#PX4DevSummit

PX4 autopilot DEVELOPER SUMMIT 2023

# PX4 on the NXP IMXRT1176 1 GHz MCU

Peter van der Perk
NXP Semiconductors

Dronecode FOUNDATION THE LINUX FOUNDATION

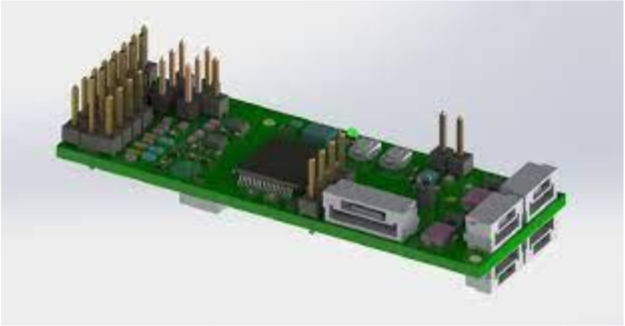
# Introduction

- Peter van der Perk - NXP Semiconductors, CTO Systems Innovations
- ADACORSA Project
  - Safe modular mobile robotics architecture system-concept
- Mobile Robotics team
  - Multinational team: Netherlands, Germany, USA, Canada, …
  - Reference designs and development platforms for autonomous rovers and drones
    - Vehicle Management Units
    - CAN/Ethernet Nodes & Hubs
    - Battery Management Systems
    - Most systems run PX4 & NuttX!
  - HoverGames & NXP Cup

# Background NXP & PX4

- RDDRONE-FMUK66
  - PX4 FMU Platform for NXP Hovergames drone
- UCANS32K146
  - PX4/NuttX Enabled reference design for CAN-based peripherals
- MR-CANHUBK344
  - ASIL-D ready lockstep FMU/Gateway running PX4

RDDRONE-FMUK66



UCANS32K146

MR-CANHUBK344

# Pixhawk Open Hardware Standard

- Coordinated by the Dronecode Foundation
- DS-012 Latest Autopilot V6X Standard
- DS-010 Pixhawk Autopilot Bus (PAB) Standard
  - DS-011 V5X and DS-012 V6X use the same PAB

pixhawk®

**DS-010**
**Pixhawk Autopilot Bus Standard**

**Revision**: 0.6.0
**Revision date**: July 29, 2021

**Abstract**
This document is the formal version of the Pixhawk industry standard that includes all aspects of the hardware standard required to build compatible autopilots.

pixhawk®

**DS-012**
**Pixhawk Autopilot v6X Standard**

**Revision**: 0.4.0
**Revision date**: December 08, 2022

**Abstract**
This document is the formal version of the Pixhawk industry standard that includes all aspects of the hardware standard required to build compatible autopilots.

Available at:
https://github.com/pixhawk/Pixhawk-Standards

# NXP based Pixhawk FMUv6X-RT

pixhawk

- FMUv6X-RT
  - Dronecode open hardware standard
  - Open-source schematics / hardware design
- Extension to Autopilot V6X
  - Utilizing the i.MXRT1176 MCU
- Implementing the Pixhawk Autopilot Bus (PAB) Standard
  - Goal: Drop-in replacement
  - In theory compatible with all existing PAB baseboards
  - Dronecode helps us to verify on all existing baseboards

