Hardwareprojekt: Raspberry Pi

Generelle Vorgehensweise

Sie sind in Gruppen zu 3-4 Personen aufgeteilt und haben einen Gruppennamen. Die Verwaltung des Codes findet über GitHub statt.

Ziel des Hardwareprojekts

Ziel ist es Temperaturmessdaten von einem Raspberry Pi über das Netzwerk zyklisch an einen MQTT-Server zu senden. Diese Messdaten sollen dann von einen GUI-Programm an einem Desktoprechner abgefragt und grafisch dargestellt werden. Die Art der Visualisierung wird selbst gewählt.
Denkbar ist die Anzeige aller Werte auf Knopfdruck oder etwas ähnliches wie ein Oszilloskop.
Das GUI soll per Push-Button beendet werden, eventuell könnte man die empfangenen Daten speichern. Zusätzlich soll noch eine per Raspberry Pi gehostete Website die Messdaten visualisieren.

Hilfestellung

Es wird der Raspberry Pi 3B mit installiertem Raspbian bereitgestellt, dieser ist für das HAW-Netz freigeschaltet. Zusätzlich können Sie den MQTT-Server der HAW nutzen.

Raspbian OS

Raspbian OS ist ein an den Raspberry Pi angepasstes Debian, eine bekannte Linux-Distributionen.
Viele Server und auch Embedded-Geräte nutzen Linux als Betriebssystem.
Der Großteil der Software, die Sie benötigen ist bereits vorinstalliert. Nutzername und Passwort für den Raspberry lauten wie folgt: Name: pi, Passwort: raspberrz
Die Steuerung findet über die Konsole (BASH) statt. Wichtige Befehle hierfür sind:

Befehl	**Bedeutung**
ls <Pfad>	Listet die Dateien und Ordner im aktuellen Verzeichnis auf.
cd <Pfad>	Ändert das Verzeichnis in dem man sich befindet.
mkdir <Pfad>/Ordnername	Legt einen neuen Ordner an.
sudo <Befehl>	Den Befehl mit Administrator-Rechten ausführen.
ssh <User>@<Computer>	Stellt eine SSH-Verbindung her
screen -S <Jobname>	Startet eine Konsole im Hintergrund. Verlassen mit: STRG+A+D
screen -r <Jobname>	Kehrt zu der Konsole zurück.
STRG+C	Beendet den aktuell laufenden Befehl
date --set 'JJJJ-MM-DD hh:mm:ss'	Stellt die Uhr ein, nötig, da es keine CMOS-Uhr gibt

MQTT-Server

Das MQTT-Protokoll ist gut für die Übertragung von Sensordaten geeignet. Es setzt das Publish–Subscriber-Model um. Der **Publisher** sendet die Daten an ein benanntes **Topic** auf den Server. Von dort aus wird dieses an alle **Subscriber** weitergeleitet, die das **Topic** abonniert haben.
Verwenden Sie bitte den Gruppennamen als übergeordnetes Merkmal.

Installation unter Linux: *apt install mosquitto-clients*
Unter Windows gibt es den MQTT-Explorer
Für Python gibt es z.B. das Modul MQTT-Client

Die Adresse des HAW/MuP MQTT-Servers lautet: 141.22.194.198
Testen lässt sich dieser indem man den MQTT-Client startet und auf alle Topics lauscht:
mosquitto_sub -h 141.22.194.198 -t \#
Und auf das „Topic“ "*<Gruppenname>/Test*" sendet man mit einem anderen PC:
mosquitto_pub -h 141.22.194.198 -t <Gruppenname>/Test -m "hello world"

Wiederkehrende Abfragen

Sensorik sollte nicht in einer while-true-Schleife abgefragt und versendet werden, die nur durch die CPU-Geschwindigket begrenzt wird. Es ist sinnvoll zumindest abzuwarten, damit es nicht sehr viele Daten zu zufälligen Zeitpunkten gibt. Dafür kann man die Funktion sleep(<SEKUNDEN>) aus dem Modul time nutzen. Mit Zahlen im Bereich zwischen 0 und 1 aufgerufen, wartet diese im ms Bereich.
In der fortgeschrittenen Programmierung wird mit Timern gearbeitet. Die Ausführungsschleife blockiert bis ein Timer die Ausführung wieder freigibt. Dies führt dazu, dass die Aufrufe ohne Verschiebung etc. in genauen zeitlichen Abständen ausgeführt werden.

SSH-Verbindung

Um an dem Raspberry Pi nicht mit Tastatur und Maus arbeiten zu müssen, kann man sich über das Netzwerk per SSH mit diesem verbinden. Hierüber können alle vorgestellten Linux-Befehle und die GIT-Befehle ausgeführt werden. Unter Linux ist es über X11-Forwarding möglich, dass Programme mit GUI ausgeführt werden. Unter Windows kann man für SSH die Powershell oder Putty nehmen.

