2021 | WHITE PAPER

PX4 AUTOPILOT ON A UAV CONTROLLER

Flying a Xilinx ZU7EV based processing board

Gijs van Esch | Embedded software developer

Dirk van den Heuvel | Embedded systems architect

TOPIC

CONTENTS

2
An introduction to autopiloting
PX4 Autopilot
Running PX4 on the platform
4
The UAV/Robotics platform
5
Reading the sensors
Motor control
Remote control
Using PX4
Actual flying
9
Conclusion
10
About TOPIC Embedded Systems

For more information go www.topic.nl or contact us via +31 (0)499 33 69 70 | info@topic.nl

AN INTRODUCTION TO AUTOPILOTING

Flying an Unmanned Aerial Vehicle (UAV), also referred to as a drone, introduces a typical functionality requirements for the software stack of the controlling platform. For this white paper a quadcopter UAV is taken as a reference to demonstrate autopilot functionality integration in Linux.

For other drone constructions like flying wings, helicopters, regular airplanes or other flying platforms similar requirements apply. When airborne, an UAV has to maintain a minimum level of functionality to stay reliably in the air. In case of our quadcopter reference design, it must remain in a stable position without active involvement of the operator. This is denoted as hovering: maintaining the same position in a 3 dimensional space. To be able to hover, the motors need to be controlled and sensors need to be read to determine position, movement and environmental context. This is exactly the functionality an autopilot should implement.

In this paper an integrated platform is presented that implements the PX4 autopilot as part of an experimental quadcopter drone concept. The heart of the system is the controller board, formed by a compact but powerful and highly integrated processor board, running Linux. The paper will address dependencies and implementation considerations with respect to autopilot functionality.

PX4 AUTOPILOT

The PX4 autopilot is selected from a collection of Open Source autopilot projects, such as Paprazzi UAV, ArduPilot, Dronecode and LibrePilot. The Dronecode project forms a platform where PX4 is part of. Based on the initiatives around Dronecode, sponsoring by the Linux Foundations and community focus, PX4 was chosen. PX4 was originally developed at the ETH in Zurich, but has gained a world-wide support base. It provides a flexible set of tools for drone developers to share technologies to created tailored solutions. It is supported by more than 300 global contributors and is used by various well-known companies. As PX4 is part of the Dronecode project, it can seamlessly integrate with other tools from this project, such as a communications protocol for UAV systems (MAVLink) and a control station (QGroundControl).

RUNNING PX4 ON THE PLATFORM

The aim of the this effort is to implement the PX4 software stack on a Linux based processing platform, the Xilinx Ultrascale+. This System-on-Chip (SoC) integrates a quad core ARM Cortex A53 application processor, a dual core ARM R5 real-time processor, an ARM Mali400 GPU and FPGA based free programmable logic. The aim is to run the autopilot on the A53 core to investigate the real-time requirements, the performance behavior, as well as the system integration complexity. Choosing the real-time processor would be more obvious, but the additional software functionality on top of the autopilot, like path finding, obstacle avoidance will also run on Linux based application processor. Integration within the same context simplifies application development significantly.

The platform runs a Linux distribution on the A53 processor, kernel version 4.19. The build environment used is the Yocto project in combination with OpenEmbedded. A specific build recipe was to be developed for PX4 in order to incorporate the software stack on the Linux platform. These Yocto recipes should be created in such a way that they point to the source files or a personal github clone of PX4. The PX4 source files had to be altered in such a way the PX4 Github is not accessed directly, but forked to a local branch. This recipe is responsible for compiling the PX4 software and copying the correct files (binary, ROMFS and posix-configs) to this image. When this is completed correctly a PX4 executable of the image is available which enables PX4 to run on the platform.