MR-VMU-RT1170

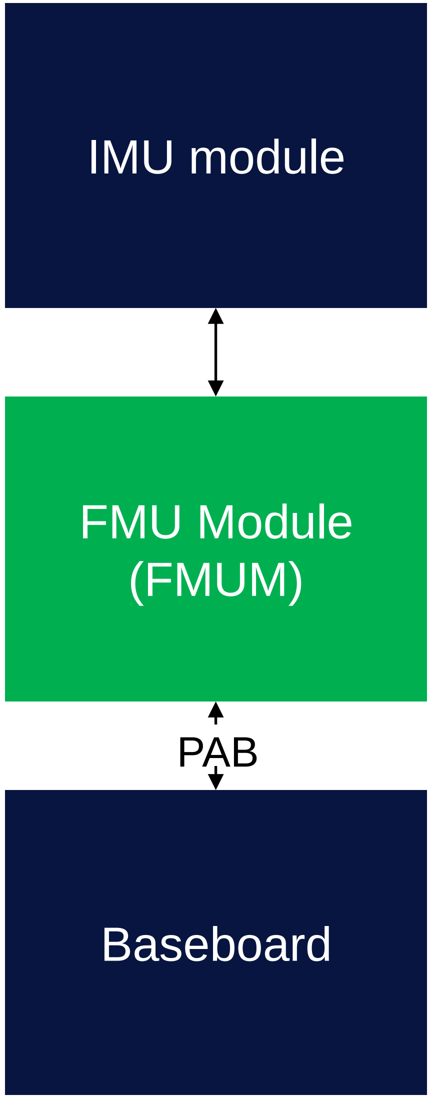
# Requirements for an V6X compatible FMU

- i.MX1176
  - Cortex-M7 @ 1GHz
  - Cortex-M4 @ 400MHz
  - 2MB SRAM
  - External XIP Flash
- Check requirements imposed by V6X standard

| | Required by V5X/V6X | i.MX RT1176 |
|---|---|---|
| Pack. | | LFBGA289 |
| ADC | 9 inputs | 20 inputs total |
| CAN | 2x CAN 2.0 (v5X)<br>2x CAN FD (v6X) | 3x CAN FD |
| ETH | 1x 10/100 Mbps | 1x 1 Gbps w/ AVB<br>(1x 1 Gbps w/ TSN)<br>1x 10/100 Mbps |
| I2C | 4x | 6x |
| SDIO | 1x | 2x |
| SPI | 6x | 7x *(FRAM on FlexSPI)* |
| UART | 5x + 3x w/ HS | 12x |
| USB | 1x | 2x HS |
| CSI | - | 1x MIPI-CSI Camera |

