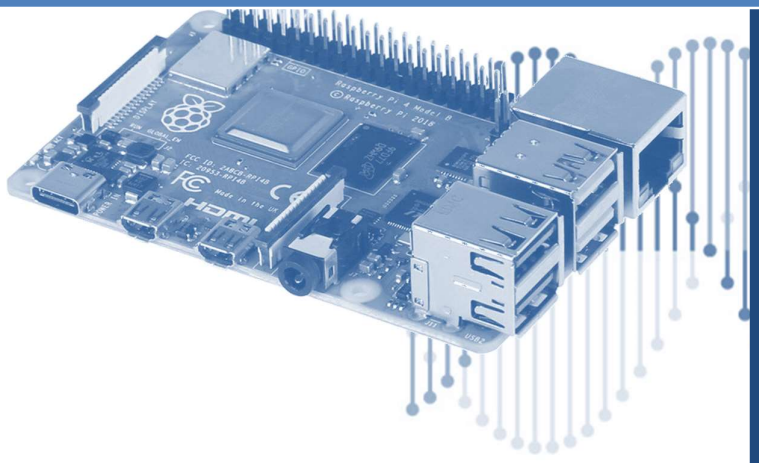
Assignment

Systems Engineering and Engineering Management ■ M.Sc.

Microprocessor based Systems

Assignment 2021 – Embedded Data Logger



Remote Version

Author:

Prof. Dr. Dominik Aufderheide

South Westphalia University of Applied Sciences, Division Soest

Faculty of Electrical Engineering

Section Industrial Measurement (*i*Me)

Fachhochschule Südwestfalen

University of Applied Sciences

iMe Industrielle Messtechnik / Industrial Measurement

1. Introduction

In this assignment, the PI-Trokli development system shall be used to implement an embedded data logger with an extended functionality. The data logger can be used to visualize sensor data acquired from the ultrasonic ranging module installed within the PI-Trokli development system. For this, a numerical and graphical representation of the actual measurements is provided by the system. This basic functionality shall be defined in section 2.1. Furthermore, the actual update rate, which defines how often a new sample is taken, can be defined by the user as described in section 2.2. Besides the ability to show current measurements in real time, the system also archives a set of a hundred measurements taken previously. When the actual gathering of data is stopped by the user, it is possible to scroll through the archive, as presented in section 2.3. As an additional routine, the system implements a calibration procedure, where the actual readings of the sensor are scaled according to a set of measurements taken. The exact routine shall be defined in section 2.4.

This assignment is designed in a way, which guarantees that the different basic functionalities can be implemented as separate Python scripts. Therefore, it is possible to implement each function more or less independently from the other elements, which shall support the development process. The following section will introduce all functionalities one-by-one with specific hints on how the associated scripts shall be developed, while all elements shall be combined in order to finalize the overall program. For each individual section the associated programming and documentation tasks are defined and the associated points within this assignments marking scheme are defined in order to provide a transparent and comprehensible rating of each students performance.

The deadline for the submission of the assignment solution is set to be August, 29th at 11:59 p.m. Every student who failed to submit their solution files (documentation and Python scripts) until that particular date and time will be charged with a score penalty based on how late they turned in their files. For this, the achieved score will be multiplied by a factor as defined within the following table.

Up to one day too late	Achieved points will be multiplied by 0.9
Up to two days too late	Achieved points will be multiplied by 0.8
Up to four days too late	Achieved points will be multiplied by 0.7
Up to seven days too late	Achieved points will be multiplied by 0.6
More than seven days too late	Achieved points will be multiplied by 0

If a student will not reach at least 50% of the total achievable points (after application of the late penalty multiplier), the assignment will be rated as failed and the student needs to repeat the module.

2. Tasks and Specifications (70 Points)

The following sections provide an introduction to the functional specification of the data logger. As already mentioned in the previous section, for each function specific programming and documentation tasks are formulated and the associated marking scheme is defined.

2.1 Real-Time Data Logging (20 Points)

The basic functionality of the data logger is its ability to visualize actual measurements from the ultrasonic ranging module in real time in a textual and visual way. For this, the following elements of the programming shall be implemented.

2.1.1 Acquisition of range measurement data (2 Points)

A function in Python shall be implemented, which acquires data from the ultrasonic ranging module. The function shall deliver a new measurement each time it is called. The output shall contain the measured pulse time of the echo pulse as provided by the echo pin of the ultrasonic ranging module.

