019. URL: https://www.mouser.ch/datasheet/2/54/cdsod323_txxsc-777083.pdf.

ARDUINO MKR GSM 1400. 2020.

0.96 INCH OLED SCREEN WITH I2C FOR ARDUINO. 59. 2021.

TLV703x and TLV704x Small-Size, Nanopower, Low-Voltage Comparators. 923. 2021. URL: https://www.ti.com/product/TLV7031?dcmp=dsproject&hqs=#tech-docs.

RN73H long term precision thin (metal) film flat chip resistors (high reliability, for automotive). 52,55. 2022.

TI. *OPAx391 Precision, Ultra-Low IQ, Low Offset Voltage, e-trim™ Operational Amplifiers*. 931. 2022. URL: https://www.ti.com/product/OPA391?dcmp=dsproject&hqs=#tech-docs.

— *REF35 Ultra Low-Power, High-Precision Voltage Reference*. 921. 2022. URL: https://www.ti.com/product/REF35.