https://doi.org/10.2352/ISSN.2470-1173.2021.3.MOBMU-038

IoT-Based Real-Time Monitoring System for a Smart Energy House

Lukasz Rojek[12], Saiful Islam[1], Michael Hartmann[1], Reiner Creutzburg[13]

[1] SRH Berlin University of Applied Sciences, Berlin School of Technology, Ernst-Reuter-Platz 10, D-10587 Berlin, Germany

[2] Beuth University of Applied Sciences, Berlin, Luxemburger Straße 10, D-13353 Berlin, Germany

[3] Technische Hochschule Brandenburg, Department of Informatics and Media, IT- and Media Forensics Lab, Magdeburger Str. 50, D-14770 Brandenburg, Germany

E-mails: lukasz.rojek@srh.de; saiful.islam@srh.de; michael.hartmann@srh.de; reiner.creutzburg@srh.de

Abstract

Monitoring Systems and the Internet of Things have become increasingly important, among others, in Renewable Energy applications. A combination of measuring sensors and actuators controlled and processed by an intelligent central system is necessary to reduce energy consumption automatically.

This paper develops a modern concept for a Smart Energy House. The functionality and the hardware implementation are explained in detail based on a concrete simulation of the self-refilling water tank. The system comprises various separate located IoT modules integrated with a central host using TCP/IP network infrastructure, as well as communication technologies and protocols, such as WiFi and MQTT. Those sensors, also called clients, are wireless devices designed to measure different environmental conditions, such as temperature and humidity, in real-time. Changes in the current water level and battery charging progress are also being monitored. The host consists of three parts: storage service, web-based monitoring platform, and program logic for decision-making methods. Most of the processes, such as electrical control, data collection, information query, and analyzing functions, were implemented using Python libraries and self-written algorithms.

An essential part of an intelligent home monitoring and automation system is secured remote accessibility and maintenance in any emergency. Therefore, the system supports both local and remote access. Data visualization and alarming routines are implemented within the web-based free software "Grafana" combined with the time-series database "InfluxDB".

Finally, the results demonstrated in this article show that the system has excellent application prospects due to its stability, low cost, high performance, user-friendly and customary configuration. The concept has been improved and prepared for the Smart Energy House test object located in Berlin.

Introduction

Renewable Energy (RE) is nowadays a significant life aspect, especially in Germany. According to the Federal Ministry for Economic Affairs and Energy (ger. Bundesministerium für Energie und Wirtschaft, BMWi), by 2025, 40-45% of electricity consumed in Germany is to derive from renewables. The share of renewable energies in electricity consumption is growing steadily from 6% in 2000 to around 38% in 2018. The target of 35% for 2020 came to prior [1]. Optimization of maintenance costs through modern technologies is becoming increasingly important.

In modern households, so-called Smart Energy Houses (SEH), integrated intelligent monitoring systems are getting more and more popular. They not only guarantee a high standard of safety but also offer enormous potential for reducing energy consumption. For Instance, continuous tracking of the electrical parameters such as voltage and current ensures that devices, which are plugged in but not currently in use do not run unnecessarily in standby mode to avoid flow of electricity [2].

Within this study project, the authors combine the potential of the Internet of Things (IoT) with Renewable Energy technologies to create home automation and monitoring system for reducing energy usage and, consequently, the costs. The term Internet of Things is becoming the trend of network convergence [3]. It can be defined as a dynamic global infrastructure of networked physical objects augmented with sensing, processing, and communication capabilities [4]. In this vision, various uniquely identified devices, such as smart-phones, microcontrollers, sensors, and actuators, are intelligently linked together to interact with each other and exchange information with people [5]. In this way, users can get certain services with any communication devices and any communications methods at any time [3].

A real-time monitoring system collects and processes data periodically at zero or low latency to immediately evaluate and react to unwanted events. The instantaneous data update is significant in situations where the reaction time is a very critical aspect. Moreover, the measured data are being logged over time for further post-processing analysis to identify trends and train the system for predicting the environment's behavior.

The complete process is divided into five steps, as shown in Figure 1, and explained in this paper. Firstly, environmental data such as temperature, humidity, water, and battery level are collected using low-cost sensors and stored in the database. The next step is to analyze the measurement data for making decisions and applying changes in the system environment. Furthermore, Artificial Intelligence (AI) has proven to be a suitable method for predicting energy consumption and costs in RE [6]. Based on the forecasting, it is possible to optimize the cost with a better prognostication of maximum power production from photovoltaic (PV). A central unit (server) automatically controls electrical devices depending on the system configuration in order to reduce energy consumption. The current status of all components is updated in the real-time monitoring and warning software. Some features can be controlled

by remote access for more flexibility. The system is designed in a modular way, easy to implement and upgrade with additional sensors if necessary. This project aims to create a zero-emission smart house that responds to the needs of the user.

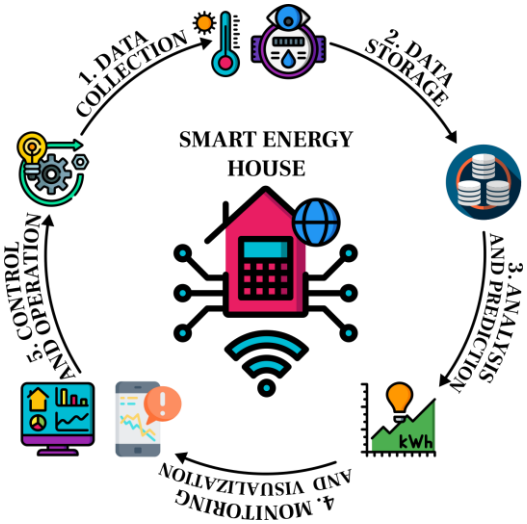


Figure 1. Process diagram of the Smart Energy House Monitoring System.

Although the project consists of two significant parts, system automation based on the Internet of Things and cost prediction using Artificial Intelligence, this article focuses only on the first part about the technical implementation and is structured as follows: Section 2 describes the model design and system's approach according to the self-refilling water tank simulation. Section 3 explains the hardware components, information flow, and the functionality of each integrated sensor. The network architecture and communication protocols are presented in Section 4. Additionally, data visualization and warning features are explained in Section 5. The paper is concluded with a short outlook and a summary of the system's potential and limitations.

Regarding the second part on the financial background, cost calculation, and system feasibility, the authors would like to refer to Saiful et al. (2020).

Simulation Model Design

In Figure 2, a self-refilling water tank simulation demonstrates the implementation of the Smart Energy House and the functionality of the monitoring system. The PV panel charges the battery. As soon as the battery is full, the connection to the PV panel will be interrupted. The energy from the accumulator is used for running different electrical devices. In this particular example, the water tank is simulating the resources used by the consumer. With the help of basic low-cost sensors, water consumption can be adjusted to the consumer's needs. Two independently running devices check the current water level continuously. Every time the water tank is getting empty, the water pump energized by the battery will refill the tank until the water reaches the maximum limit. Additionally, two temperature and humidity sensors located on the water tank and the PV panel control the environmental conditions essential in the analyzing process.