# Quick introduction to the Autopilot V6X Standard

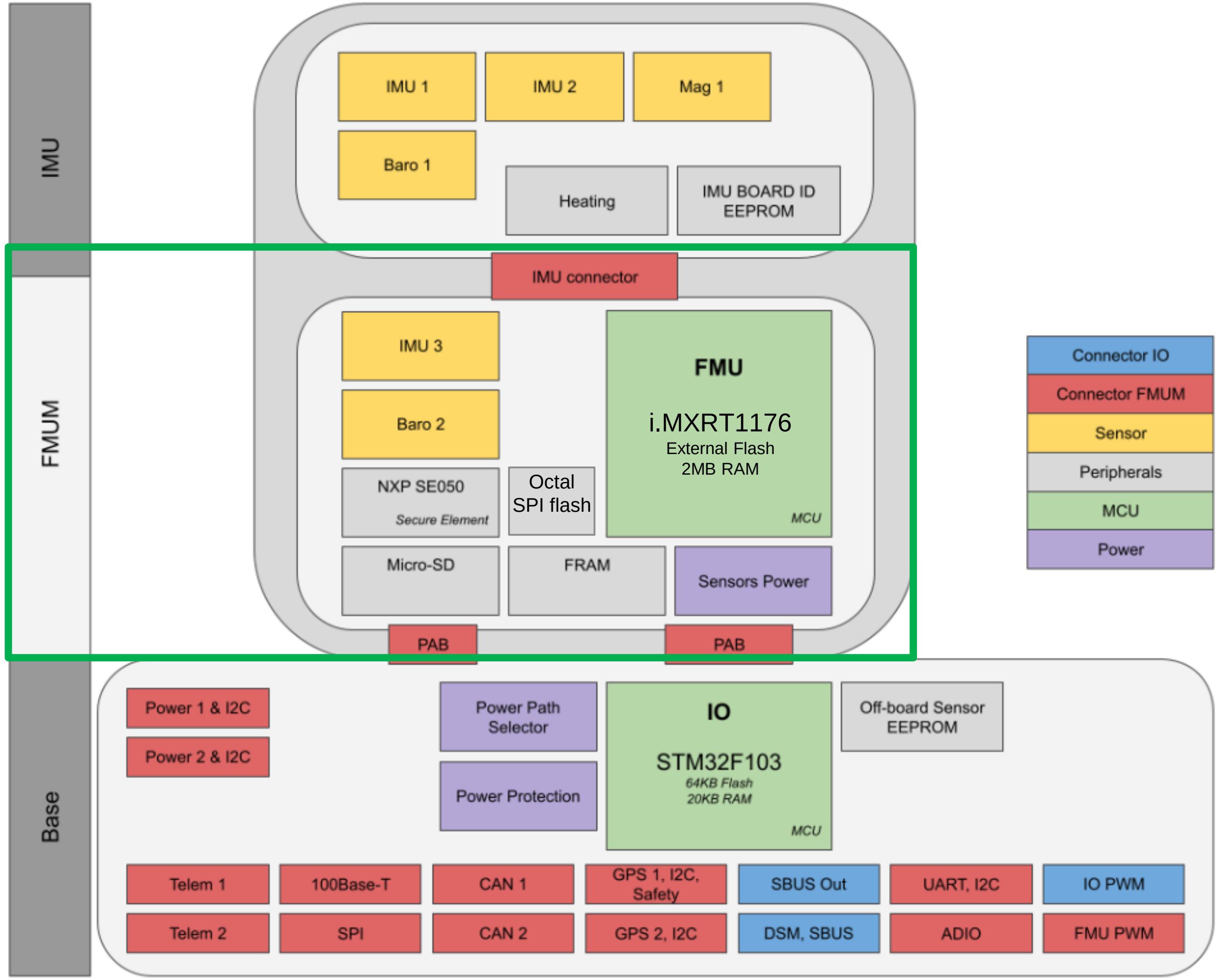
- The Autopilot V6X decomposes the flight controller in 3 boards:
  - IMU board
  - FMU Module (FMUM)
  - Baseboard
- Focus on this talk is the FMUM module
  - Connected to an FMUvxX IMU using a pinout described in DS-012
  - Connected to a baseboard following the PAB standard
- Pixhawk Autopilot Bus (PAB)
  - 100-pos Hirose + 50-pos Hirose connectors
  - Allows for custom baseboard for custom vehicles



# Pixhawk FMUv6X-RT

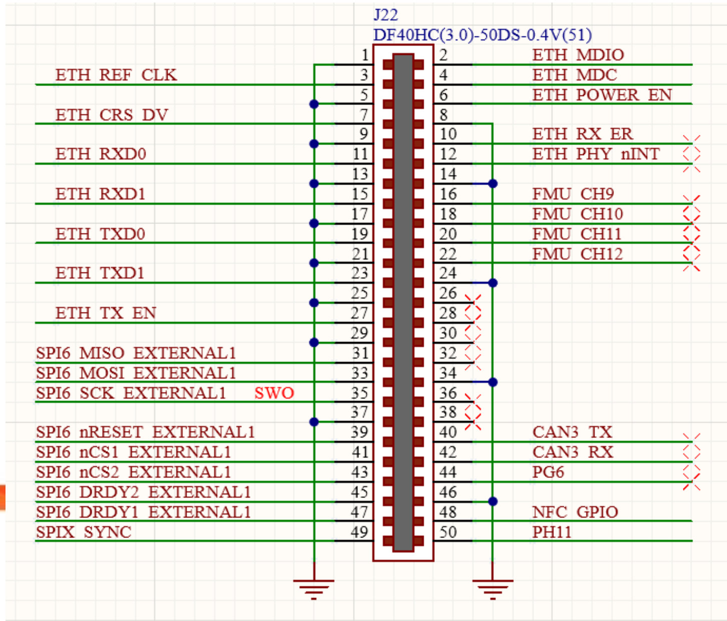
- FMU-V6X Compatible
  - DS-012 Pixhawk Autopilot V6X Standard
  - Specifies base sensor set
- i.MXRT based FMUM
  - Compatible with IMU connector
  - Pixhawk Autopilot Bus (PAB)

