# General problems with the Flight Controller

During the planning and implementation of the XCopter flight controller, the team encountered several problems that complicated the development of the flight controller. Most of those problems was the lack of experience and expertise.

The most general problem was, that most of the team members never dealt with UAVs or RC model making in general. Therefore nobody really knew what a flight controller is or how a flight controller works. This know-how of course is essentially important when developing an own flight controller. The problem was more or less solved by reading articles, tutorials and documentations of open-source solutions. Because the XCopter flight controller is implemented on an Altera DE1-SoC Board and the sensors and motors are not standardized, the code of most of the free and open-source solutions didn’t help a lot.

What also was a problem, was that no team member really knew, how to design and structure a software system with a medium complexity like the flight controller. Because of this, the designing phase took an above average time to finish. It was needed to change the design several times, what of course leads to regression in progress. When the design of the flight controller was done and the work was distributed to the team members, there was the problem with specifying the interfaces of the single components.

The interfaces were an issue because of the missing expertise in sensor, controlling and regulation technology. To connect the particular components of the flight controller, it is essentially to know how the data has to be interpreted and converted in the respective parts. The biggest problem was to convert all the sensor and remote control data, so that the ranges and units fits together. When the ranges or units don’t fit together, the PID regulator as well as the motor mapping cannot work properly. Especially when configuring the PIDs and the motor mapper, a know-how in regulation technology would have helped a lot. Because the error of distributing work without bringing out the interfaces was committed, the problem with the data ranges and units was realized at a very late state, it isn’t solved in the current version of the flight controller.

Because the development took place on the Altera DE1-SoC, a special version of the Eclipse IDE has to be used. In this special version, when a program for the DE1-SoC board has to be devolved, the IDE creates two projects. One project is where the own source code and files are added and the other project is a so-called BSP-project. The acronym BSP stands for board support package. This project is generated with the SoPC information file, which contains all the hardware addresses of the devices on the board. That BSP-Project is then linked to the actual (own) project to make the devices available there. To develop professional software, a version tracking system such as GIT or SVN is mandatory, so that the team members can share and merge their work easily. Normally you would just create a new GIT repository containing the software project and every team member would push and pull their work to or from the repository. The laboratory computers, which were used, had no pre-installed GIT so it had to be installed on them. The laboratory PCs are multi-user workstations and the GIT software that was used didn’t worked well. Whenever another user logs in to the PC, GIT gets reset and the repository have to be cloned again. Because of the special environment in the lab and the special IDE, GIT is not working properly, and the code cannot be built and tested all computers. This led us to use one computer as the main developing PC, where the code is built and loaded onto the DE1-SoC board. This of course aggravated the work in the team and in the whole project.