HAN'S ROBOT
WE SERVE HUMANITY

smartPORT
TUHH
Technische Universität Hamburg

Development of an Embedded Communication Hub for the Acquisition of Sensor Data in a Robotic System

Research Project and Seminar

Juan Carlos Reyes Andrade, ICS

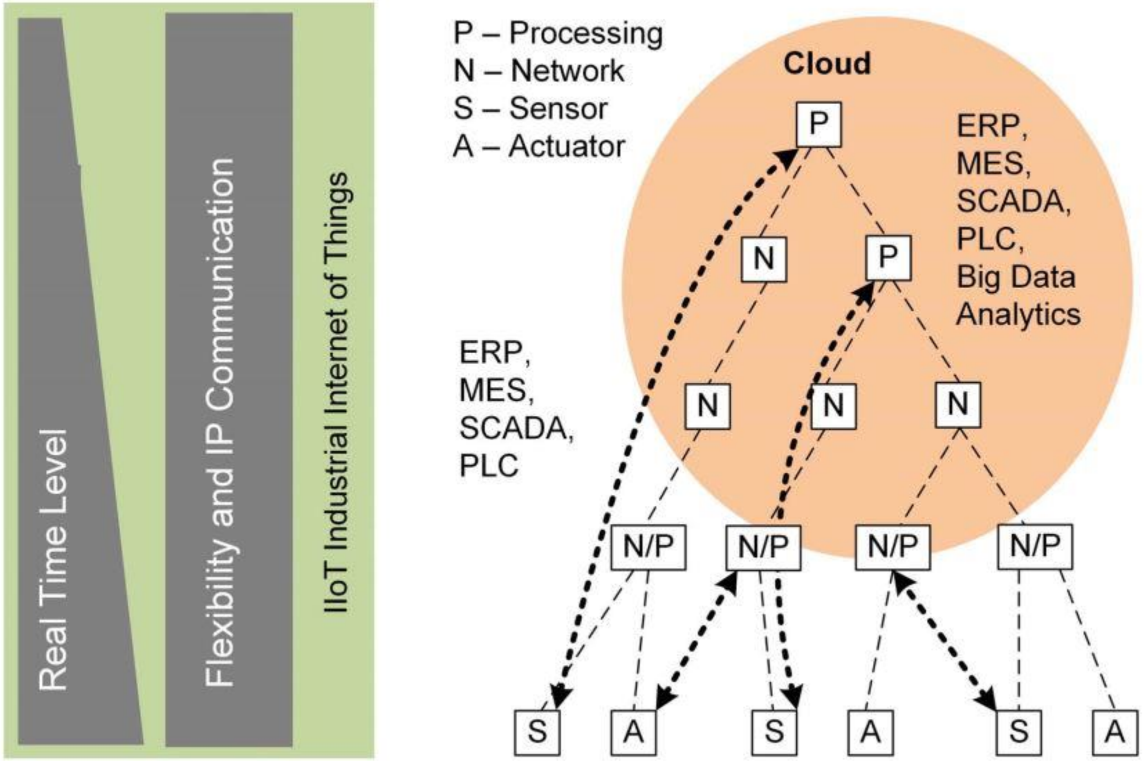
Contents

1. Background:
    - » RTE Networks, IIoT, EtherCAT
2. Objectives summary
3. Layered structure of solution
4. Implementation:
    - » DSMs and RTOS
5. Results:
    - » Successful EtherCAT transaction
    - » PCB
6. Conclusions and further work
7. Questions