2.1.2 Visualizing the distance measure on the 7 segment display (3 Points)

The gathered pulse duration shall be scaled to a value between 0% and 100% and subsequently visualized on the 7-segment display (7SD) as a floating point number with a single decimal digit. At this stage the scaling can be done with a random dependency, since the scaling factor shall be later automatically be calculated by a calibration routine. The following *Figure 1* provides an example of the associated visualization on the 7SD for different measurements.

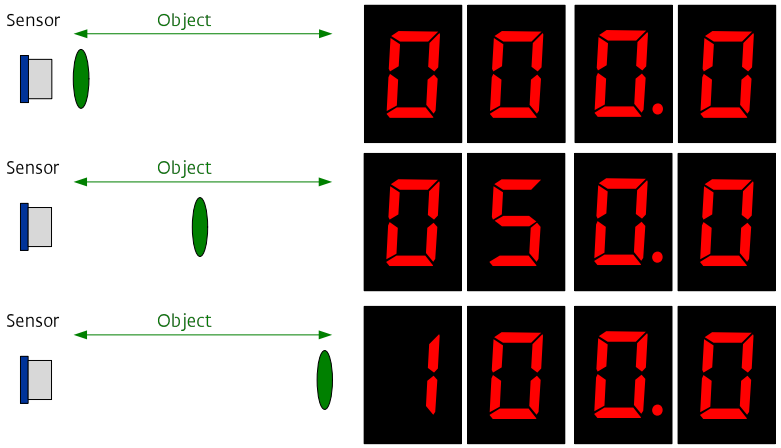


Figure 1 - Visualization of range data on the 7-segment display

The actual functionality of showing the measurement on the 7SD shall be implemented within a Python function, which has the actual distance measure as a parameter.

If the remote PI-trokli system is used, the functionality of the function can be tested by using keyboard inputs from the console. For this the `input` function can be used in Python.

2.1.3 ■ Visualizing the distance measure on the Matrix LED Display (MLD) (10 Points)

In addition to the textual representation of the measurement data on the 7SD, the data shall be also graphically visualized on the matrix LED display (MLD). Here the last eight samples shall be shown as a graph on the MLD, as indicated in the following figure. Here, each sample is indicated by a single data point where its position on the MLD is defined by the range (x-coordinate on MLD) and sample index (y-coordinate on MLD). Here the latest sample x[n] is always shown on the rightmost column of the MLD, while the sample x[n-7] is shown on the leftmost column. Every time a new sample is acquired, the "oldest" sample will be removed from the MLD, so that a moving graph is generated.

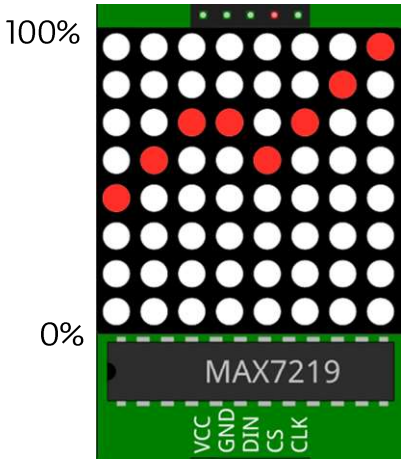


Figure 2 - Visualization of range data on the Matrix LED Display

2.1.4 ■ Test setup for the basic functionality (5 Points)

In order to test the aforementioned elements of the basic functionality a first test setup shall be implemented, where each second a new range measurement is gathered and then visualized on the 7SD and the last eight data points shall be shown on the MLD.
In order to implement the possibility for a simple test on the remote access systems, the reading of the ultrasonic sensor shall be emulated by a random value. A random value between 0 and 1 can be generated in Python by using the `random` function from the `random` module. The following example shows the generation of a random value in a simple Python script. In

order to provide a more realistic test setup, the random value shall be scaled to a floating point number between 0 and 200.

```
import random

rand_number = random.random();
print(rand_number)
```

2.2 ■ Update rate (10 Points)

In the former task the program was implemented in a way, that the update rate is fixed to one second. As a first extension to the basic functionality, a possibility to change the update rate shall be implemented. For this, a dedicated button of the PI-trokli system shall be utilized. For this the "up"-button, as indicated in *Figure 3*, shall be used. Each time the button is pressed the delay time between two successive measurements is incremented by one second, where nine seconds would be the maximum, so that the time is then reset to 1 second in case the button is pressed again.