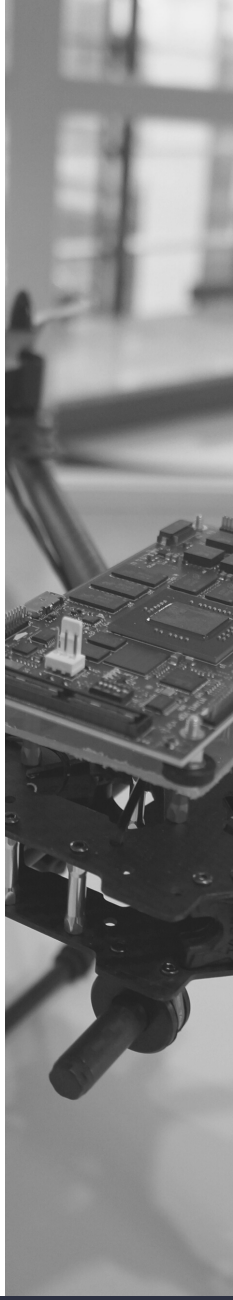
The Yocto recipes solve the software dependencies for running PX4 on Linux, using the abstracted sensors readings by the Linux drivers. This way a bridge is created between the physical sensor readings and the sensor data format required by PX4 with a sample rate compatible with the execution time of the PX4 control loop.

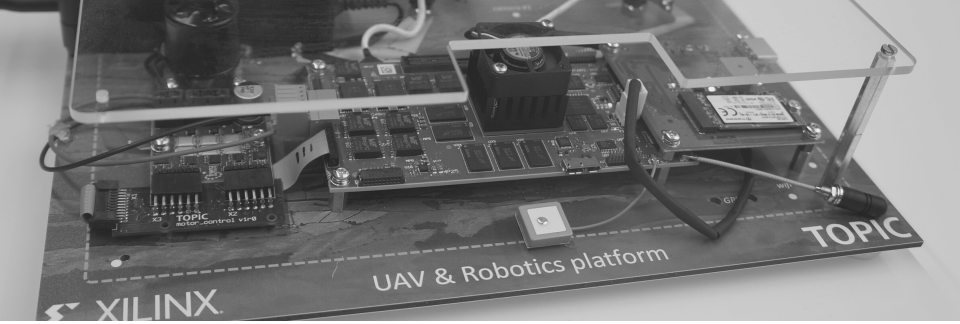
THE UAV & ROBOTICS PLATFORM

An autopilot like PX4 needs specific environmental, motion sensors and user interfaces as well as motor drives for operating the drone in a reliable manner as an output. Typical drone motors are brushless DC (BLDC) motors. The selected Xilinx UAV & Robotics platform, physically the size of an iPhone, has all the hardware integrated that is required to fly autonomously:

- The Inertial Measurement Unit is a Bosch BMI088 which is a 6-axis motion (3-axis accelerometer and 3-axis gyroscope) tracking sensor.
- The applied low-power and low-noise Bosch BMM150 is a 3-axis digital geomagnetic field sensor.
- For environmental sensing, the Bosch BME680 is used. It has the capability to measure the barometric pressure which can be used for the altitude estimation as well as the environment temperature and humidity.
- The low-power ZOE-M8B module uses the Global Navigation Satellite System (BeiDou, GLONASS, GPS / QZSS) to receive positional information.

The Xilinx UAV & Robotics platform integrates a System-on-Chip of Xilinx. This Zynq Ultrascale+ ZU7EV is a heterogeneous multi-processor core. It allows the integration of all required processing on a single device. In addition to the autopilot, the low-level brushless DC motor controller, the video pipeline for e.g. stereo vision and object recognition, an integrated H264/265 video compression block, dedicated processing power for path finding/plotting, software defined radio as well as a machine learning (ML) or deep learning (DL) are available on the platform. This reduces the need to implement additional boards in the system and reduces the power consumption significantly. However, the basic functionality of a drone is flying, requiring a motor driver and controller as well as an autopilot to keep the drone reliably airborne.





READING THE SENSORS

Based on the mentioned sensors on the board, the PX4 autopilot is able to control the brushless DC motor drivers. The control loop takes the sensor readings with deterministic intervals and derives from these readings the updated motor/motion control settings.

The sensors are connected to the processing system using SPI and I2C interfaces via the FPGA, allowing for relative high data acquisition rates. Especially for stability purposes, this is advantageous. The acquisition of sensor data is implemented on the platform using the Linux industrial IO subsystem and can be read either via libiio or file io. The decision was made to use libiio as this is faster and more flexible than the file io concept. To get libiio operational, it is important to include libiio as a dependency in the build recipe for PX4.

