Präsentation der Studienarbeit

von Manuel Stübler

im Fach Technische Informatik

der Fakultät Informationstechnik

an der Hochschule Esslingen

im Wintersemester 2010/2011

Stand **08.02.2011**

Table of Contents

1 Project Overview....3

2 Design and Methodology....4

3 Protocol Specification....6

4 Example and Explanation....15

5 Usage and Patterns....16

6 Any Questions?....18

1 Project Overview

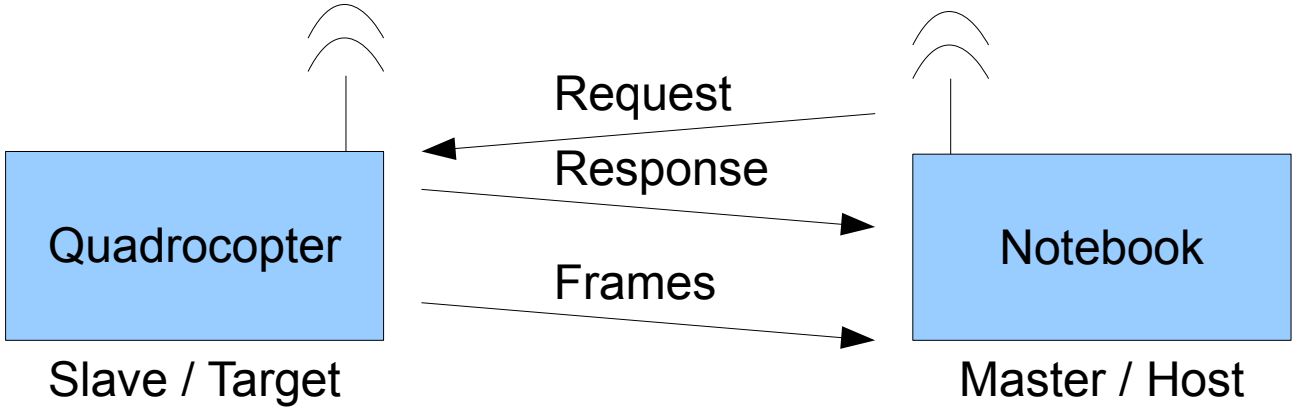

Figure 1: Project Overview

- Development of a Master-Slave protocol for the Quadrocopter
- The master requests information, the slave responds to it
- Based upon a radio-frequency communication technology called ZigBee
- Developed for remote monitoring, debugging and analyzing purposes
- Framing mechanism introduced to reduce traffic and protocol overhead