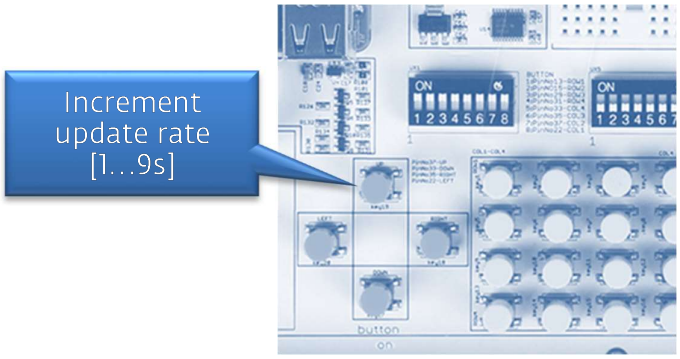


Figure 3 - Visualization of range data on the Matrix LED Display

Since the buttons are not easy to be accessed on the remote access modules, the functionality can be also tested by using a keyboard entry, where the update rate can be directly set from a keyboard entry.

2.3 ■ Archive option (30 Points)

In addition to the possibility to gather and visualize data in real time, the program shall be also equipped with an archive, which stores the last 100 values captured from the ultrasonic sensor. For this, the Python program shall be extended by the following functions.

2.3.1 ■ Implementation of FIFO data buffer (10 Points)

A First-In-First-Out (FIFO) buffer shall be implemented which can store up to one hundred data points. As soon as the buffer is completely filled, the oldest value within the buffer will be moved out of the buffer (dequeued) when a new value is acquired (enqueued). The buffer can be implemented as an array in Python.

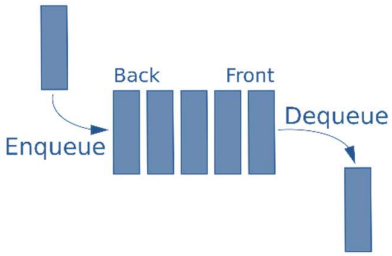


Figure 4 – FIFO-buffer structure

2.3.2 ■ Visualization of archive elements (20 Points)

The system shall now be extended by an additional mode, where the actual gathering of new values is paused. The system can be paused or restarted by pressing the "left"-button as shown in the following figure. If the button is pressed once, the system will be paused, while in paused mode an additional operation of the same button will restart the system.
During times where the system is paused, the user can navigate through the archived values by using a dedicated button of the user interface ("right"-button). The associated value of the actual buffer element is then visualized on the 7SD and the actual value and the 7 values gathered before are also shown on the MLD.

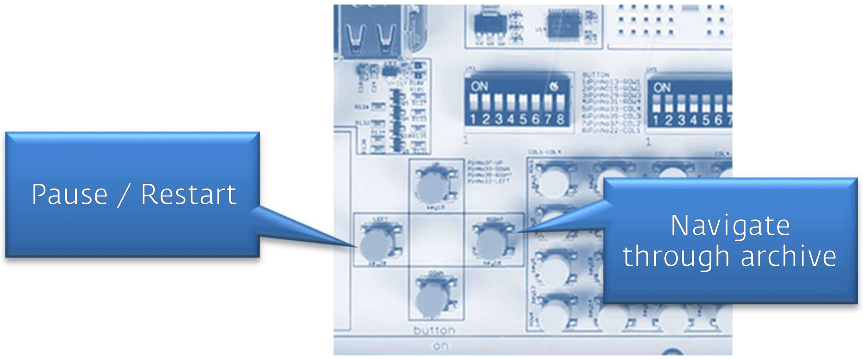


Figure 5 - Visualization of range data on the Matrix LED Display

Since the buttons are not easy to be accessed on the remote access modules, the functionality can be also tested by using a keyboard entry, where the update rate can be directly set from a keyboard entry.

2.4 Calibration (10 Points)

For all functions above a random scaling of the range measurements is used in order to derive a value between 0% and 100%. In order to provide the possibility to define a possibility for the user to redefine the scaling an automatic calibration routine shall be implemented. The idea of the calibration routine is the automatic calculation of the calibration coefficient for scaling the echo pulse time to a value between 0% and 100%. For this, after a start of the calibration routine, which can be triggered by pressing the "down"-button, a set of 10 measurements is taken within a 5 second period (each 0,5s one measurement) from the ultrasonic ranging sensor. During this time, the buzzer of the PI-Trokli system is indicating that a calibration is ongoing. As soon as all measurements are taken, the relation α between the echo pulse time t_{echo} and the percentage value P [%] is automatically calculated from the mean value of those measurements $\overline{t_{echo}}$.

$$P = \alpha \cdot t_{echo} \quad \text{with} \quad \alpha = \frac{100\%}{\overline{t_{echo}}}$$

3. Documentation (30 Points)

Each student shall create a documentation for the individual code developed. The documentation shall contain a detailed description of the developed program according to the program structure presented in section 2. Here adequate structural elements, such as pseudo code representation and/or flow diagrams shall be used in order to visualize the developed code.