The sensor data should be published within PX4. There are some examples available within PX4 reference designs that show the overall structure. The drivers for the sensors are already implemented as part of the standard board BSP. The main change is that the data has to be read via libiio and scaled from the raw sensor data values into real, meaningful data. The calculations to do this can be found in the data sheet of the sensors in combination with the PX4 data format requirements. For details on PX4, have a look at https://px4.io/**.**

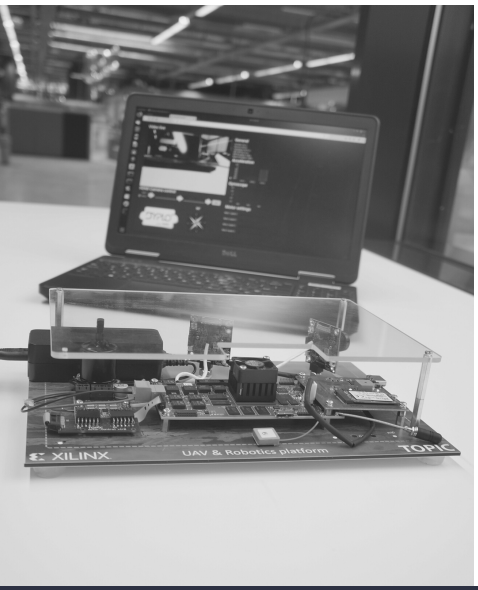
For the accelerometer and gyrometer the data is published within PX4 using the px4_accelerometer and px4_gyrometer parameters. This ensures that the necessary mathematical calculations, like the derivative and integral are performed correctly. The data of the environmental sensor and magnetometer can be published directly.

The applied uBlox GPS module is supported by PX4 by default and only has to be enabled in order to be read. This can be done by enabling the GPS in de sensor startup script, part of the standard Linux driver. When all the sensors are correctly implemented and started within PX4 the data should be published on the parameters sensor_gyro, sensor_accel, sensor_mag, sensor_baro and vehicle_gps_position. The data can be verified using the PX4 listener functionality.

MOTOR CONTROL

The motor controllers are implemented in FPGA fabric. This allows full parallel operation of the four motor current loops and related velocity loops. A brushless DC motor controller IP block of Qdesys is applied to implement the back-EMF based low-level motion control. The outputs of the FPGA are used to directly drive the power stages of the motors. No additional control logic is required. For the drone, standard power stage driver modules are applied. The analogues motor feedback signals are sampled, digitized and acquired by the FPGA logic via an I2C bus.

The motors are controlled via the linux_pwm_out module of PX4. In order to do so this output has to be altered in such a way that the data format is compatible with the Qdesys motor control driver.

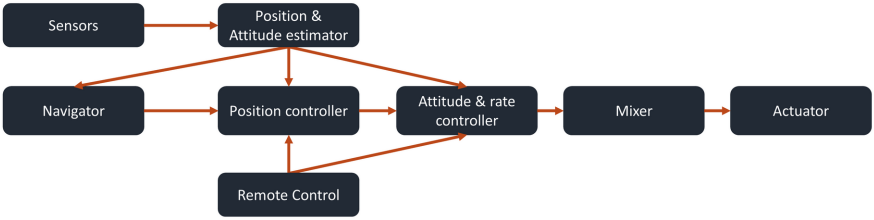


REMOTE CONTROL

The last interface required is the user interface via the remote control functionality. The drone implementation can accept different radio communication interfaces, including software defined radio. In the implemented application the interface runs over a WiFi connection. The drone platform forms a WiFi access point. By opening a secure shell (SSH) connection, the drone can be controlled via terminal commands from a PC, notebook, or mobile device.

USING PX4

With the interface drivers in place the platform is ready to accept the PX4 implemented control functionality. It is important to setup PX4 the correct way, matching the system requirements as initially set out. The image below illustrates the functional flow of the typical PX4 control loop.



- The data from the different sensors are sent to the sensors module, which will combine the data from the different sensors in a single message.
- This single message is then sent to the Position and Attitude estimator which can be either EKF2 (Extended Kalmann Filter 2) or LPE (Local Position Estimator). For autonomous flying the LPE offers the best performance. This consists for the LPE and attitude_estimator_q.
- The resulting data is then sent to the Navigator, position controller and finally the attitude and rate controller.
- The mixer in this case will be incorporated in the linux_pwm_out which will directly control the motors.