Background: RTE Networks, IIoT

- ISO/IEC/IEEE 8802-3:2015, IEC 61158:1999-2000, IEC 61784-Part 2:2008, IEC/IEEE 60802

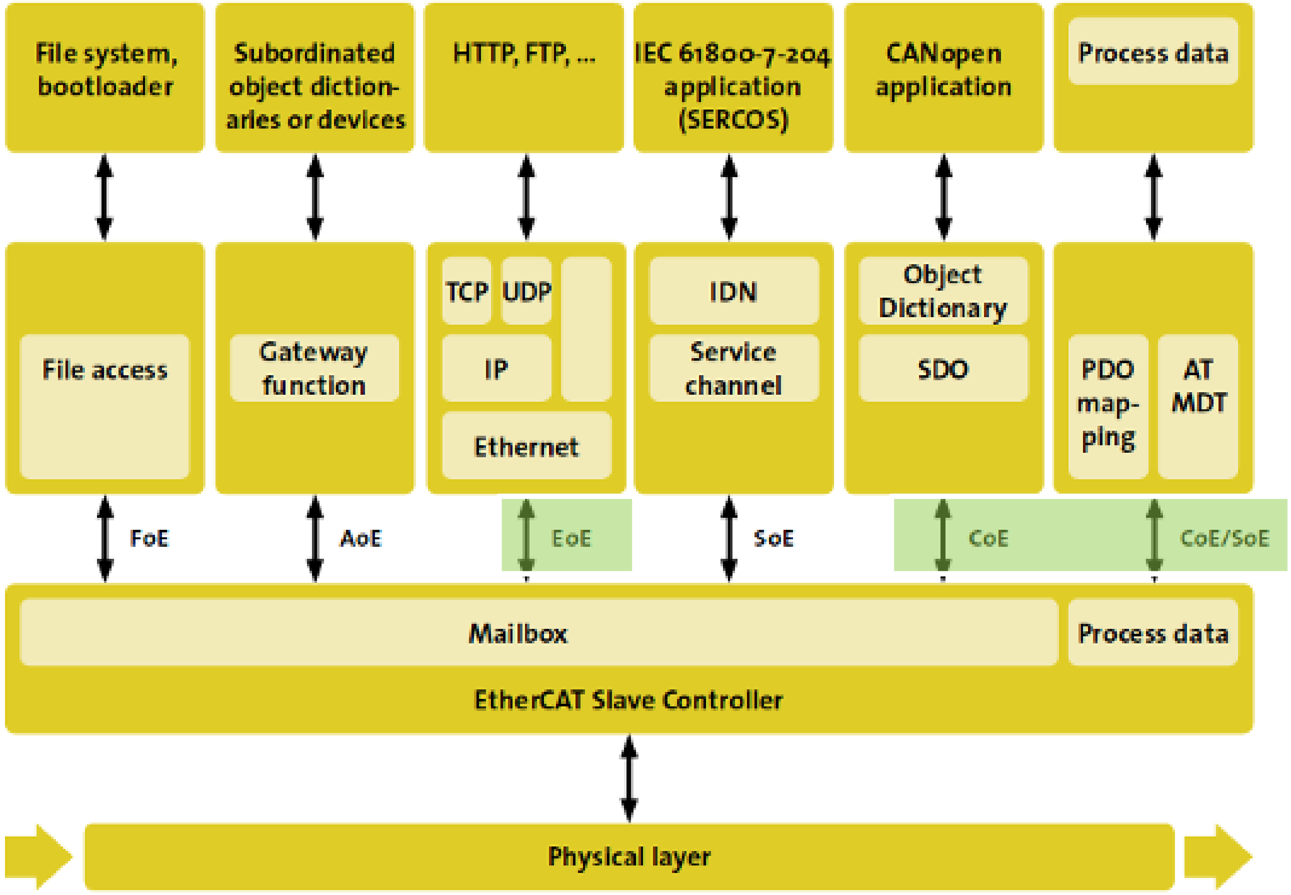
- Interoperability, less gateways
- Direct device access through layers, reliable protocols



Industry 4.0 a flexible automation structure. Industrial Internet of Things, source from [SKJ18]

Background: EtherCAT

- Open industrial protocol application fields
  - Industrial: [MGSR20] and [Nor20]
  - Control loops in robotics: [XJY11] and [DC17]
  - Real Time Robot Software Platforms: [DC17] and [MHF+20]

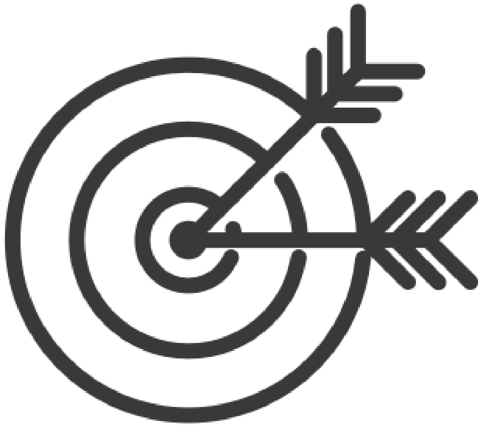


Different communication profiles can coexist in the same system. Source from [Bec20b]