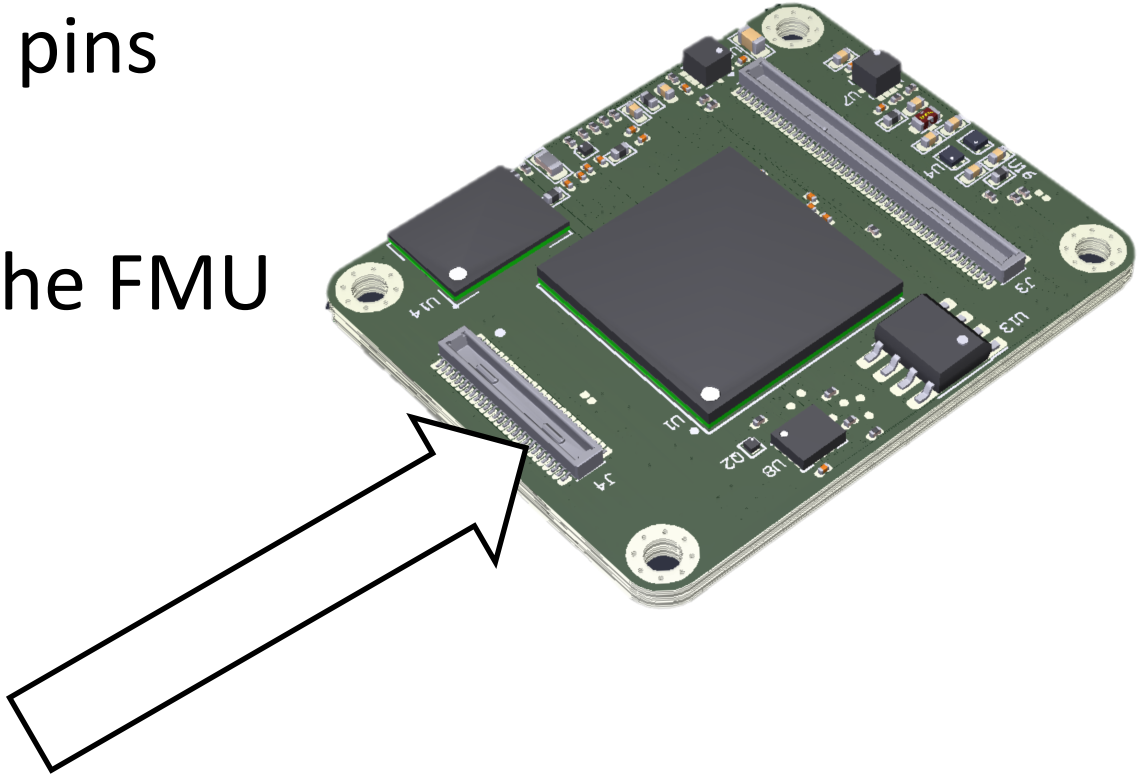
Overview



# FMUv6X-RT PAB X2 Spare pins

- FMUv6X-RT utilizes some spare pins on the PAB X2 port
  - Ethernet RX_ER and PHY_INT pins
  - CAN3
  - 4 Extra PWM channels from the FMU





# FMUv6X-RT FMUM Module

- MCU i.MXRT1176
  - Cortex-M7 @ 1GHz
  - Cortex-M4 @ 400MHz
  - 2MB SRAM
  - 64MB Octal SPI Flash
- SDHC interface
- 12-Channel PWM
- 8x UART
- 4x I2C
- 6x SPI
- 3x CAN-FD
- RMII 100M Ethernet
- USB
- Open-Source Software enablement
  - PX4 Autopilot
  - NuttX RTOS
  - Zephyr RTOS

