TECHNISCHE UNIVERSITÄT
CHEMNITZ

Professur
Betriebssysteme

Embedded Software Lab

Lab 4 - RASPNet Layer 4

Christine Jakobs, Martin Richter

In this lab, you will implement layer 4 of the RASPNet protocol. You need a complete implementation of layers 1 to 3.

Initial Remarks

The tasks should be solved in the given order. If you do not manage to solve all tasks in preparation of the next lab meeting, make sure that you solve them afterwards. Your final grade at the end of the semester depends on the availability of *all* solutions in your code repository.

The presentation of your results during the lab slot is oral. There is *no need for preparing slides* or any other kind of written material.

Be ready to answer questions, both from the peer group reviewing you and the supervisors. Have your code, positive and negatives experiences and according documentation available.

Both team mates must participate in the presentation and discussion of results.

For the coding part, please avoid playing with the boot / read / write lock bits, playing with the Fuse bits or writing to the boot loader memory. Writing to EEPROM regions is also not needed at the moment.

Task 4.1

The transport layer of RaspNet is responsible for realizing the handling of discarded packages, which typically happens when layer 2 detects a CRC problem during frame receiving.

Study the design of RaspNet layer 4 [1]. Answer the following questions:

- Assuming a working implementation of layers 1..4, what kind of network messages / packages are possible in RaspNet?
- What kind of problems can be detected (and fixed) with the identification number approach?
- Is it possible that a RaspNet receiver node gets a message multiple times, while the sender still thinks that sending failed?
- How can timeout be implemented? Name at least two possibilities. What are the pros and cons about any of these options? Which one do you prefer and why?
- Re-sending is triggered by a decision on layer 4. Can you use a send buffer from layer 2 to realize that? What is the impact on buffer locking?
- If you could change the RaspNet protocol, would it be possible to realize acknowledge handling on lower layers? What are the pros and cons of this decision?

Task 4.2

Prepare the Gertboard wiring according to Figure 1. If a wiring already exists, check it for correctness.

Task 4.3

Implement the sending part of RaspNet layer 4 [1], first by leaving out the re-sending facility. Make sure that the implementations of Layer 1+2, Layer 3 and Layer 4 are decoupled, so that they would be exchangable by just swapping source code files. You layer 3 receiver implementation should be able to receive such packages correctly.

Task 4.4

Implement the receiver side of layer 4 on top of layer 3. Consider the different possible message types. Send ACK-messages in your receiver implementation, if needed.

Note:

State handling is an important aspect here. Make sure that you maintain an internal understanding of which packages were / are about to be acknowledged.

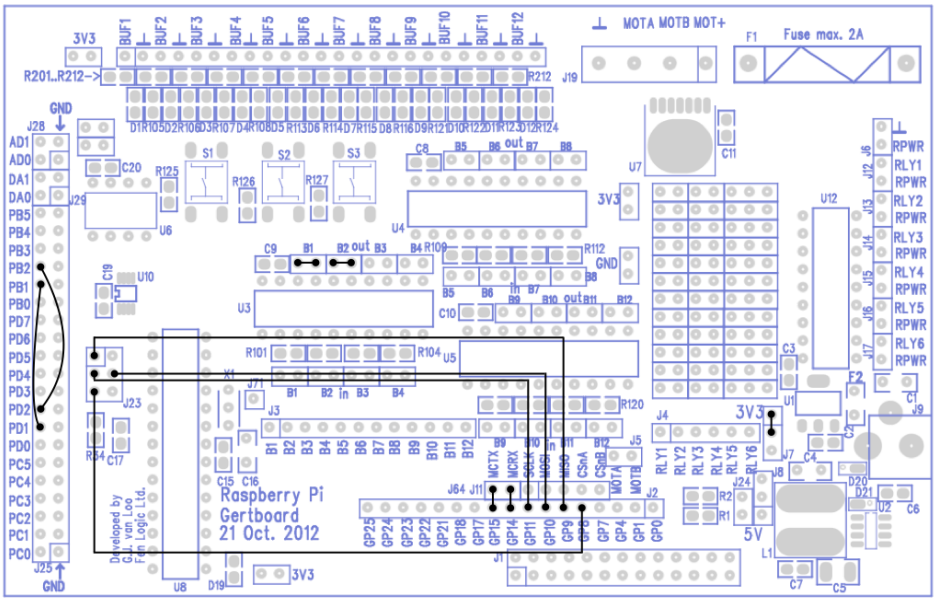

Figure 1: Gertboard wiring scheme for layer 4 development.

Task 4.5

Implement timeout-based re-sending on the sender side. It should be possible that the sender implementation notifies the application about a timeout case with a callback.

Task 4.6

Implement timeout-based re-sending on the sender side. It should be possible that the sender implementation notifies the application about a timeout case with a callback.

References

[1] Stefan Naumann. RaspNet - A simple realtime network protocol for microcontrollers. `https://osg.informatik.tu-chemnitz.de/lehre/emblab/raspnet.pdf`.