Objectives summary

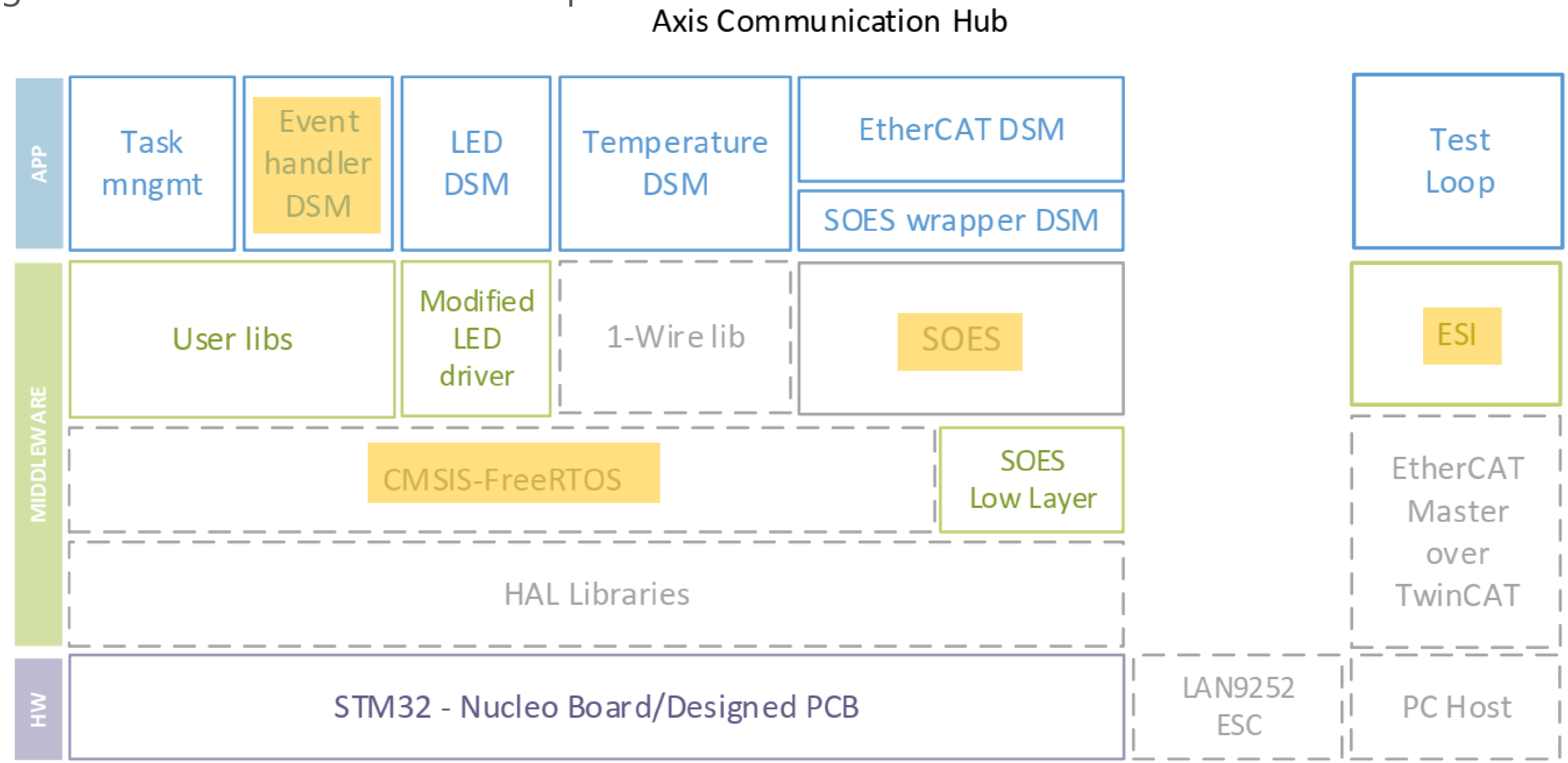
“Develop a device using open-source tools to read out sensor data from a robot axis that can be interfaced with an RTE Network.”

- » Integrating CMSIS-FreeRTOS with SOES library (Open-source tools)
- » Reading out of axis temperature sensors
- » Controlling the axis LED Ring (WS2812b)
- » To design and manufacture a PCB