All components communicate permanently with the central unit, Raspberry Pi (RPi), through the lightweight Message Query Telemetry Transport (MQTT) protocol. The data transfer works over the local network. The system can be monitored and controlled from anywhere. Thus, the router was used to enable communication between the local network and the university network. The whole process can be observed in real-time from any end-device.

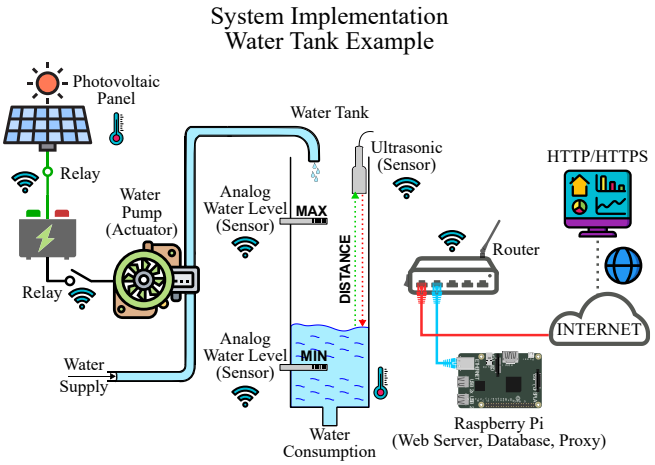


Figure 2. System approach of the self-refiling water tank simulation.

Hardware Components

The control system uses various electronic devices that are related to each other in a certain way. Figure 3 shows a schematic diagram of the information flow between the hardware components. Those are classified into the group of sensors and actuators. The information is passed through the central unit that not only integrates all devices but also runs control and analysis processes. A sensor converts a physical quantity such as temperature and humidity to an analog electrical signal output as a voltage or a current or a change in resistance [7]. The analog signal, often amplified, filtered, or otherwise conditioned, is being passed on to an analog-to-digital converter (ADC) [7]. There it is converted into a digital representation sent to the control unit. Once the program logic analyzes the measured data, the controller provides an output signal to the actuators in order to apply changes in the system. In this way, the relay (actuator) will stop the water pump based on the measured water level by the ultrasonic sensor.

Many sensors communicate over standard interfaces and protocols. Others involve additional work for the one-wire data transmission, which requires very precise and exact timing.

The user can modify and adjust the decision-making processes by changing the parameters according to new circumstances.

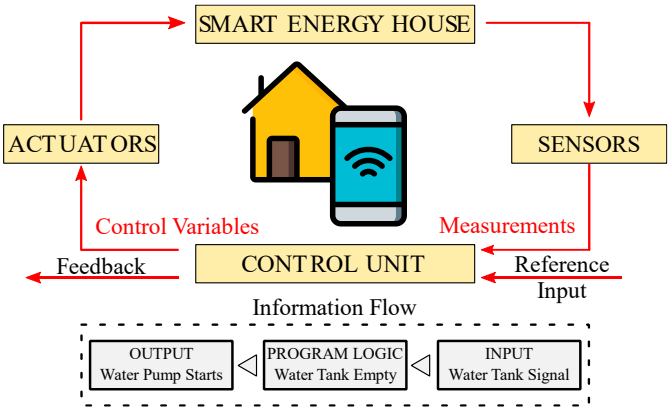


Figure 3. Schematic diagram of the information flow between the hardware components.

Single Board Computer

In principle, a Single Board Computer (SBC) is a device that contains all the components, such as the microprocessor, memory, power management, physical input and output connectors, and other features required to operate the system [8]. All those components are integrated and soldered onto a single circuit board, providing a complete hardware and software integrated solution [8]. The most well-known SBC is the Raspberry Pi (RPi), which appeared in 2012 [9]. Since then, RPi went through several hardware versions. The newest model showed in Figure 4 (a) is Raspberry Pi 4 B+. This product offers up to 8 GB LDDR4 and a 64-bit Quat-Core 1.5 GHz Cortex-A72 ARM processor [10]. Due to the advanced functionality and performance in a credit-card-size and the low price, RPi is well suited, especially for IoT applications. The standardized interfaces enable direct communication with the system. The built-in Ethernet and dual-band wireless LAN modules allow the board to be integrated into a network and designed for remote control and maintenance. The General Purpose Input/Output (GPIO) interface enables various sensors from different types to be wired to the board. Officially, Raspberry Pi uses Raspbian Linux operating system, which is based on Debian. Other alternative distributions are available and compatible with this device as well. For less intensive applications, such as data acquisition and forwarding to the server, the full resources of RPi 4 B+ are not required and can be replaced by the lightweight, smaller, and cheaper 32-bit version of Raspberry Pi Zero W (Fig. 4 (b)).

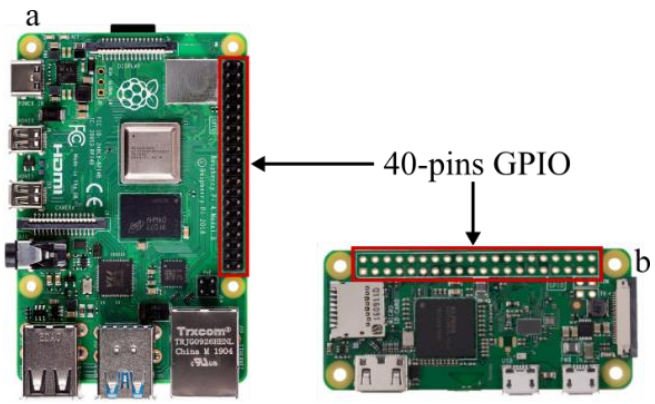


Figure 4. (a) Raspberry Pi 4 B+; (b) Raspberry Pi Zero W.

It does not have a built-in Ethernet interface but can still be integrated into the network through the Wi-Fi module. Furthermore, wireless connection gives more flexibility, as the sensors can be replaced without problems, as long as they are still within the signal range.

In this project's scope, one Raspberry Pi 4 B+ was used as the primary host (server) for providing all the services required for data transmission, backup, analysis, and visualization. Most of the control processes are written in Python. Multiple RPi Zero W were used mainly for collecting sensor data. The monitoring system was designed in a modular way. It means that additional components, such as water level sensors, can be added without extra effort from the user. An appropriate system image with client software is required. The new device will be automatically detected and set up by the monitoring system as soon as it is connected and reachable over the network. Both microcontrollers use an SD-card as primary storage simplifying the backup process. It is only a matter of one system image file from the SD-card that can be restored at any time.

Temperature and Humidity

The DHT11 in Figure 5 (a) is an ultra-low-cost temperature and capacitive humidity sensor with an integrated analog to digital converter of 8-bit resolution [11, 12, 13]. It can detect relative humidity between 20 and 90 % RH within the temperature compensation of 0 to 50°C with an accuracy of ± 5 %. Temperature is measured in the range of 0 to 50°C with an accuracy of ± 2 °C [7, 9]. According to the general circuit diagram in Figure 5 (b), the connection between the sensor and the microcontroller (Raspberry Pi Zero W) requires three pins. Pin 1 provides the + 3.3 V power supply, while pin 4 goes to the common ground. Pin 2 is responsible for the two-way data transmission and can be connected to a chosen GPIO pin defined in the program. Additionally, a 5 kΩ pull-up resistor is recommended [7, 9]. The resistor's task is to pull the line up to a high level of + 3.3 V for eliminating the floating signal during the listening process when the DHT11 is not sending data.