# DS-010 Pixhawk Autopilot Bus (PAB)

- System-on-Module (SOM) Designs
  - Well documented
  - Reference circuits
  - Rapid prototyping/development
- Open points
  - For the i.MXRT1170 series the 100-pos Hirose + 50-pos Hirose connectors limits the potential of the SoC
  - Any interest for a new PAB?
    - Camera CSI interface
    - 1Gbit Ethernet / 2nd Ethernet / 2-wire Ethernet
    - I²S Audio / DAC
    - Wi-Fi + BLE

# Porting PX4/NuttX to the i.MX1170

NuttX RTOS

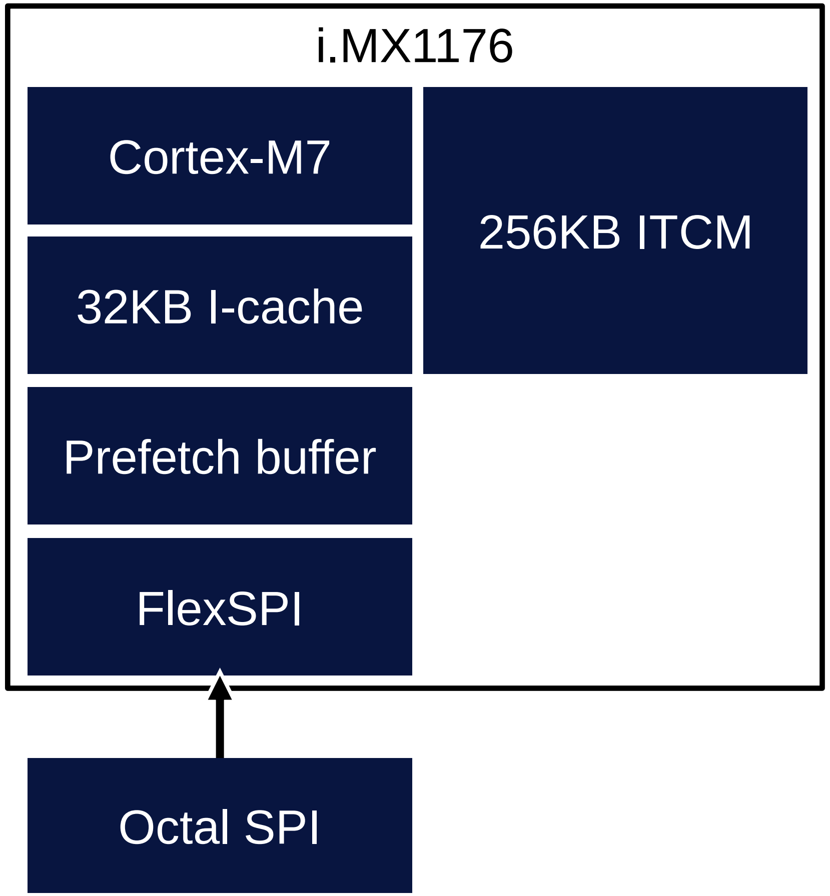
- PX4 runs on top of the NuttX RTOS
  - i.MX Family 1020/1050/1060 already supported by NuttX but needed to be extended for 1170.
- PX4 itself also need specialized drivers for
  - ADC
  - High resolution timer (HRT)
  - GPIO IRQ
  - PWM
  - Tone Alarm / Buzzer
  - Hardfault logging

# Optimizing PX4 for the i.MX1170

- NXP i.MX RT is *flashless* this means that an external flash must be used to store the program and executed on that.
- On the FMUM design we decided to use an Octal flash chip that runs @ 200MHz DDR
- The Cortex-M7 runs @ 1Ghz how do we maximize performance?
  - Run from RAM? *Not possible PX4 binary itself is roughly 2MB which consumes all available memory*
  - Run from flash, execute in place XIP? Works but is slower
  - **Hybrid approach?**