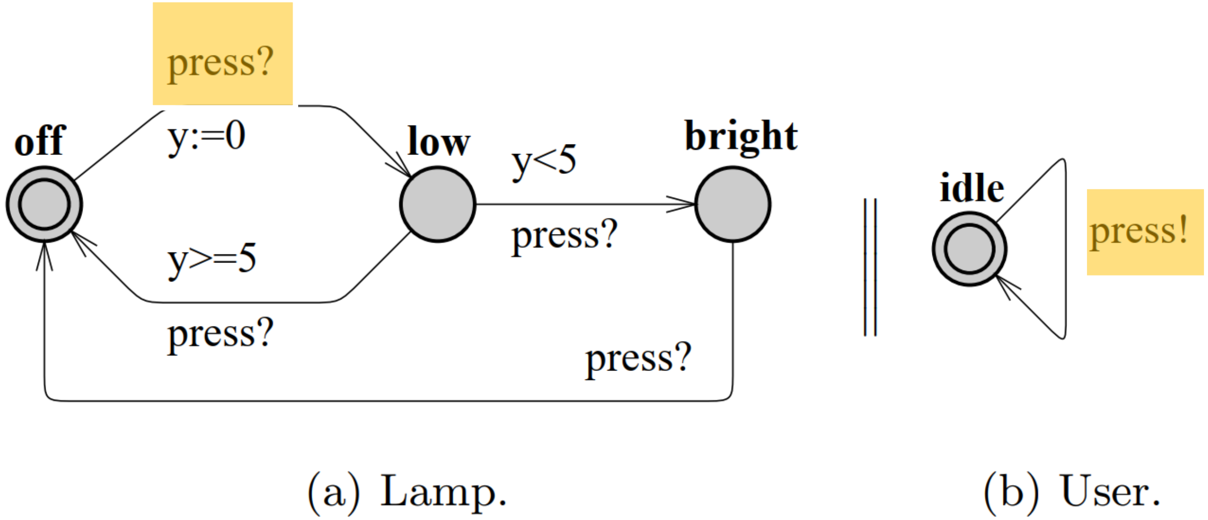
Layered structure of solution

- Layered structure of functional blocks: modular, flexible, open source tools
- Highlighted the most relevant in this implementation



Implementation: background

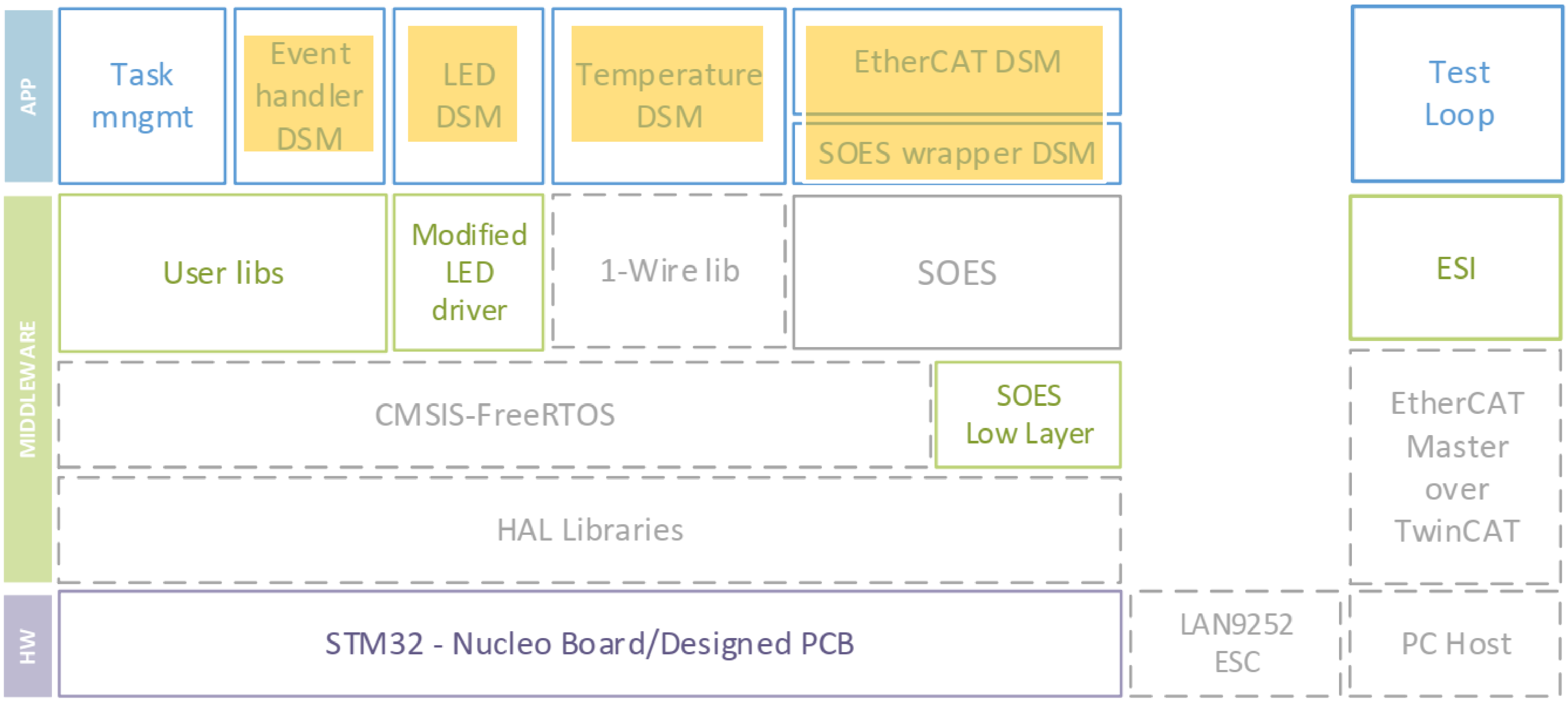
- Each upper functional block as DSM
- Considerations:
  - Moore and Mealy State machine representations.
  - Synchronization channel as in UPPAAL 4.0.