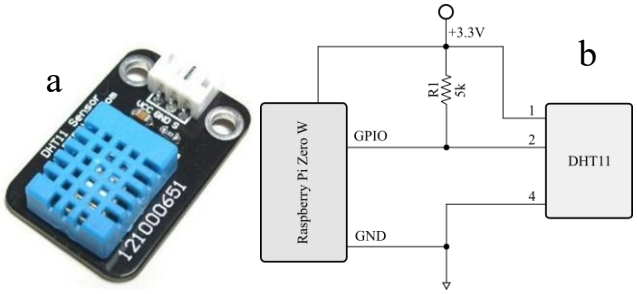


Figure 5. (a) DHT11 temperature and humidity sensor [7]; (b) general circuit connections between the Raspberry Pi Zero W and the DHT11 sensor [9].

A microcontroller cannot use a single pin as *OUTPUT* and *INPUT* at the same time. That is why the single-wire bus requires additional complexity in exact timing for the data exchange protocol. Figure 6 illustrates the overall communication process. DHT11 does not provide measurements automatically. It stays in listening mode until the MCU (RPi Zero W) sends a request for new data. The signal is *HIGH* by default because of the pull-up resistor, indicating the idle state. Initially, the microcontroller configures pin 2 as *OUTPUT* and pulls it to *LOW* for at least 18 ms before releasing the control to the sensor by reconfiguring the pin to *INPUT* [7]. Consequently, the pin is *HIGH* again. This step indicates the interest of the microcontroller for new measurement data. It takes the sensor about 20 to 40 µs for responding first with a *LOW* and then with a *HIGH* output for 50 µs each. This pattern informs the microcontroller about incoming sensor data in the next step. The sensor continues with a 40-bit dataset containing information about the temperature and humidity transmitted to the pin. The dataset is sent bit by bit, beginning with the most significant bit (MSB) based on the following convention. Each bit starts with a 50 µs *LOW* signal followed by a *HIGH* pulse of 28 µs to indicate a 0-bit (logical false) or 70 µs to indicate a 1-bit (logical true). According to the technical documentation, the measured humidity and the temperature are represented by 16-bit values each. However, the second half of 8 bits equals 0 because of the limited sensor resolution. The data set ends with an 8-bit checksum. The checksum is a binary sum of all 4 bytes and allows verification of the dataset's completeness. Finally, the transmission ends when the signal goes one more time for 50 µs *LOW* and returns to a *HIGH* idle state.

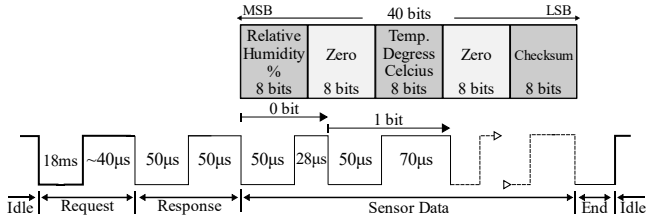


Figure 6. Schematic diagram of the single-wire communication protocol. The thin line represents the signal controlled by the sensor, while the thick line represents the signal controlled by the MCU.

Data collection is done automatically using a script written in Python. The implementation of already existing open-source libraries, such as *adafruit*, minimizes the programming effort. Due to the time-sensitive communication process, the data query frequency cannot be higher than 1 Hz. Nevertheless, the sensor characterizes an excellent price-quality ratio and better accuracy compared to some of the infrared thermometers [11]. In this study, several DHT11 sensors were used to determine not only the water temperature but also the photovoltaic panel's heat level. Furthermore, one more device was combined with the ultrasonic water level sensor for estimating the sound speed correction.

Water Level Detectors

In this project, the authors decided to use two completely different measuring methods for automatic water level detection to increase the system's accuracy and redundancy. Both techniques were implemented at the same water tank and tested simultaneously. The next two chapters explain the technical background, functionality, as well as their advantages and disadvantages.

Analog Water Level Sensor

The first measuring method showed in Figure 7 implements two analog water sensors same type placed in the upper and lower part of the tank, signalizing if the tank is full or empty.

The analog water level sensor consists of three pins identified as VCC for the power supply, GND for the common ground, and SIG for the outgoing signal to the microcontroller. Because it is an analog sensor without a built-in digital converter, the output signal is analog as well. By technical documentation, an input voltage between + 3.3 V and + 5 V is recommended. Depending on the voltage level provided to the sensor, the analog output voltage will vary by the same water coverage.

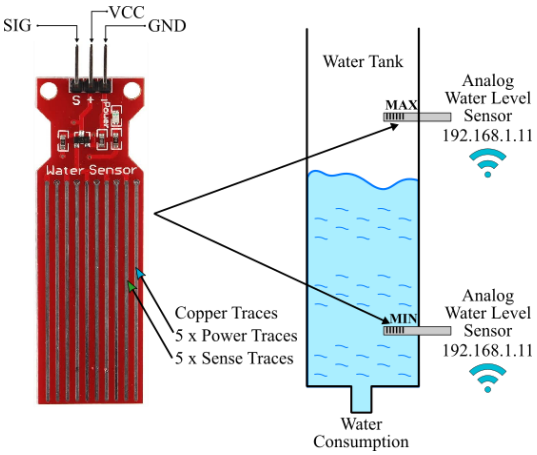


Figure 7. Analog water level sensor placed in the upper and lower part of the water tank.

The sensor has a series of parallel exposed copper stripes across the board divided into power traces and sense traces ordered alternately (Fig. 7). However, those parallel traces are not connected so that they can be bridged by water. In principle, conductors act together as a resistor, whose resistance is changed by increasing the conductivity between the stripes according to the water level [7]. The functionality of the sensor is straightforward and presented in Figure 8. The more liquid is covering the sensor board, the higher the conductivity between the stripes. Consequently, the lower is the resistance, and the measurement results in a higher output voltage, indicating water level raised. Similarly, the higher resistance caused by a lower water level will result in lower output voltage.

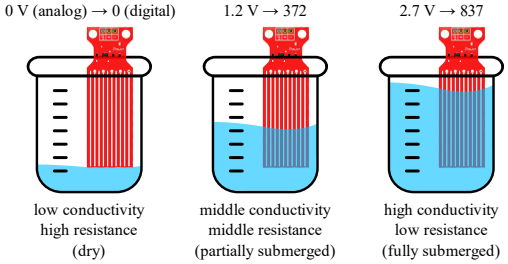


Figure 8. Functionality of the analog water level sensor.

The sensor uses a single pin for unidirectional communication with the microcontroller. Since the GPIO of Raspberry Pi can only process digital signals, an analog to digital converter (ADC) is required [6]. The MCP3008 in Figure 9 is an 8-Channel 10-bit ADC. The microchip contains 16 pins in total. The first eight pins (CH0 – CH7) measure up to eight different analog input voltages with a resolution of 10-bit. It means that the highest possible digital output value is 1023 (2^{10}-1). Unlike the water sensor, the ADC requires the reference voltage (VREF) to calculate the unknown input voltage in addition to the operational power (VDD). Respectively to this, the chip needs two ground connections (GND and AGND). The remaining four lines are dedicated for the data transmission with the microcontroller. The output digital value can be estimated using following formula:

$$\text{Digital Value} = \frac{V_{AM} \cdot \text{ADC Resolution}}{V_S} \quad (1)$$

where V_{AM} stands for analog measured voltage and V_S represents system voltage. For Instance, an analog voltage of 1.2 V will result in 372 digital integer number by system voltage of 3.3 V.