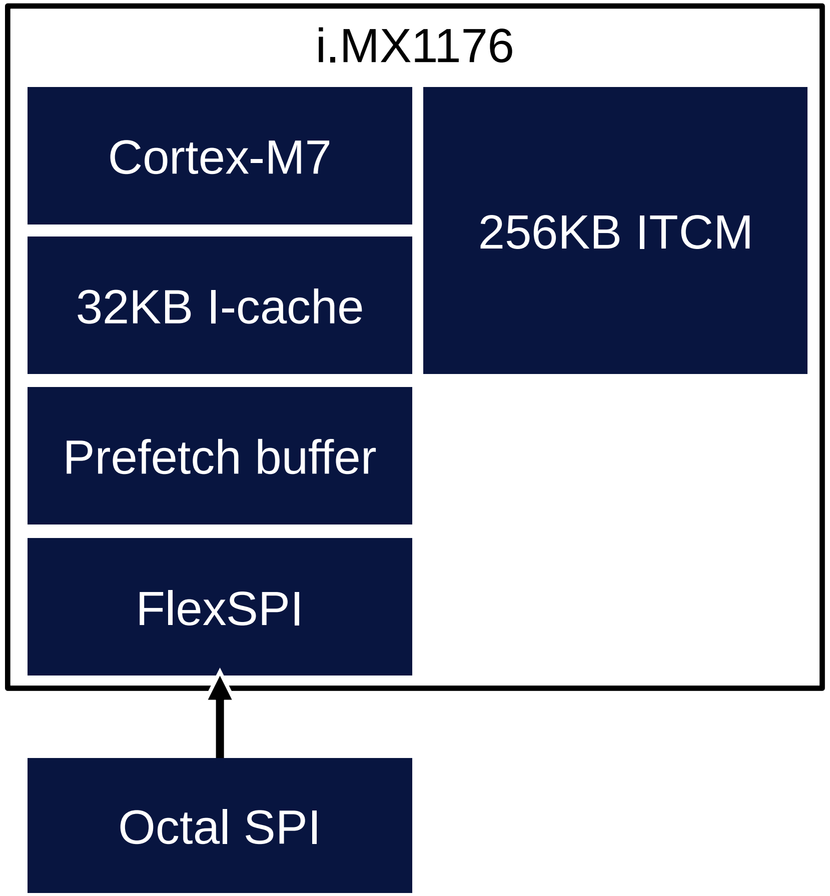
# Optimizing PX4 for the i.MX1170

- The Cortex-M7 core complex comes with 32KB instruction cache
  - Temporal Locality
- FlexSPI Flash controller
  Prefetcher that continuously fetches the next byte
  - Spatial Locality
- Instruction Tightly-Coupled Memory (ITCM)
  - Low-latency memory
  - Doesn’t get cached
  - But you’ve to manually specify map the code