2 Design and Methodology

2.1 The V-Model

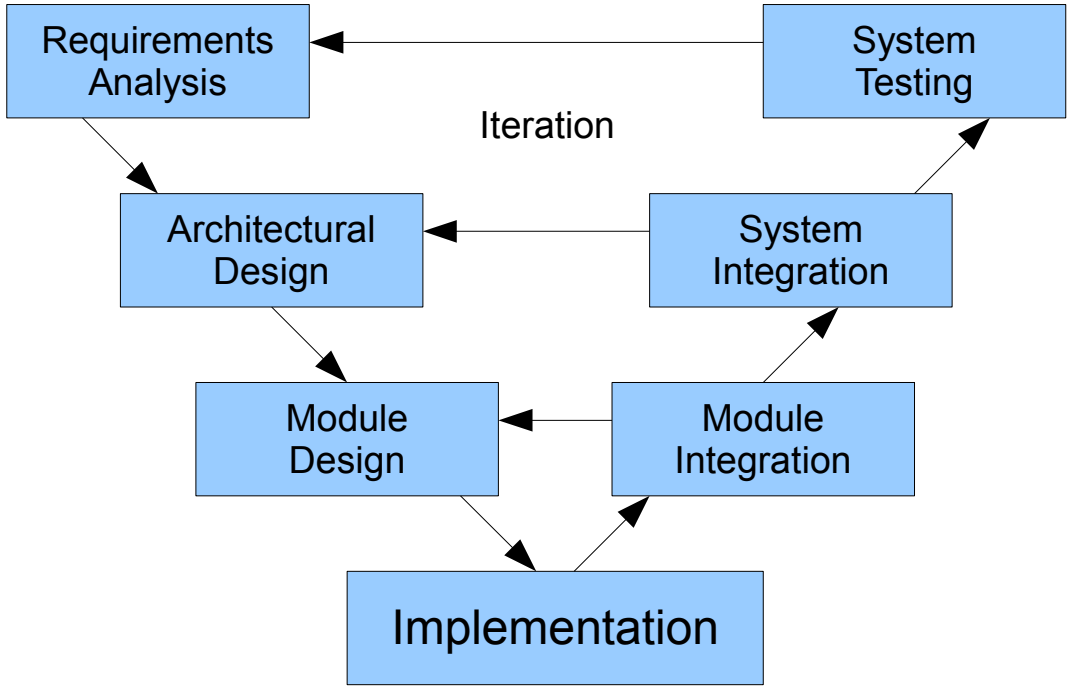

Figure 2: V-Model

- The process of the development was done keeping to the established V-Model

2.2 The ISO/OSI-Reference-Model

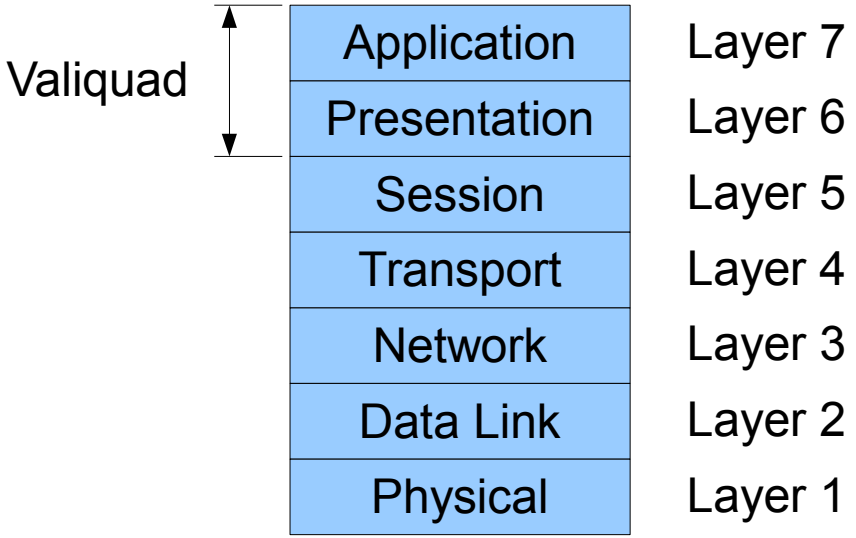

Figure 3: ISO/OSI-Model

- Layers 6 and 7 were part of the development within the Valiquad project
- The other layers may vary from application to application (currently using ZigBee)
- This approach guarantees flexibility and independence from the physics

3 Protocol Specification

3.1 Packet Structure

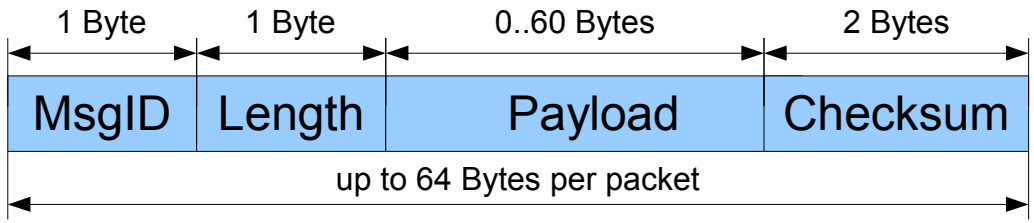

Figure 4: Packet structure

- All types of messages (request, response, frame) use the same basic packet structure
- The message ID identifies the type of a message, for example indicates the getting or setting of a parameter, or the configuration of the frames
- The check-sum is a 16-bit CRC value with a hamming-distance of 4
- The protocol overhead is 4 bytes, including the header and a trailing check-sum
- Up to 60 bytes of payload data can be transmitted per packet