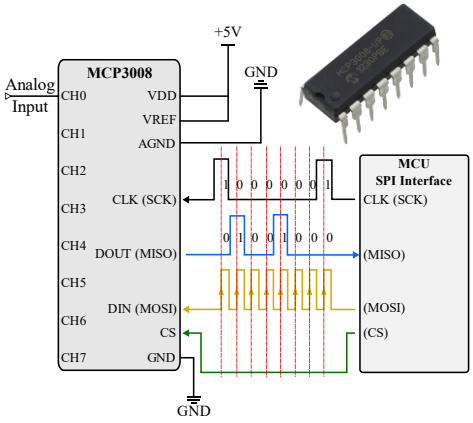


Figure 9. Circuit connection between MCP3008 8-Channel 10-bit analog to digital converter and the microcontroller (RPi Zero W) over the SPI-Interface.

MCP3008 communicates over a Serial Peripheral Interface (SPI), a popular industry-standard protocol next to Inter-Integrated Circuit (I2C) and RS232. The principle of SPI wiring is presented in Figure 9. It is a bus system for synchronized and serial data transmission between a master (server) and one or multiple slaves (client). The communication works bidirectionally means the data flows in both directions simultaneously. The microcontroller generates a clock signal (SCK) to synchronize all connected devices. The client is only active when addressed by the master. The slave device is being selected by a separate control signal (CS). The additional two pins are used for the data transmission from master to slave (Master Out Slave In – MOSI) and slave to master (Slave Out Master In – MISO).

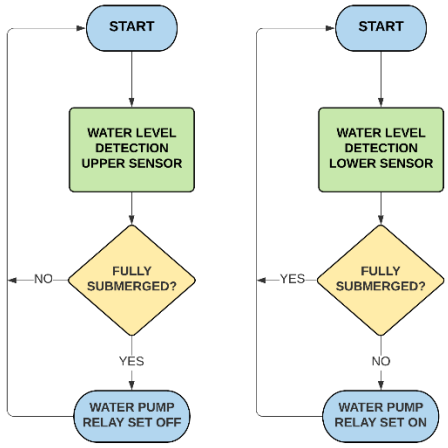


Figure 10. Flow Diagram for decision-making of automatic water pump (method 1).

In this study, one Raspberry Pi Zero W is used to handle both sensors. With this type of device, it is not exactly possible to determine the tank's current water status, but only the last phase when the water level is approaching the minimum or maximum limit. Both sensors operate as on/off switches (Fig. 10). All values above or below the predefined threshold will be interpreted as logical True, the rest as logical False [6]. The first sensor at the bottom of the tank controls whenever the water pump must start, while the second sensor at the top of the tank stops the motor. It is recommended to calibrate the sensor for the type of measured water to get the most accurate results. Because pure water is not conductive, the sensor may be more or less sensitive depending on the amount of the minerals and impurities inside the measured waster.

Water Level Detector (Ultrasonic)

The second technique for detecting water level inside the tank is based on a permanent distance measurement from the top of that tank to the remaining water surface. Whenever the estimated distance reaches the defined minimum or maximum limit, the water pump will be automatically turned on or off to guarantee the optimal water amount. One device that fits this task's requirements is the HC-SR04 ultrasonic sensor, shown in Figure 11. The sensor emits ultrasound at ~ 40 kHz and operates basically like sonar [7]. It uses two components build-in in the same device for transmitting and receiving the ultrasonic waves. The distance is based on the measured travel-time and sound speed, which equals approximately 340 m/s. Except for two standard pins for power supply (VCC) and common ground (GND), the distance sensor comes with two more pins for triggering the sound burst (TRIG) and signalizing the measured result (ECHO). Since the sensor operates mainly with + 5 V and the maximum input voltage for the GPIO pins of Raspberry Pi is + 3.3 V, an additional resistor of 10 kΩ is required.

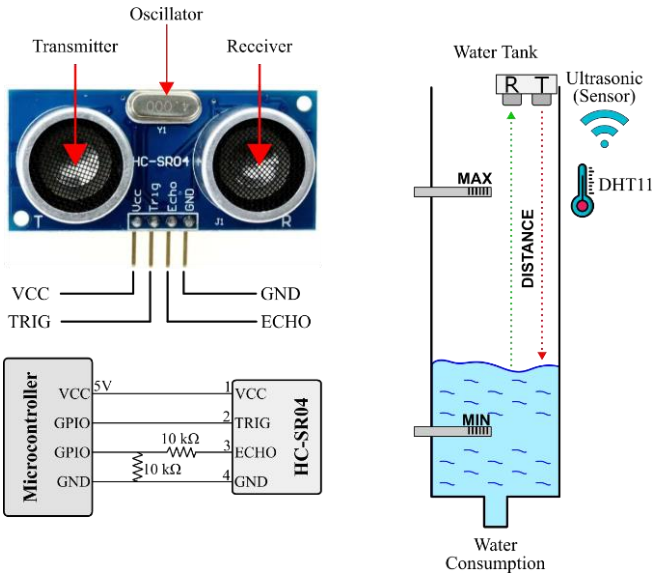


Figure 11. Wiring and functionality of the HC-SR04 ultrasonic sensor.

The measuring process illustrated in Figure 12 starts with a pulse of at least 10 µs applied to the trigger pin. In the next step, the ultrasonic sensor transmits a sonic burst of 8-pulses. This pattern allows the receiver to differentiate the transmitted waves from the ambient noises. The echo pin for the output signal is *LOW* by default. The sensor sets the echo pin automatically *HIGH* after the last sonic pulse was emitted and returns it to *LOW* once the echo arrives or some specified timeout expires. If the sonic waves are not reflected, the echo signal will timeout after 38 ms. This result indicates that no object within the range of the sensor was detected. If the transmitted pulses are reflected, the echo pin goes *LOW* as soon as the signal is received. This time difference for the transition from *HIGH* to *LOW* signal on the echo pin represents travel-time. The following formula gives the final distance L:

$$L = \frac{\Delta t}{2} \cdot v \qquad (2)$$

where Δt is the travel-time and υ is the sonic speed. The travel-time must be divided by two, due to the go-and-return process. For Instance, an output of 250 µs results in 4.25 cm.

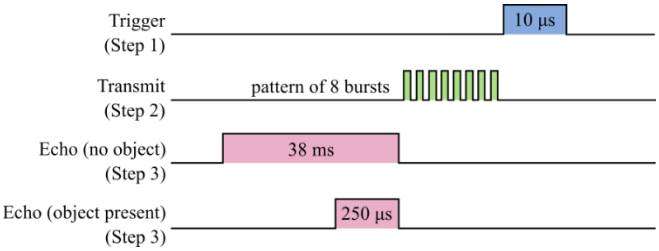


Figure 12. Measuring process of the HC-SR04 ultrasonic sensor.