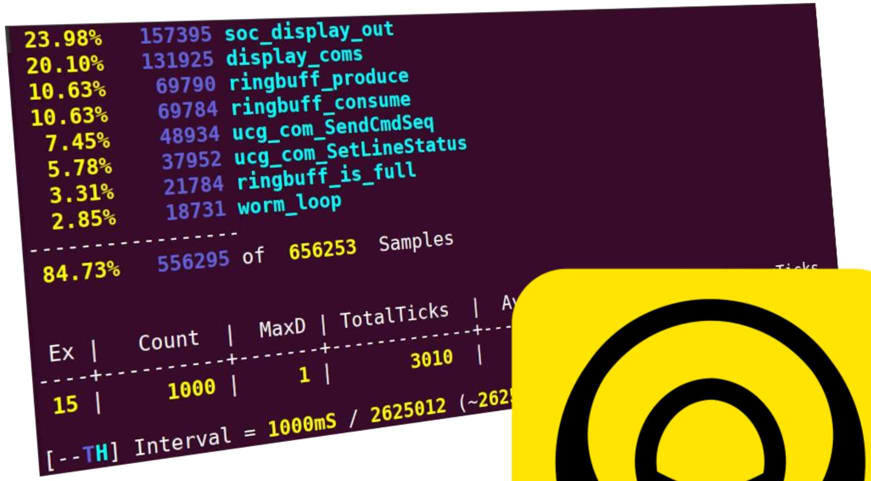
# Optimizing PX4 for the i.MX1170

- Instruction Tightly-Coupled Memory (ITCM)
- How to determine which function should be in the ITCM?
  - Put functions interrupt the spatial execution flow
    Interrupt Service Routines (ISR), this improves spatial locality and thus the prefetcher performs better
- How about the other functions?
  - Profiling!



# Profiling PX4 on the i.MX1170

- Orbuculum
  - Open-source – BSD-3
  - Implements ARM DWT, ETM, ITM trace
  - We've used orbtop tool to log MCU utilization
    - Log to a json file
  - Generate linker script to map the ITCM
- For more information about tracing and orbuculum checkout:
  - [https://orbcode.org](https://orbcode.org)
  - PX4 Developer Summit
    Sunday, October 22
    4:40pm - 5:10pm
    Debugging PX4 - Niklas Hauser

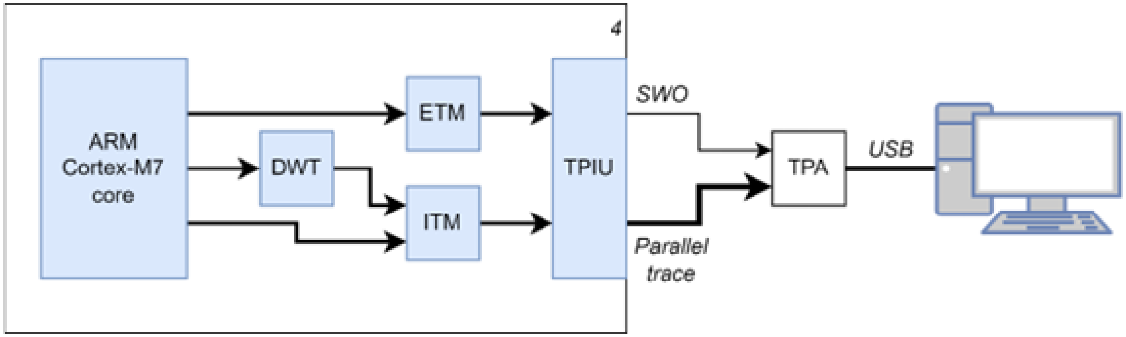


```
23.98%   157395 soc_display_out
20.10%   131925 display_coms
10.63%    69790 ringbuff_produce
10.63%    69784 ringbuff_consume
 7.45%    48934 ucg_com_SendCmdSeq
 5.78%    37952 ucg_com_SetLineStatus
 3.31%    21784 ringbuff_is_full
 2.85%    18731 worm_loop
-----------------
84.73%   556295 of  656253  Samples

                                        Ticks
 Ex |   Count  |  MaxD | TotalTicks  |  A
----+----------+-------+-------------+---
 15 |     1000 |     1 |        3010 |

[--TH] Interval = 1000mS / 2625012 (~262
```

orbcode
better embedded



https://github.com/orbcode/orbuculum

# Profiling PX4 on the i.MX1170 Test flight

- CPU Load ~ 30%
- Ram usage ~ 25%

- Flight test time
  ~ 8.5 Hours
  ~ 40 flights

# FMUv6X-RT Timeline

- Publication of the Dronecode FMUv6X-RT Standard
- Upstream PX4 FMUv6X-RT support
- FMUv6X-RT FMUM Module
  - Developer edition will be released
- Future work
  - Pixhawk baseboard with 100BASE-T1 and without an PX4IO processor

# Thank you

**Peter van der Perk**
Embedded Software Engineer
NXP Semiconductors b.v.

Peter.vanderperk@nxp.com