3.2 Response Messages

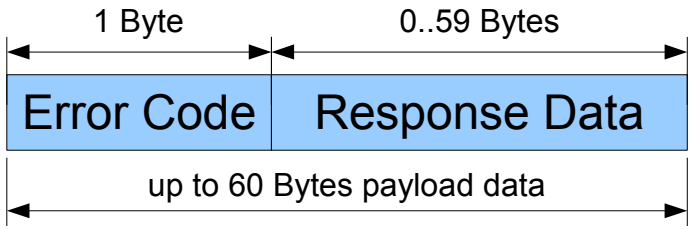

Figure 5: Response Message

- The slave will respond to a request from the master with a response message
- The response message has the same message ID as the according request, but it has an additional leading error code in the payload data indicating the status of the request
- The remaining payload data (up to 59 bytes) can hold additional data and information, such as the values for requested parameters

3.3 Parameter IDs (ParIDs)

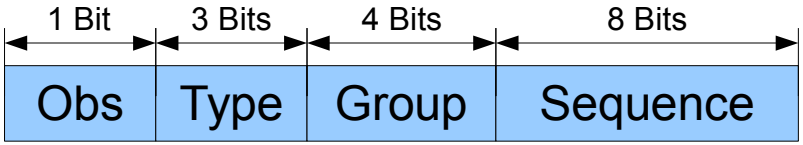

Figure 6: ParID Composition

- Parameter IDs ensure a dynamic and flexible approach for the handling of sampling values and parameters that are not known in advance and may join the system at any state of operation
- The group ID and the sequence number form a unique identification
- The observable-bit and the type ID encode additional meta-data for every parameter
- Observables are read-only sampling values that are sent periodically
- Normal (none-observable) parameters can be set and read by polling

3.4 Parameters → Polling (Request / Response)

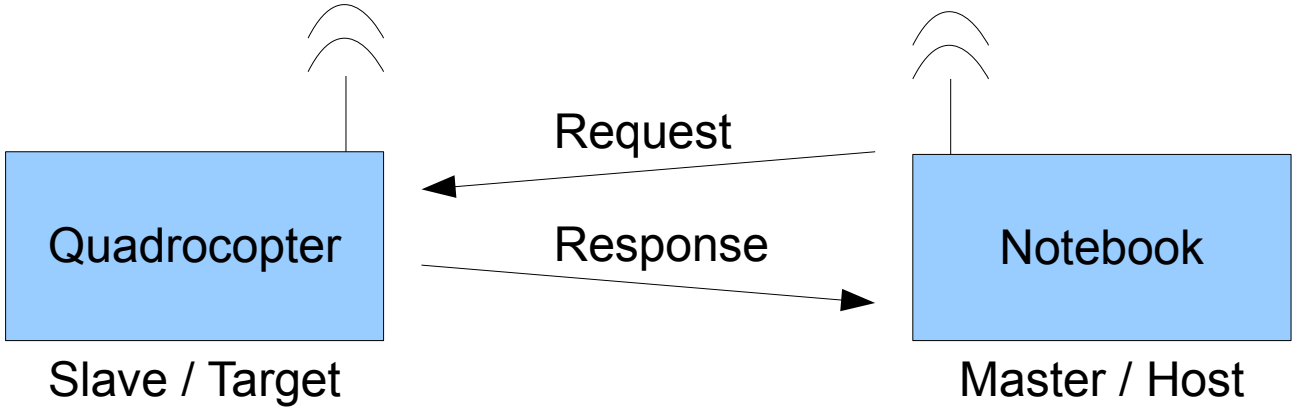

Figure 7: Project Overview

- Parameters are set and read by polling the slave, using the appropriate parameter IDs
- Up to 30 parameters can be set and read with one request/response-sequence