Since the speed of sound varies depending on ambient temperature and relative humidity, the correct velocity for the ultrasonic waves needs to calculate according to the formula [14]:

$$v = 331.4\ \frac{m}{s} + \left(0.606\ \frac{m/s}{C} \cdot T\right) + (0.0124 \frac{m/s}{\%} \cdot RH) \quad (3)$$

where T ($°C$) is the temperature of the air and RH (%) is the relative humidity measured by the DHT11 sensor.

However, the sensor does have some limitations that need to be considered before applying it into the system. As such, the distance between the device and the target should be within the sensor range. Furthermore, the angle should not be higher than 15 degrees so that the waves can still be reflected. The type of the object's material is also very significant. Since ultrasound can partially pass through the water, it is advisable to use a material, e. g. styrofoam, which floats on water and is large enough so it can reflect the pulses.

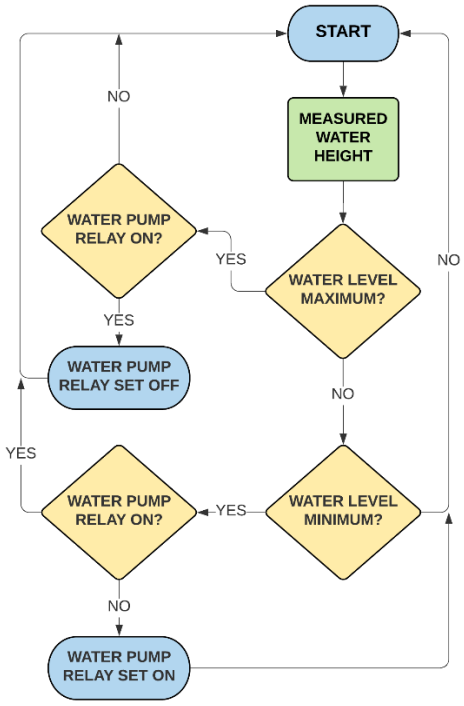


Figure 13. Flow Diagram for decision-making of automatic water pump (method 2).

According to the tank's measured height, the maximum water level has been defined and included in the programming part. The pump will not operate when the water reaches the predefined limit (Fig. 13). Unlike the first method, only one device in the upper part of the tank is required. Furthermore, real-time monitoring of the refilling process instead of the standard empty/full status is possible.

Relay

A relay consists of a small electromagnet (coil) that mechanically closes or opens a contact [7,15]. A single two-channel relay controls the connectivity between the PV-array and battery and the battery and water pump. Figure 14 presents the system implementation. The four pins on the bottom are wired to the Raspberry Pi's GPIO interface, while the top pins are connected to the device pairs. As soon as the output signal from RPi is *HIGH*, the coil is being energized and the gate will close (Fig. 14 (b)). In this way, RPi can control whenever water pump is turned on or off and the battery's charging process.

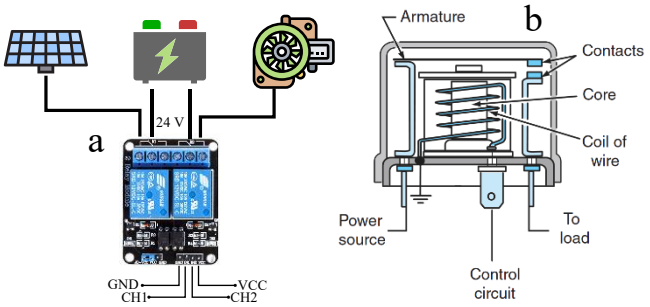


Figure 14. (a) Relay control; (b) Relay components [16].

Photovoltaic Panel

Renewable Energy sources are being used to overcome the grid instability situation, an electrical fault in the distribution system. The Maximum Power Point Tracking (MPPT) is imposed to provide an electrical balance between the incoming voltage from the photovoltaic array and the battery. The power equation (Eq. 4) expresses the relationship between the maximum power point voltage (V_{MPP}) and the system voltage (V_{sys}). The Maximum Power Point, abbreviated MPP, is the point of a solar cell that delivers the highest energy. By high differences between V_{MPP} and V_{sys}, the voltage will be decreased according to the maximum power point tracking inverter, so it corresponds to the required charging voltage. It is essential to ensure that the V_{MPP} voltage is enough to charge the battery and avoid system failures. Both parameters need to be continuously monitored.

$$P = V_{MPP} \cdot I_{MPP} = V_{sys} \cdot I_{charge} \quad (4)$$

Equation 5 is used to estimate the total electrical energy contribution [17]:

$$E_{served} = AC_{prim} + DC_{prim} + E_{served,def} + E_{grid,sales} \quad (5)$$

where AC_{prim} is the essential load served to the system. The DC_{prim} represents the critical load served. The total electrical load served E_{served} is a combination of both (DC_{prim} *and* AC_{prim}) including $E_{grid,sales}$ and the deferrable load $E_{served,def}$ if applicable. In case of any surplus energy, it will be lead-back to the utility-grid.

The challenge in the upcoming days will be how this sector can process the existing electrical supply such as utility grid or fossil fuel-based energy source (e.g., diesel generator) and its economy. Prediction of economic factors and its forecasting in terms of energy supply are useful tools to increase future cost reduction. For this purpose, the financial analysis is being calculated to determine the system's potential value.

Figure 15 shows the photovoltaic power (KW) economic analysis output, solar irradiance (KW/m^2), and the total electrical load served according to Eq. 5. The resulting energy savings are 728 KWh/year, total CO_2 savings are 60.8 kg/year, and the entire production from the renewable sources is around 95.5%. The total renewable fraction is defined by the ratio between the energy supply from nonrenewable sources (grid) and full electrical served load [17].

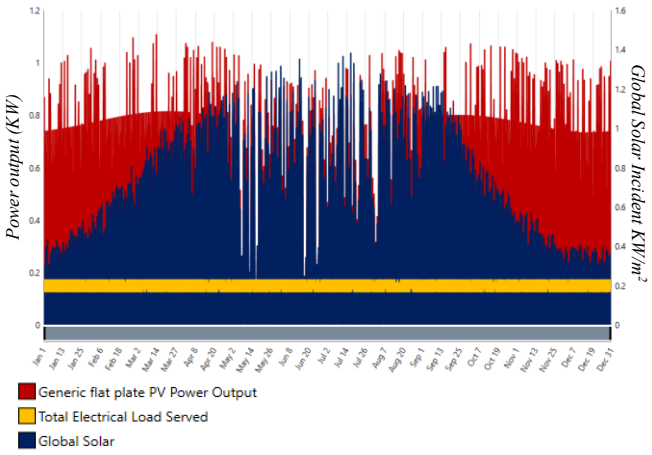


Figure 15: Load profile and total energy distribution.