The lamp example. From [BDL06]

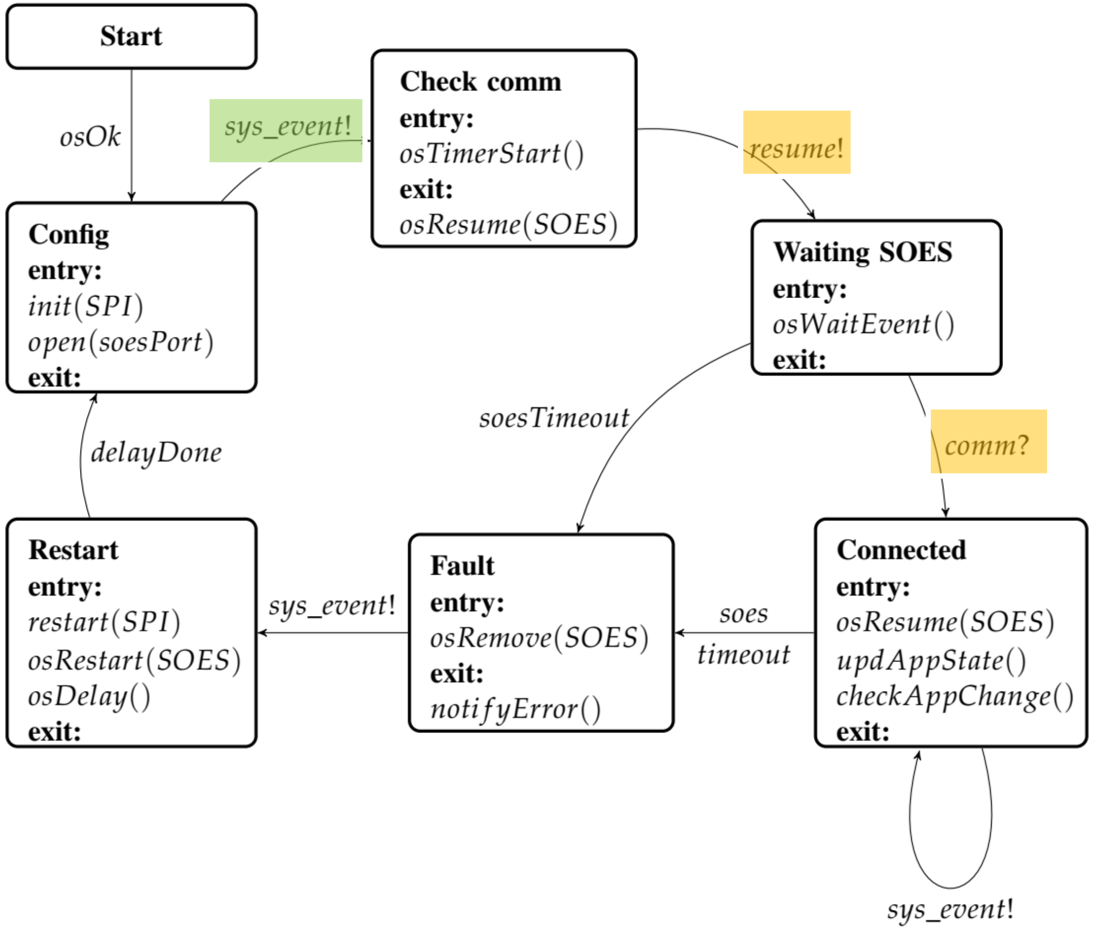
Layered structure

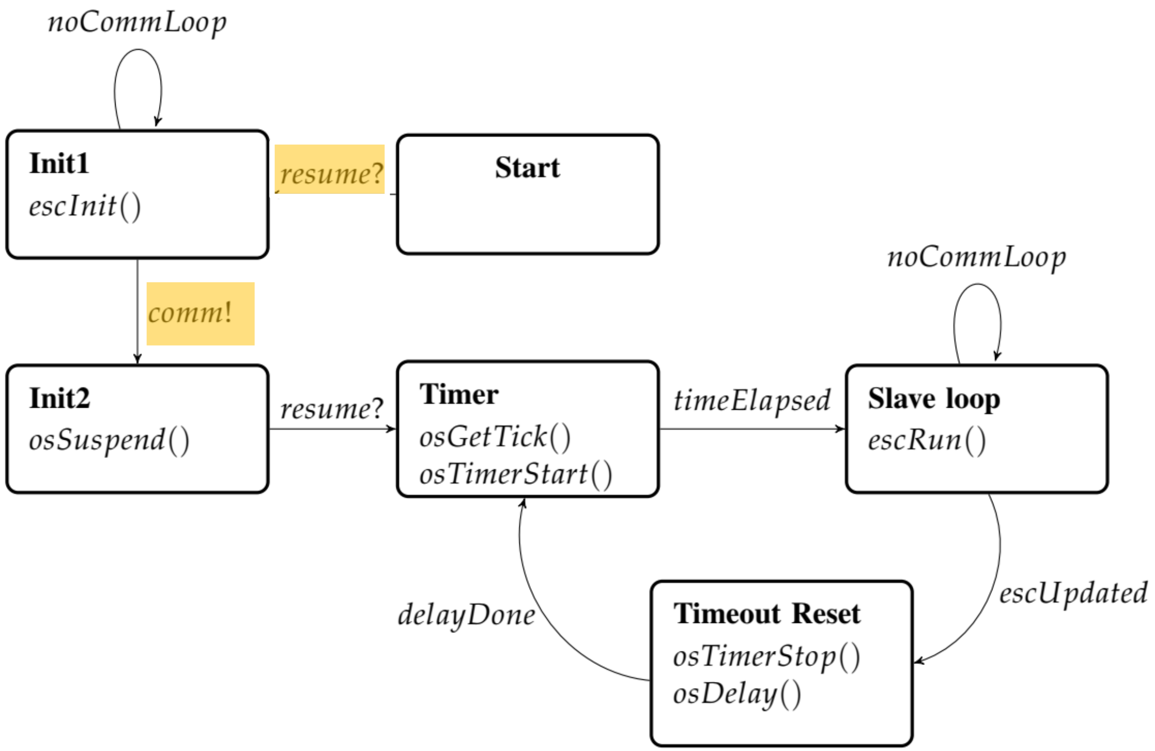
Axis Communication Hub



Implementation: DSMs I

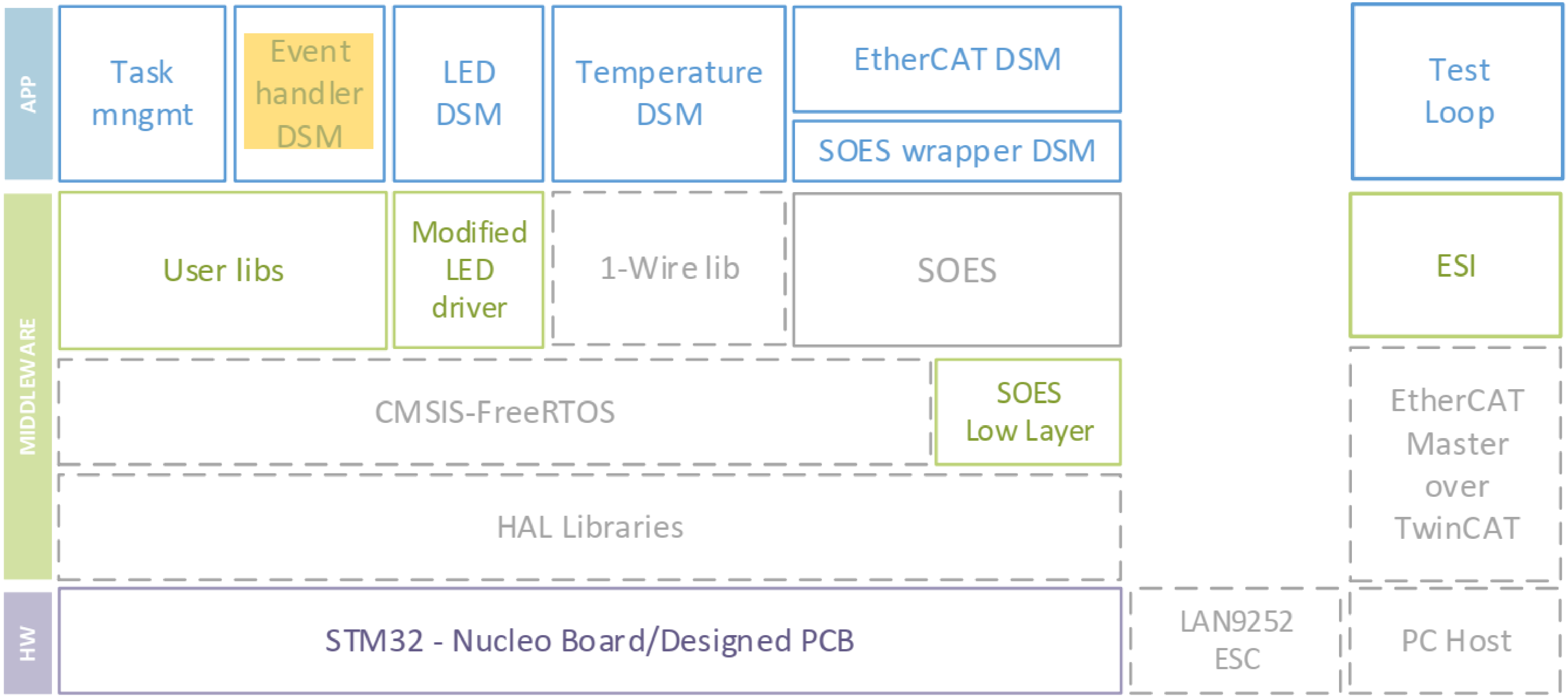
» SOES implementation





Layered structure