3.5 Observables → Sampling (Frames)

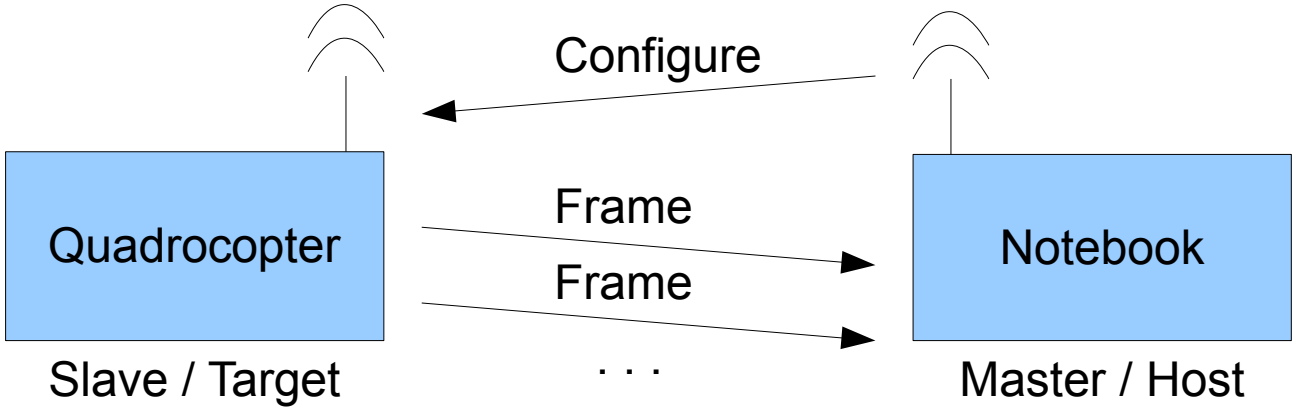

Figure 8: Project Overview

- Observables are configured once and sampled/sent from the slave to the master at a defined rate without the need for polling the actual value over and over again
- For every observable a different sample-rate can be set (low, medium, high)

3.6 Frames (Part 1)

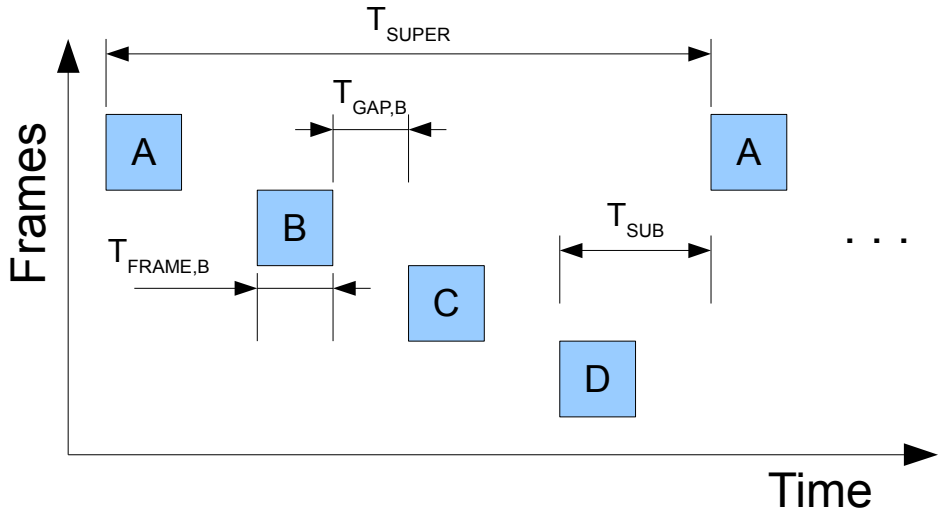

Figure 9: Frames over time.