Network Architecture

The connection between the hardware components is based on the traditional client-server network architecture, also known as the master-slave model. Nowadays, this concept is the most widely used form of communication in standard computer network systems due to its simplicity, flexibility, and effectiveness. It keeps various processes centralized and makes them accessible to multiple other devices. In this way, different client units request and receive service from the primary server. However, this data exchange model requires accessibility on both nodes (client and server). In a typical TCP network infrastructure, it is done by default using unique Internet Protocol Addresses (IP). These are 32-bit addresses (IPv4) divided into four 8-bit integer numbers from 0 to 255 separated by a dot, such as 192.168.0.10.

Moreover, each connection is assigned a fixed port number, respectively, on the client- and server-side to allow parallel data transfer to different processes between the same pair of endpoints. Many of those numbers are already predefined and assigned to a particular communication protocol and service type. For Instance, port 22 is reserved for Secured Shell (SSH) and port 1883 for Message Queuing Telemetry Transport (MQTT). Consequently, the server can precisely identify the addressed requests and immediately forward them to the corresponding processes according to the port number. Depending on the parent network structure, additional devices, such as routers, are required to send the data packets between two components from different subnetworks and increase the security aspect. The router contains IP addresses from both subnetworks and can, therefore, respectively to its firewall, forward the data packets correctly. The subnetwork is defined using the so-called subnetwork-mask.

According to Figure 16, the smart energy house system consists of various sensors for temperature and humidity measurement, water level detection, control of the connection between the battery and the water pump, as well as the photovoltaic panel and the battery. All components continuously communicate wirelessly with the central unit, Raspberry Pi (RPi), over a router. The router is connected to the already existing parent network of the SRH University of Applied Sciences. Due to the public IP-Address of the University and appropriate Port-Forwarding, remote control of the measuring units and access to the measurement data with web-based software is possible out of the intranet.

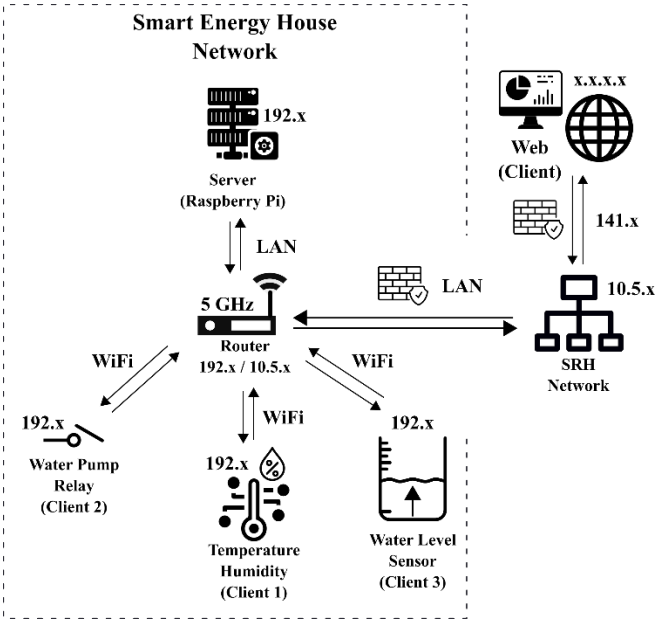


Figure 16. Schematic diagram of the hardware connection within the Smart Energy House Network.

Messaging System and Telemetry Services

A fundamental part of an intelligent control system is a stable and real-time messaging mechanism to integrate a large number of sensors and actuators. One such tool that has become an ISO-standard (ISO/IEC 20922) in IoT is the Message Queue and Telemetry Transport (MQTT) protocol. Originally, it was created by, among others, IBM in order to collect information from widely distributed infrastructures [7]. MQTT is a very lightweight machine-to-machine (M2M) communication protocol for passing messages between devices. It is especially suitable for units with limited resources as well as low-bandwidth, high-latency and unreliable networks [18]. MQTT is based on TCP/IP, but it might also be used for other communication protocols like UDP or ZigBee, although an extension MQTT for Sensor Networks (MQTT-SN) will be required [19, 20].

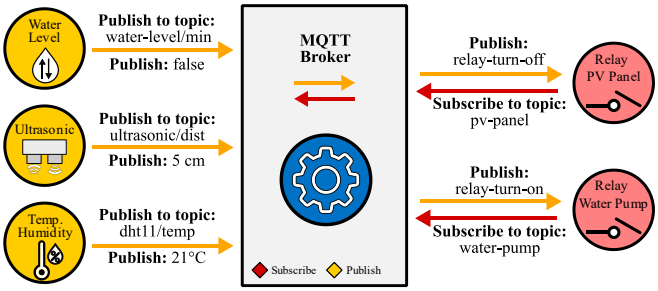


Figure 17. Principle of the publish/subscribe communication concept.

As shown in Figure 17, the protocol implements the so-called publish/subscribe communication concept, an event-based model between clients with a message-passing broker in the middle. A device (client) can send a specific message, an event, or subscribe to a particular topic to receive the notification. The event represents information that one client wants to share with other devices. It could be a physical value such as temperature or humidity measured by a sensor or a request for action addressed to an actuator. Every single message is tagged using a topic name, which is a simple string

separated by slashes. Each forward slash indicates a topic level, specifying the type and source of the content. This way, a topic tree, which defines a hierarchical structure for the topic names, can be set up. Additionally, MQTT offers two wildcard options: single-level (+) and multi-level (#), which can be integrated into the string. For Instance, the following topic: *relay/ch1/status* refers to the relay's first channel informing if it is switched on or off. Suppose the subscriber is interested in the status of all channels. In that case, he could use the single-level wildcard in the topic filter *relay/+/status* or the multi-level wildcard with the topic filter *relay/#* to get all messages related to the relay. Once the topic name matches a topic pattern, the information is forwarded to the subscribed client. The data transmission is controlled by a central unit, the broker (server). The broker is responsible for receiving published messages, filtering the topic names and forwarding the messages to the interested clients. The devices can be very simply connected and disconnect from the server.

MQTT has three defined Quality of Service (QoS) levels: 0 – at most once, 1 – at least once, 2 – exactly once [21]. The lowest level 0 implies that the sensor only sends the data without any broker confirmation. Levels 1 and 2 are advanced handshake methods that increase the reliability of message delivery. The protocol supports even a Last Will and Testament (LWT), a mechanism to inform the subscribers about the disconnected publisher. The communication through the MQTT can be encrypted using Transport Layer Security (TLS) and additionally protected by modern authentication protocols, such as OAuth.

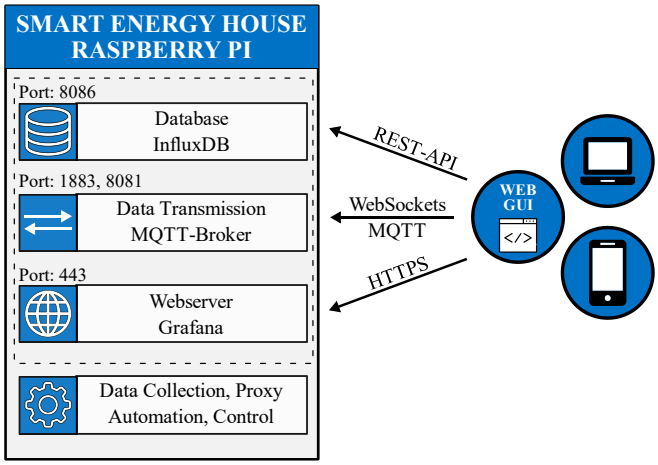


Figure 18. Schematic diagram of server services and access methods.