The PX4 autopilot engine has been executed using a standard embedded Linux kernel (4.19). One of the PX4 porting objectives was to determine how timing critical the real-time behavior of PX4 is. The application runs without the real-time patch of Linux applied. As all the timing critical interfaces are handled by the inherently real-time FPGA fabric, the PX4 autopilot code only needs to run the control loop in real-time. The raw sensor readings are acquired individually using the FPGA fabric and then down-sampled to the unified sample rate. This dictates the heartbeat of the PX4 control loop.

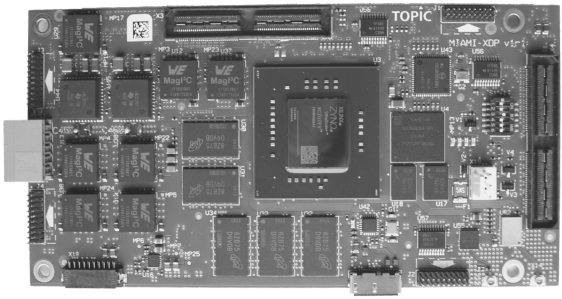
ACTUAL FLYING

The autopilot is the core of the drone to be able to fly. To put this to the test, a drone is constructed based on a:

- Tarot RC Iron Man 650 quad copter frame
- 4x PMOD motor power stages (100W minimum each)
- The Xilinx UAV & Robotics platform
- 3S LiPo battery pack with sufficient capacity

The drone is controlled by sending positional setpoint commands to PX4 via a SSH terminal connection over WiFi. Based on the sensor readings PX4 takes care that the displacement of the drone from one positional setpoint to another is executed.

As expected, the initial flight was not perfect. This had to do with tuning of the BLDC motor controller settings and the system PID control loop settings. Calculated parameters needed to be tuned, depending on the actual system behavior. Especially the tuning of the roll, pitch and yaw parameters of the PID control loop (MC_ROLLRATE_P, MC_ROLLRATE_I, MC_PITCHRATE_P etc.) needed attention. The testing of the PX4 autopilot on top of an embedded Linux platform is demonstrating sufficient quality and potential to apply the result in an industrial application context.



CONCLUSION

This white paper described the successful porting and testing of the PX4 autopilot software stack integrated using embedded Linux on a dedicated drone platform. More in-depth technical implementation details are available from TOPIC.

Creating a Yocto/OpenEmbedded build recipe for Linux integration needed specialized know-how, but was realized with limited effort. Publishing on-the-fly sensor data for PX4 within Linux is a relatively straight-forward operation. Running PX4 on a drone in practice rapidly provided the required results. The require CPU performance to execute PX4 is limited, leaving sufficient headroom to execute other applications. Therefore, the implementation of PX4 allows the addition of other software stacks to the system for e.g. path planning, object detection and payload control applications without expecting performance issues.

Further information on the UAV & Robotics platform can be found at https://topic.nl/en/products/building-blocks/autonomous-control-robotics.

ABOUT TOPIC EMBEDDED SYSTEMS

“We make the world a little better, healthier and smarter every day”. Our mission statement reflects exactly what we do: developing innovative systems for our customers. The way we do that, is by combining our customers domain specific know-how with our expertise in hardware and software development. This results in the most optimal product for our customers. TOPIC has a strong background of more than 24 years in developing systems, which can contain embedded-, application- and cloud software, FPGA code and PCB designs. We help customers in different domains such as medical, imaging, machine control & safety. With over 150 employees, we are a strong and established company with our headquarters in Best, the Netherlands. TOPIC has an ISO13485 (medical) certified Quality Management System and adopted the Agile way-of-working for optimal interaction with the customer.

Premier Partnership with Xilinx | TOPIC is one of the few Xilinx Premier Alliance Partners in the world and the only one in the Benelux. Our partnership with Xilinx started in 2009 and since than we have been working closely together over the last years.

Materiaalweg 4, 5681 RJ Best, The Netherlands

+31 (0)499 33 69 79

info@topic.nl

www.topic.nl

linkedin.com/company/topic-embedded-systems