- The central point of the protocol is the framing mechanism
- This approach is used to ensure a dynamic way of sampling values from sensors and other data sources without knowing their actual constitution in advance
- Four frames are sent one after another in a cyclic manner, they are named from A to D

3.7 Frames (Part 2)

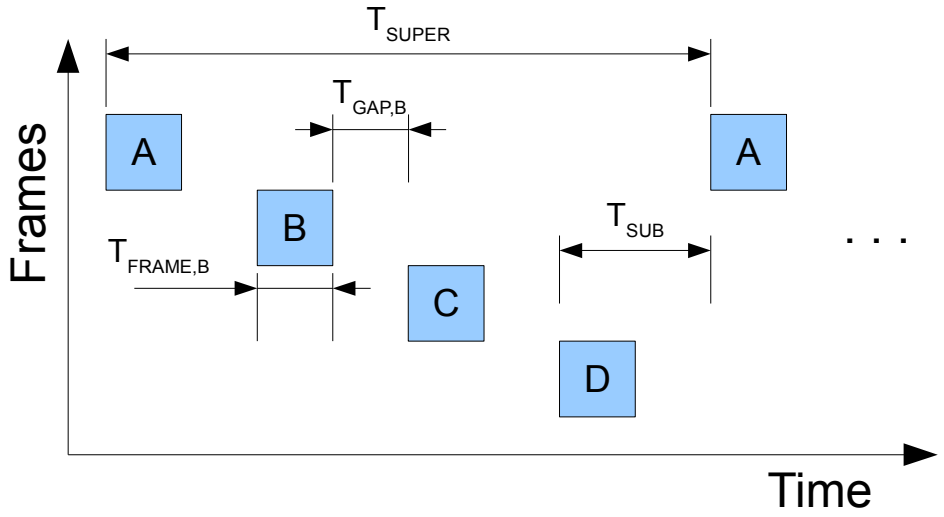

Figure 10: Frames over time.

- After a super-period the frames will be repeated, holding the updated sampling values
- Between the frames there must be an adequate inter-frame space called gap
- The transmission of a packet with its trailing gap is called a sub-period
- The super-/sub-period is dependent from the baud-rates and user settings

3.8 Frames (Part 3)

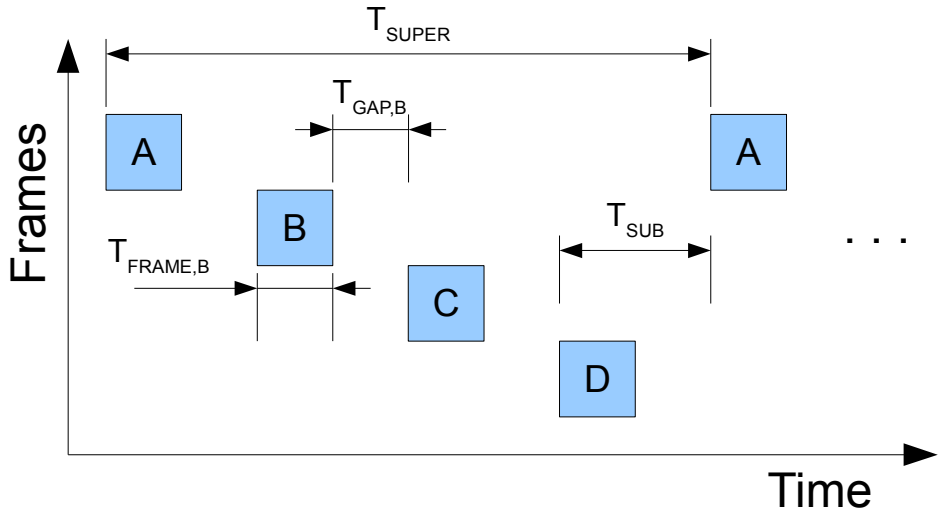

Figure 11: Frames over time.

