# Overview about embedded Linux for ARM

## Reasons for using a custom embedded Linux

There are many Linux distributions that are designed to specifically run on ARM devices. When thinking about what OS to run on the ARM part of the DE1-SoC system, the question arises if a pre-built OS, or a custom built OS should be chosen. Choosing a pre-built OS would have been the way of least resistance when considering the OS-question, especially as Altera has pre-configured OS-images on their website to download. A custom built OS on the other hand has several advantages:

* **Customizable**
  + The fact, that a custom built OS is more customizable, than a pre-built is self-evident and doesn’t need to be explained in more detail.
* **Light weight**
  + Light weight means, that the operating system can be stripped from any functions that are not needed for what the system is intended to do. Functions can be added for development, debugging and testing for a better experience while working with the OS. Added features can then be removed from the custom distribution in a later stage of the project when everything is set up correctly and the system is running stable. In the case of the X-Copter, the final OS can be stripped from anything that is not related to logging sensor data, communication with the ground station and eventually 3D mapping in the future.
* **Performance**
  + Light weightiness brings also the benefit of better performance. It is obvious, that a stripped down system is working much faster on its delegated tasks, when there are no unneeded services running in the background that use up CPU-power.
* **Security**
  + Security is an important point to consider when designing drones. With a stripped down OS there are less attack vectors for hackers and the system should be more secure overall.
* **Learning effects**
  + Custom built operating systems are widely used in modern embedded design and being able to work with such a system on the X-Copter project yields great opportunities to have hands on experience with an embedded OS.

The biggest disadvantage is the steep learning curve for working with custom embedded Linux distributions, especially for first timers. It is also a big challenge to get additional hardware like Wi-Fi-Sticks working as drivers sometimes have to be customized, or even written from scratch, to work with the system. Sometimes drivers and support for specific hardware is already implemented into a toolchain for building the embedded operating system (like Buildroot or Yocto). But even if drivers are present in the toolchain, finding them in convoluted menus and choosing the right dependencies (like the correct eeprom support etc…) is time consuming. For our project the Buildroot toolchain was chosen for generating, and configuring the real-time embedded operating system.   
A stable Kernel with version number 3.10-ltsi-rt (custom repository version) is used for the embedded Linux. It is not important to have the newest Kernel that is currently on the “Kernel-market”, but it is of importance to have a stable and supported version.