Arbeiten außerhalb der Hochschule

Außerhalb der Hochschule ist die Arbeit leicht anders: Die IP-Adresse ihres Raspberrys ist eine andere, diese lässt sich meist über die Weboberfläche des Routers in dem „DHCP-Menü“ nachschauen.
Der zentrale MQTT-Server ist evtl. nicht erreichbar. Mit einem Linux-Rechner, wie z.B. dem Raspberry, kann man sich schnell einen MQTT-Server selber aufsetzen: sudo a*pt install mosquitto*
Die Raspberrys haben einen Aufkleber mit der IP, die IPs beginnen mit 141.22.36.*.
Mosquitto-Server für Windows: https://mosquitto.org/download/

Vorschlag zur Vorgehensweise

Eine Aufteilung in vier Klassen (Sensor, Kommunikation: Pi zu MQTT und MQTT zu PC, Webserver, sowie Visualisierung) wird empfohlen. Das Grobkonzept wird im laufe der Zeit verfeinert.
Eine Bearbeitung des Projekts könnte auf dargestellte Weise erfolgen:
Milestone 1: Konzept, Inbetriebnahme Raspi, Implementation: Sensor-Stubs und Pi zu MQTT Komm., Implementieren der Abfrage von MQTT mit Desktop-GUI
Milestone 2: Implementieren der Sensoransteuerung, Implementation Website

Beispielprogramm

Mit den Codebeispielen von https://www.emqx.com/en/blog/how-to-use-mqtt-in-python wurden zwei Beispielprogramme erzeugt, welche eine minimale Möglichkeit zur Kommunikation per MQTT zeigen.

Installation der Pakete etc.:

```
sudo pip install paho.mqtt #Systemweit MQTT installieren
```

Test-Code "sendMQTTShort.py":

```
from paho.mqtt import client as mqtt_client
from random import randint

MQTT_PORT=1883
MQTT_ADDRESS="141.22.194.198"
MQTT_CLIENT_NAME="Test_PiPub"
MQTT_TOPIC="Gruppenname/Kanal"

#Hier steht die Nachricht
text="Random Value: " + str(randint(1, 100))

#Send Text to MQTT
client = mqtt_client.Client(MQTT_CLIENT_NAME)
client.connect(MQTT_ADDRESS, MQTT_PORT)
client.publish(MQTT_TOPIC, text)
```

Test-Code "readTemp.py":

```
from paho.mqtt import client as mqtt_client
import time

MQTT_PORT=1883
MQTT_ADDRESS="141.22.194.198"
MQTT_CLIENT_NAME="Test_PiSub"
MQTT_TOPIC="Gruppenname/Kanal"
TICK_RATE_HZ=2
TICK_RATE=1/TICK_RATE_HZ

#Liste für die Nachrichten
message_queue = []
#Funktion die beim eintreffen einer Nachricht diese in eine Liste packt
def on_message(client, userdata, msg):
    message=msg.payload.decode()
    message_queue.append(message)
#Verbinden mit MQTT
client = mqtt_client.Client(MQTT_CLIENT_NAME)
client.connect(MQTT_ADDRESS,MQTT_PORT)
#Subscribe zum Thema und Funktion "on_message" hinzufügen
client.subscribe(MQTT_TOPIC)
client.on_message = on_message
#Für immer laufen lassen
client.loop_start()
while (True):
    last_value=""
    value_queue = []
    #Kurz warten
    time.sleep(TICK_RATE)
    #Wenn es noch neue Nachrichten gibt...
    while len(message_queue)>0:
        #...dann kopiere diese von der Empfangsliste in die Anzeige-Liste
        last_value=message_queue.pop()
        value_queue.append(last_value)
        print(last_value)
```

Starten der Programme auf beiden Geräten:

Raspberry: *python sendMQTTShort.py*
PC: *python readMQTT.py*

Sensorik

Das Beispiel oben zeigte die Abfrage des integrierten CPU-Temperatur-Sensors des Raspberry Pis.
Ziel ist es aber, einen Temperatur-Sensor zum Laufen zu bekommen.
Bei den Beispielen in der Doku wird Python 2 Syntax verwendet. Da sind manche Befehle und Klammern anders und es ist nicht direkt unter Python 3 ausführbar. Der Code muss also gegebenenfalls angepasst werden.
Aktueller Code findet sich auf der Herstellerseite: https://sensorkit.joy-it.net/de/

Für den Anschluss ist es wichtig, dass Sie sich heraussuchen, mit wie viel Volt der Sensor betrieben wird. Der Raspberry stellt 3,3V und 5V bereit.
Auch müssen Sie sich entscheiden, an welchen der GPIO-Pins das Datensignal des Sensors angeschlossen wird. Die verfügbaren Pins sind hier ganz gut zu erkennen:
https://www.elektronik-kompendium.de/sites/raspberry-pi/1907101.htm
Die Pins der Sensoren werden dann mit Verbindungskabeln mit den Pins der Raspberry-Steckerleiste verbunden. Am besten zuerst die Masse (GND), dann das Signal und am Ende die Versorgungsspannung anschließen.