In this study, Raspberry Pi is a primary and central unit of the entire monitoring system that runs different services for data transmission, data collection, processing, automation and visualization (Fig. 18). The transmission of the sensor data and the control of the actuators is entirely done through the mosquito software, widely supported on many platforms and also on the Raspberry Pi. The temperature, humidity, and water level detected by the analog and digital sensors are sent to the MQTT broker in JavaScript Object Notation (JSON). JSON is built on an ordered list of values and a collection of name/value pairs [22]. The message contains the information about the timestamp (unix time, integer), type of device (string), location (string), measured value (float) and units (string). For Instance, the following temperature measurement comes from a DHT11 sensor located on the PV panel.

```
{
    "timestamp": 000000,
    "device": "DHT11",
    "value": 38,
    "units": "degrees",
}
```

The measured data are stored in a time-series database system, such as InfluxDB. The predefined python processes control the relay channels and, consequently, start and stop the water pump or the battery and PV array connection. The system can be monitored in real-time directly in the browser. The self-programmed REST-API used with the web-based platform Grafana offers secure data access and visualization. The system can be controlled and reconfigured from any device, such as laptops, smartphones, or tablets. Many open-source desktop software, like mqtt-spy and MQTT.fx, and Android applications, such as MyMQTT support this protocol. For browser-based solutions like HiveMQ, no software installation is required, but the broker must support the WebSocket communication.

Data Visualization and Alarming

The tasks of a monitoring system go far beyond simple data collection. In addition to sensor control and local storage, the network communication, alarming and remote maintenance are essential parts of Smart Energy House.

Nowadays, data visualization is done using modern web technologies. The logs, reports, and graphs can be formatted using HTML, CSS, and JavaScript. WebSocket and REST-API enable external secured data access. Often the programming effort can be minimized by using already existing frameworks.

Combining the open-source Grafana software and InfluxDB has proven to be a useful tool for monitoring tasks. It offers alerting and data analysis functions in addition to visualization tools. Grafana allows users to create a dashboard with panels that display specific metrics over a specified time frame (Fig. 19). Each dashboard can be customized for project purposes. Additionally, the software provides an explore function for querying the datasets. This feature helps the user filter the measured data without the need to create a dashboard [23].

In the Smart Energy House project, Grafana is primarily used with MQTT and InfluxDB. The software supports a variety of other data sources, such as Prometheus, MySQL, and Postgres. For each data source, Grafana has a query editor and a specific syntax.

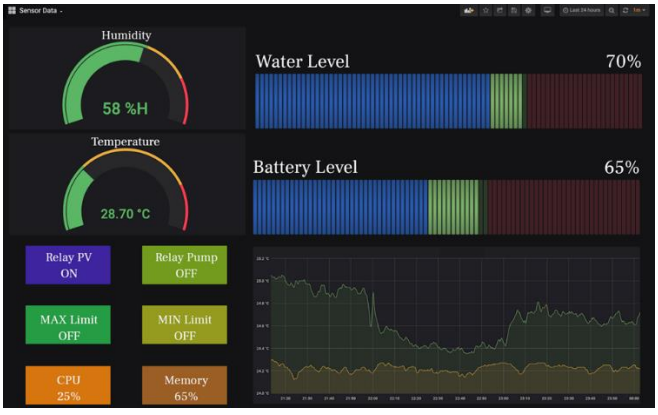


Figure 19. Dashboard of Smart Energy Home Monitoring System.

The alarm functions are intended to inform the user about problems and unusual conditions. These can be a critical limit of the battery or water level, as well as system failures. The user is

informed about the problem in time with an electronic message, e-mail. An alert consists of two parts, alert rules, and notification channels. One or more conditions define alert rules that Grafana regularly evaluates. The notification channel defines how the alert is being delivered. When alert rule conditions are met, Grafana notifies the channels configured for that alert [23].

Outlook

This simulation model will be implemented in the test object located in Berlin (Fig. 20). The house is made of clay instead of brick for better insulation. The clay block walls ensure a balanced indoor climate and protect against sudden temperature changes, which allow the owners to control all the parameters affecting their comfort and optimum management of the maintenance cost. The photovoltaic thermal panel will be installed on the roof. The PV thermal board will provide thermal energy and can also be connected directly to the water pump system. The primary function is to deliver water by a DC pump rated at 70 W. For this purpose, the pipe with inner diameter of 12 mm and outer diameter of 16 mm will be used. Its length should be around 240 m. One hot water pump will be applied as a submersible device rated at 40 W. This solar pump will run directly by the DC of PV or the battery for providing hot water from the tank. This project aims to design a small system prototype for a zero-emission house, but it can be adapted for more complex targets [6].

Figure 20: Smart Energy House in Berlin (Germany).

Conclusions

Monitoring systems often consist of individual solutions, which are adapted depending on the project's tasks. This article presents fundamentals for a smart monitoring system based on the simulation model of a self-refilling tank energized by a battery connected to the photovoltaic panel. The design and hardware implementation were improved and completed with the aid of IoT technology. The possibilities were analyzed and successfully demonstrated to reduce the temperature impact and minimize the power source's effect by controlling the system load. The system offers various features for collecting and analyzing the measured data. Besides, it provides applications for electrical control, data query, and alarming technologies. The system can be configured to the user's needs. The AI algorithms allow predicting the energy consumption and the costs.

The authors see great potential in monitoring systems in combination with the Internet of Things and Renewable Energy. This paper presents an ongoing project that will be implemented in Smart Energy House located in Berlin. However, more sensors must be integrated into the prediction and cost calculation process to achieve more precise results.

References

[1] BMWi, "Bundesministerium für Wirtschaft und Energie", Federal Ministry for Economic Affairs and Energy, 2021. https://www.bmwi.de/Redaktion/EN/Dossier/renewable-energy.html (accessed Feb. 08, 2021).

[2] M. A. Ehsan, M. M. I. Mahfuj, and G. M. I. Hossain, "Smart Home Automation and Security System", 2016.

[3] Y. Jiang, X. Liu, and S. Lian, "Design and Implementation of Smart-Home Monitoring System with the Internet of Things Technology", Wireless Communications, Networking and Applications, Springer India, 2016.

[4] G. Kortuem, F. Kawsar, D. Fitton, V. Sundramoorthy, "Smart objects as building blocks for the internet of things", Internet Computing, IEEE, vol. 14, no. 1, 44-51, 2010.

[5] R. Piyare and S. R. Lee, "Smart Home-Control and Monitoring System Using Smart Phone", ICCA, vol. 24, 83-86, 2013.

[6] S. Islam, L. Rojek, M. Hartmann & G. Rafajlovski, "Artificial Intelligence in Renewable Energy Systems Based on Smart Energy House", IJITS International Journal on Information Technologies & Security, vol. 12, no. 4, pp. 3-12, 2020.

[7] V. Ziemann, "Series In Sensors: A Hands-On Course in Sensors Using the Arduino and Raspberry Pi", CRC Press – Taylor & Francis Group, 2018.