- There are a few constraints on those parameters:
  - $T_{SUPER} = 4 \cdot T_{SUB}$
  - $T_{SUB} = T_{GAP,i} + T_{FRAME,i}$ *with* $i \in \{A, B, C, D\}$
  - $T_{GAP,i} \approx T_{FRAME,i}$ *with* $i \in \{A, B, C, D\}$

3.9 Frames (Part 4)

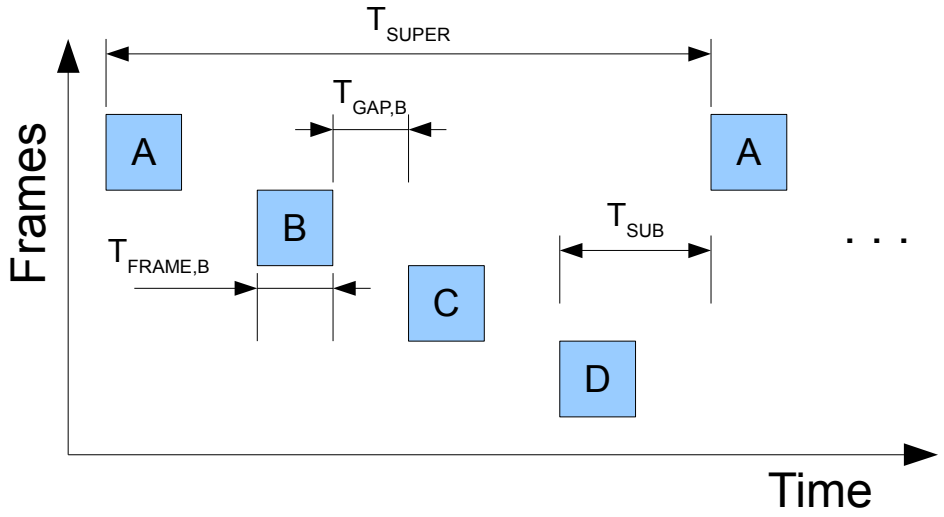

Figure 12: Frames over time.

- The highest possible sample-rate is T_{SUB} (sampling a value in A, B, C and D)
- The lowest possible sample-rate is T_{SUPER} (sampling a value in A, B, C, or D
- There is a third possible sample-rate in between (two times TSUB by sampling a value in the frames (A and C) or (B and D))