Die Sensoren sollen in einem der Seminarräumen aufgestellt werden, um dort die Nacht- und Wochenendabschaltung der Heizungsanlage zu erfassen. Nach einer kurzen Testphase ist der Sensor also physisch nicht mehr zu erreichen. Beachten Sie dies bitte bei ihren Planungen.

Projekt-Daten mit dem Raspberry synchronisieren

Man kann das Textverarbeitungs-Programm "nano" verwenden und damit die Python-Scripte direkt auf dem Raspberry ändern.
Es bietet sich an ein GIT-Repository anzulegen und zu nutzen. Dadurch kann man auf einem Desktop-Rechner eine Entwicklungs-Umgebung nutzen und von dort aus die Änderungen in das GIT hochladen. Dann kann man sich per SSH mit dem Raspberry verbinden, mit "git clone" einmal das Projekt ablegen und dann die Änderungen jeweils mit "git pull" herunterladen.

Arbeiten ohne Zugriff auf den Raspberry

Sie können an dem ersten Milestone arbeiten, welcher hauptsächlich die Visualisierung von per MQTT empfangenen Daten beinhaltet. Dazu gibt es zwei Möglichkeiten:
1.) Sie senden per MQTT-Explorer an das Topic Ihrer Gruppe händisch Daten, die dann von dem Desktop-Programm empfangen und visualisiert werden.
2.) Sie senden mit einem Python-Programm, ähnlich wie dem "sendMQTTShort.py"-Beispiel, Pseudodaten an das Topic Ihrer Gruppe.

GUI-Programmierung

es empfiehlt sich eines der möglichen Frameworks auszusuchen und dann gemäß der beigefügten Links daran zu arbeiten, um z.B. Matplot in einem Fenster darzustellen. Hilfreiche Links:
https://datatofish.com/matplotlib-charts-tkinter-gui/
https://build-system.fman.io/pyqt5-tutorial
https://www.pythonguis.com/tutorials/plotting-matplotlib/
https://matplotlib.org/stable/gallery/user_interfaces/embedding_in_tk_sgskip.html
https://matplotlib.org/stable/api/_as_gen/matplotlib.animation.FuncAnimation.html#matplotlib.animation.FuncAnimation
https://www.pythontutorial.net/tkinter/tkinter-grid/

Darstellung per Web-Oberfläche

Für die Darstellung per Web-Oberfäche wird das Framework Bottle genutzt. Der Aufbau ist wie folgt: Auf dem Raspberry Pi läuft das Programm mit dem die Temperaturen ausgelesen und per MQTT gesendet werden. Zusätzlich wird auf dem Raspberry Pi ein Webserver gestartet, welcher eine Website erzeugt, welche die Live-Temperatur im Browser anzeigt und zusätzlich die gesammelten Temperatur-Daten zum Download als csv-Datei anbietet, um diese für Machine-Learning nutzen zu können.
Auf der Projektseite werden die nötigen Informationen bereitgestellt. Es empfiehlt sich das “Hello-World-Programm“ zu erzeugen: https://bottlepy.org/docs/dev/tutorial.html#quickstart-hello-world
Um die Website zu besuchen, wird im Browser lokal die angegebene URL verwendet, soll die auf dem Raspberry Pi gestartete Website besucht werden, muss statt „*localhost*“ die IP-Adresse verwendet werden: http://<IP_ADRESSE_PI>:8080/hello
Das Framework muss wie jedes Python Modul installiert werden: pip install bottle
Man kann einen Webserver im Hintergrund starten, indem man die Threading Bibliothek nutzt:

```
from bottle import route, run
import threading, time

def background_server_function(name):
    run(host='0.0.0.0', port=8080, debug=True)

@route('/hello')
def hello():
    return "Hello World!"

threading.Thread(target=background_server_function, args=(1,), daemon=True).start()

while True:
    print("Mainloop is here.")
    time.sleep(1)
```

Fehlervermeidung

Raspberrys mit „poweroff“ herunterfahren, sonst kann es zu Problemen mit dem Dateisystem kommen.
Es muss eine Verbindung per VPN-Client oder ein RZBT-Rechner-Pool genutzt werden, um auf den MQTT-Server zugreifen zu können.
Über die „cmd.exe“ kann man mit dem Befehl „ping <IP>“ die Netzwerkerreichbarkeit prüfen.