ADVANCED EMBEDDED SYSTEMS

TP264 - Grammatikakis Miltos & Kornaros George

Project Topics

I. some idea you have about a semester-course project
II. some of the topics you will find in the Home directory » TERM_PROJECT2_EMBEDDED_DEVELOPMENT
III. bridging IoT protocols: (for example: https://github.com/dvdbng/mqtt-ws)
IV. implementation of a version of https://github.com/questdb/questdb on freertos or linux on some board (zynq, stm, other)
V. implementation of a version of https://github.com/LiamBindle/MQTT-C on freertos or linux on some board (zynq, stm, other)
VI. time-series sensor data prediction/anomaly detection by using statistical methods (least-squares regression, ARMA, ARIMA modeling), using and adapting open-source code
VII. time-series sensor data prediction/anomaly detection by using machine learning methods
VIII. time-series sensor data filtering, fusion by using kalman filtering

The goals are always: efficient implementation (fast, low resource usage, secure) by using software or hardware acceleration methods: HLS tools

Of course, whatever you decide, we will discuss it and help you get a basic example working and then you can optimize it yourself.

Project #99

Objective: Dynamic update of a function/task. This is of utmost importance in modern IoT/Embedded devices, when this device in the field should not be shutdown, but parameters and / or an algorithm needs to be updated.

For instance, replacing or updating the weight of a machine learning algorithm, or updated signature algorithm from SHA1 to SHA2, or just updating the secret/private parameters. A special memory can be used for this (probably the internal SRAM).

A negotiation protocol must be developed, in order to freeze the sub-task that needs to be updated.

You may assume that this flexible can be allocated in a separate thread (if RTOS is in place), or on another core.

Project #100:

As more and more embedded systems are battery-powered and trying to save energy, intelligent management of the large volume of data from their sensors requires new techniques for storing in the device's limited flash memory and sending this data to the cloud. For this reason, the goal of this project is to store data from a temperature sensor, for example, locally in the embedded system and (a) send a notification in case of need (e.g. temperature>threshold_red), (b) serve queries for:

i. average_100: average of the last 100 values

ii. (temp, time): the 3 highest temperature values and the time

iii. (temp, oldtime): the 1 (or 10) last temperature values before the time oldtime

Techniques:

A) since the embedded device does not have a real-time clock and the project is in FreeRTOS / baremetal, the application can start by asking to set the current time.

B) the time-series database can read the temperature sensor periodically (e.g. every 1 sec) and store time-temperature without pre-processing in a single data structure (queue?) or in multiple queues taking into account the possible queries it needs to execute.

C) the application can be in baremetal on one core, or two applications on two cores (one on each), or in threads using FreeRTOS

D) the data can be in memory, reserving at least 100KB of memory or written to SD. In any case, however, there should be a limit exceeding control.

E) Flexibility in data storage and speed in serving queries are key criteria in evaluating the solution to be implemented. For example, instead of a single database (table, circular linked-list) where raw values are stored, there may be a second, third, etc., with different information, e.g. the average, or the temperature per min, per 30 min, per hour.

Useful Links (maybe):

Lightweight databases for IoT: https://github.com/armink/FlashDB

https://www.devcoons.com/a-simple-generic-c-queue-for-static-or-dynamic-memory-usage/

https://www.freertos.org/mqtt/index.html

https://blog.reds.ch/?p=1671

https://www.techiedelight.com/circular-queue-implementation-c/

https://github.com/topics/lock-free-queue?o=asc&s=updated

https://github.com/topics/circular-queue?l=c%2B%2B

https://github.com/topics/circular-queue?l=c