[8] ARM, "Single-Board Computers", 2021. https://www.arm.com/why-arm/partner-ecosystem/single-board-computers (accessed Feb. 08, 2021).

[9] G. Warren, Technology in Action, "Advanced Raspberry Pi: Raspbian Linux and GPIO Integration", Apress, 2018.

[10] Raspberry Pi, "Raspberry Pi 4 Computer", 2021. https://www.raspberrypi.org (accessed Feb. 08, 2021).

[11] S. Rizone and A. Yusoff, "The Development of an Automated System in Detecting Environmental Data for the Monitoring of Forest Activity", International Journal of Environmental Science and Development, vol. 7, no. 7, 2016.

[12] K. B. Swain, G. Santamanyu, and A. R. Senapati, "Smart Industry Pollution Monitoring and Controlling using LabVIEW based IoT", IEEE 3rd International Conference on Sensing, Singal Processing and Security (ICSSS), 2017.

[13] K. P. Kuria, O. O. Robinson, and M. M. Gabriel, "Monitoring Temperature and Humidity using Arduino Nano and Module-DHT11 Sensor with Real Time DS3231 Data Logger and LCD Display", International Journal of Engineering Research & Technology (JERT), vol. 9, no. 12, 2020.

[14] FREDERIKSEN, "Speed of sound in air", 2016. https://int.frederiksen.eu/Admin/Public/DWSDownload.aspx?File=%2fFiles%2fFiles%2fexp%2f131000%2f131410-EN-Speed-of-sound-in-air.pdf (accessed Feb. 08, 2021).

[15] O. S. Sutresman, R. Syam, A. H. Muchsin, Z. Djafar, S. Arief, A. A. Mochtar, A. Hayat, O. S. Sutresman, Amiruddin and A. M. Radja, "Experimental Analysis of Smart Green House with Rotary Garden",

The 5th International Symposium on Material, Mechatronics and Energy, 2019.

[16] J. A. Bell, "Modern Diesel Technology: Electricity & Electronics", 2nd Edition, Delmar Cengage Learning, 2014.

[17] HOMER, "Hybrid Renewable and Distributed Generation System Design Software", 2021. https://homerenergy.com (accessed Feb. 08, 2021).

[18] S. Herle, R. Becker, and J. Blankenbach, "Bridging GeoMQTT and REST", RWTH Aachen University, 2016.

[19] S. Herle and J. Blankenbach, "GeoPipes Using GeoMQTT", Geospatial in a Changing World, Springer, 2016.

[20] A. Stanford-Clark and H. L. Truong, "MQTT for Sensor Networks (MQTT-SN) Protocol Specification Version 1.2", IBM Zurich Res. Lab., 2012.

[21] HIVEMQ, "Getting Started with MQTT", 2021. https://www.hivemq.com/blog/how-to-get-started-with-mqtt/ (accessed Feb. 08, 2021).

[22] JSON, "Introducing JSON", 2021. https://www.json.org (accessed Feb. 08, 2021).

[23] GRAFANA, "Grafana Lab", 2021. https://grafana.com/docs/grafana/latest/ (accessed Feb. 08, 2021).

Author Biography

Lukasz Rojek (M.Sc.) is a Ph.D. candidate at the University of Bamberg in the field of Geography. He is also a Research Associate at SRH Hochschule Berlin and guest lecturer at the Beuth University of Applied Science in the field of Surveying and Geoinformatics.

Saiful Islam (M. Eng.) is a research associate at SRH University of Applied Sciences in the field of Renewable Energy and Electrical Engineering.

Prof. Dr. Michael Hartmann is the Academic Director of SRH Berlin School of Technology; Head of the Study Programs: Engineering and International Business; Engineering and Sustainable Technology Management.

Prof. Dr. Reiner Creutzburg is a retired Professor for applied computer science at the Technische Hochschule Brandenburg in Brandenburg, Germany. Since 2019 he is a Professor of IT Security at the SRH Berlin University of Applied Sciences, Berlin School of Technology. He is a member of the IEEE and SPIE and chairman of the Multimedia on Mobile Devices (MOBMU) conference at the Electronic Imaging conferences since 2005. In 2019, he was elected a member of the Leibniz Society of Sciences to Berlin e.V. His research interest is focused on Cybersecurity, Digital Forensics, Open Source Intelligence (OSINT), Multimedia Signal Processing, eLearning, Parallel Memory Architectures, and Modern Digital Media and Imaging Applications.

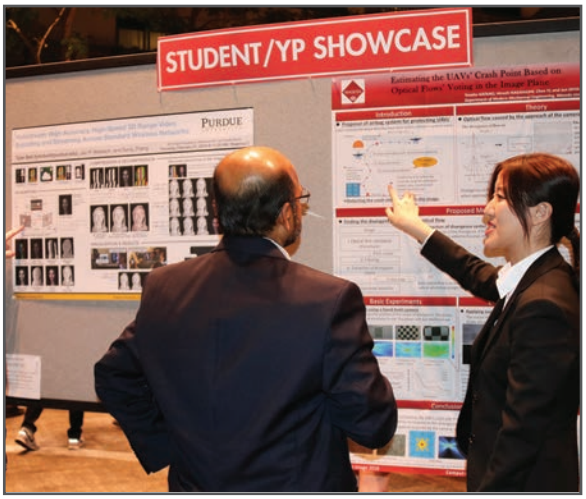
JOIN US AT THE NEXT EI!

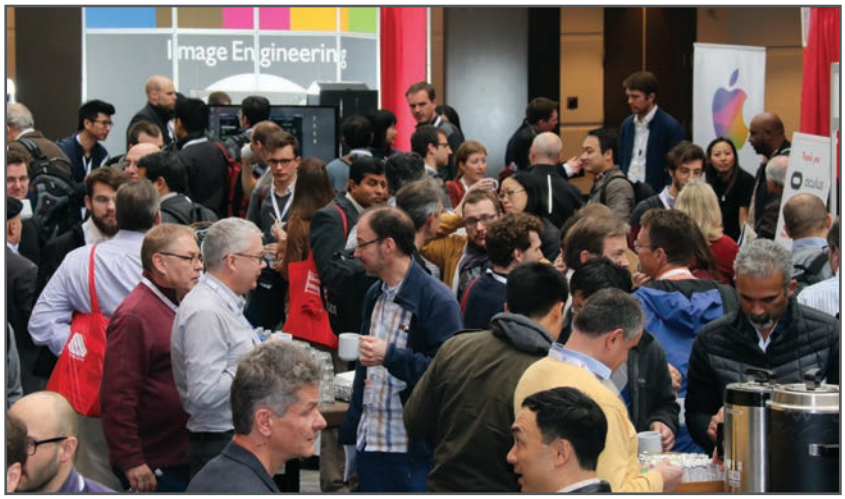
IS&T International Symposium on

Electronic Imaging

SCIENCE AND TECHNOLOGY

Imaging across applications . . . Where industry and academia meet!





• SHORT COURSES • EXHIBITS • DEMONSTRATION SESSION • PLENARY TALKS •
• INTERACTIVE PAPER SESSION • SPECIAL EVENTS • TECHNICAL SESSIONS •

www.electronicimaging.org

IS&T
imaging.org