4 Example and Explanation

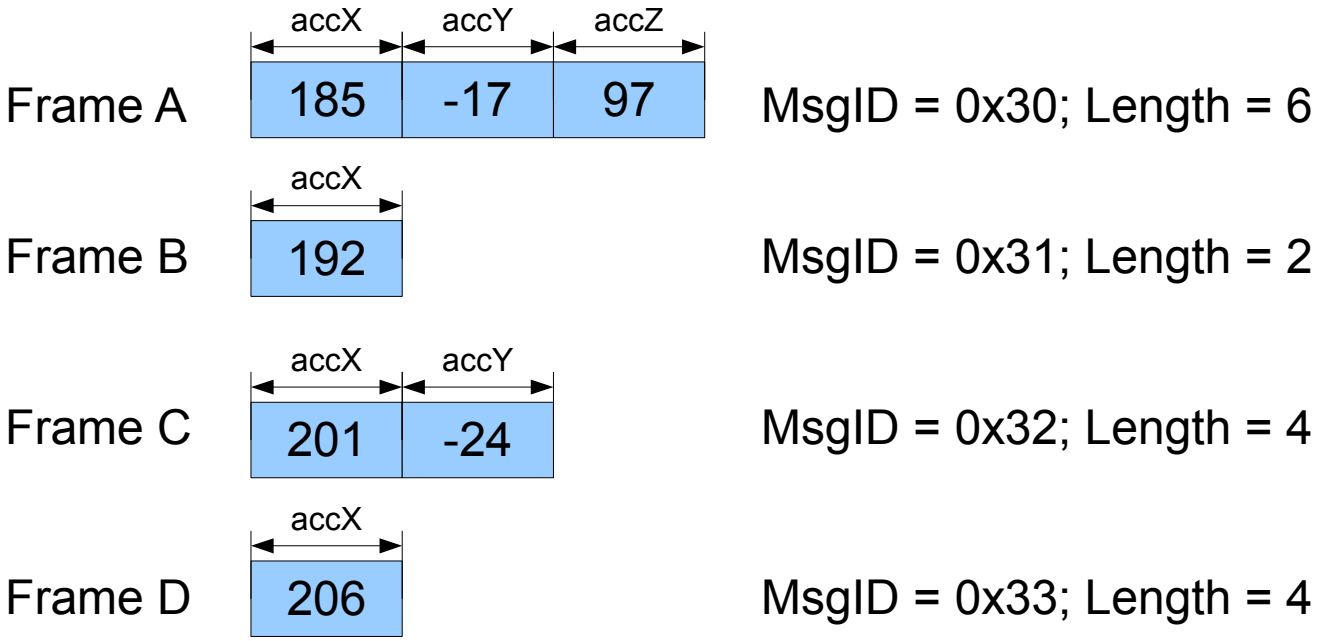

Figure 13: Frame Example

- T_{SUB} = 20ms (configured) $T_{accX} = T_{SUB}$ = 20ms (high)
- $T_{accY} = 2*T_{SUB}$ = 40ms (medium) $T_{accZ} = 4*T_{SUB}$ = 80ms (low)
- ParID(accX) = 0xD001 = $1101\ 0000\ 0000\ 0001_{(2)}$

5 Usage and Patterns

5.1 Middleware Technology

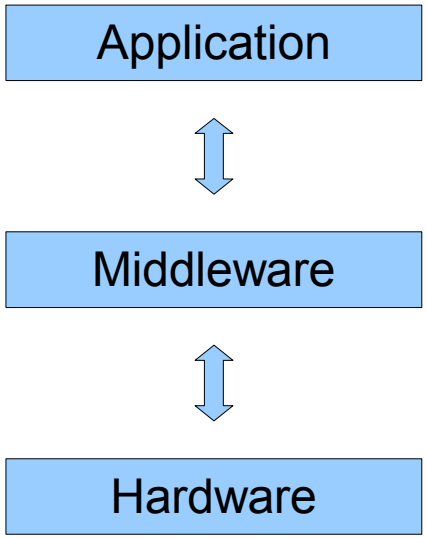

Figure 14: Middleware

- The Valiquad project is shipped as a hardware library enabling the communication between the host and the target in a dynamic and flexible way
- It is some kind of a middle-ware technology useful for several purposes

5.2 The Model-View-Controller Pattern

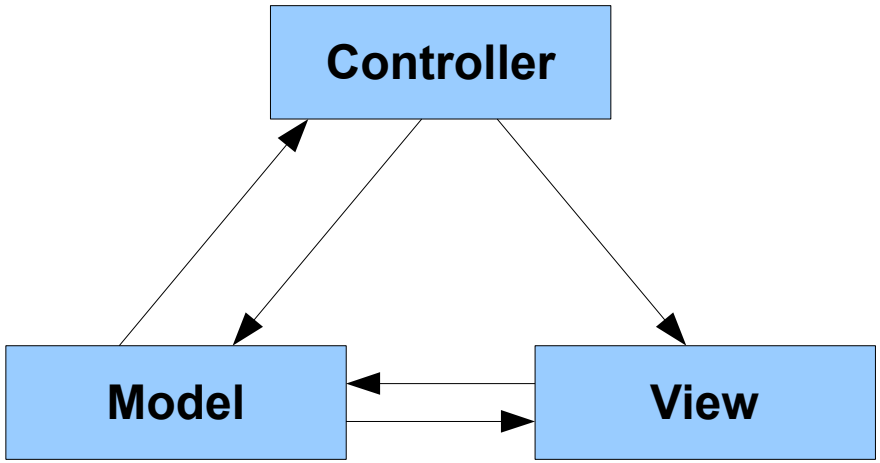

Figure 15: Model-View-Controller

- The Model-View-Controller pattern is one possibility on how to use the Valiquad hardware library within an application that is generating a graphical user interface
- In this scenario the Valiquad hardware library fulfills the data handling and storage of all relevant parameters and takes over the role of the “Model”
- But there might be many different approaches for integrating Valiquad to generate new applications and services for the software laying on top of it

6 Any Questions?

Vielen Dank für Ihre Aufmerksamkeit.

Sind noch Fragen offen?