Axis Communication Hub



APP

Task mngmt

Event handler DSM

LED DSM

Temperature DSM

EtherCAT DSM

SOES wrapper DSM

Test Loop

MIDDLEWARE

User libs

Modified LED driver

1-Wire lib

SOES

ESI

CMSIS-FreeRTOS

SOES Low Layer

EtherCAT Master over TwinCAT

HAL Libraries

HW

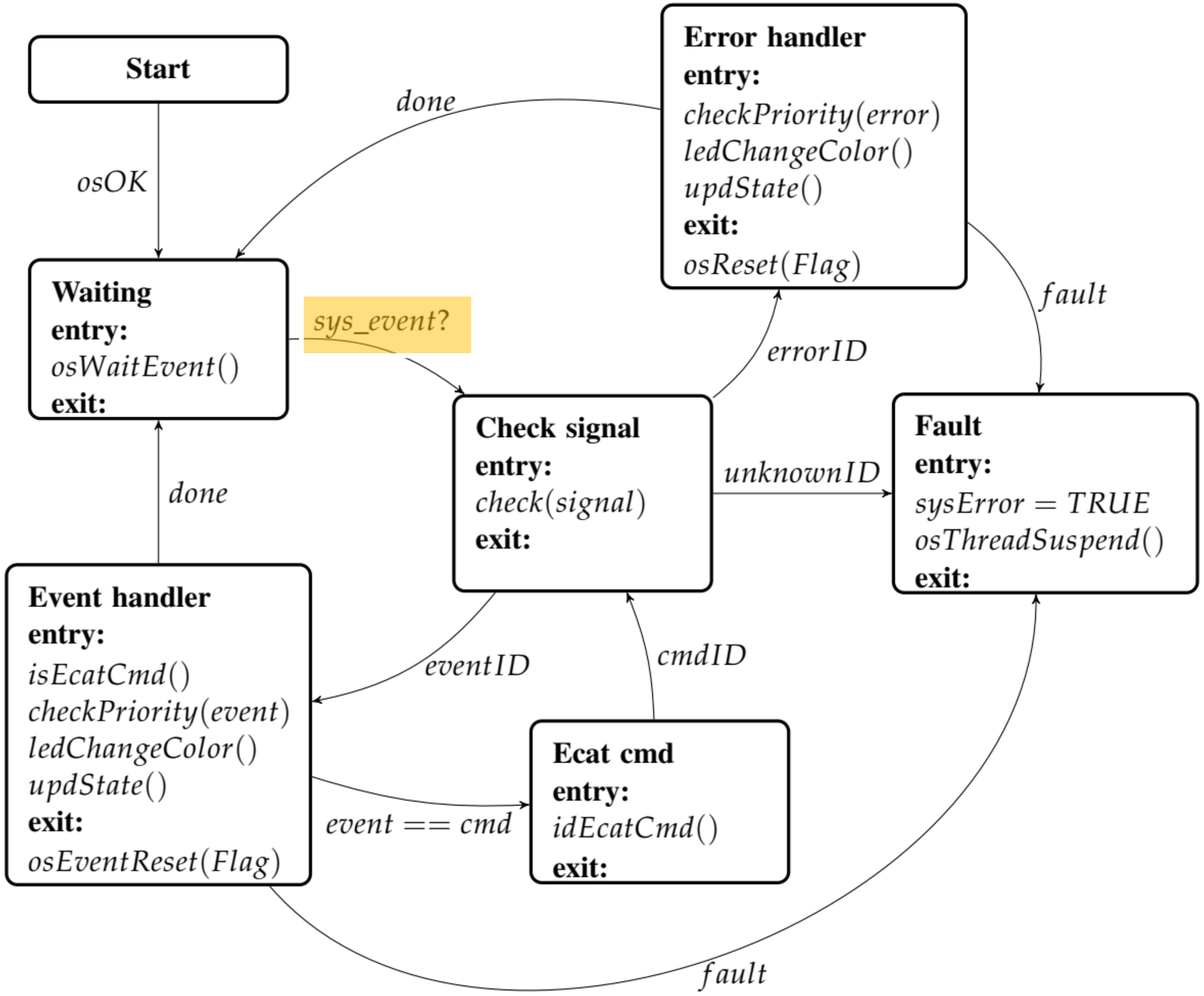
STM32 - Nucleo Board/Designed PCB

LAN9252 ESC

PC Host

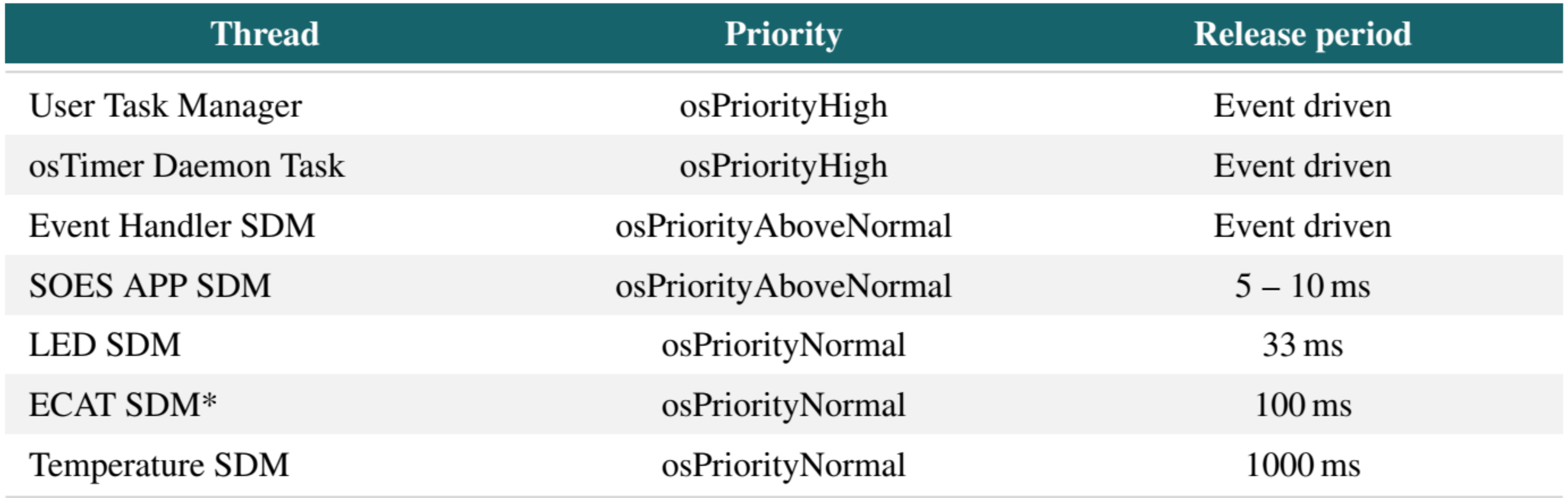
Implementation: Synchronization among DSMs

- Flexible and modular
- Functions for specific error handling can be added
- ECAT command handler foreseen



Implementation: RTOS for synchronization and scheduling

- Fixed priorities
- Pre-emptive priority-based scheduling
- Round robin execution for same priority tasks
- osThreadX, osEventFlagsX, osTimerX, osWaitX, osMutexX helpful functions, info in [Arm20]

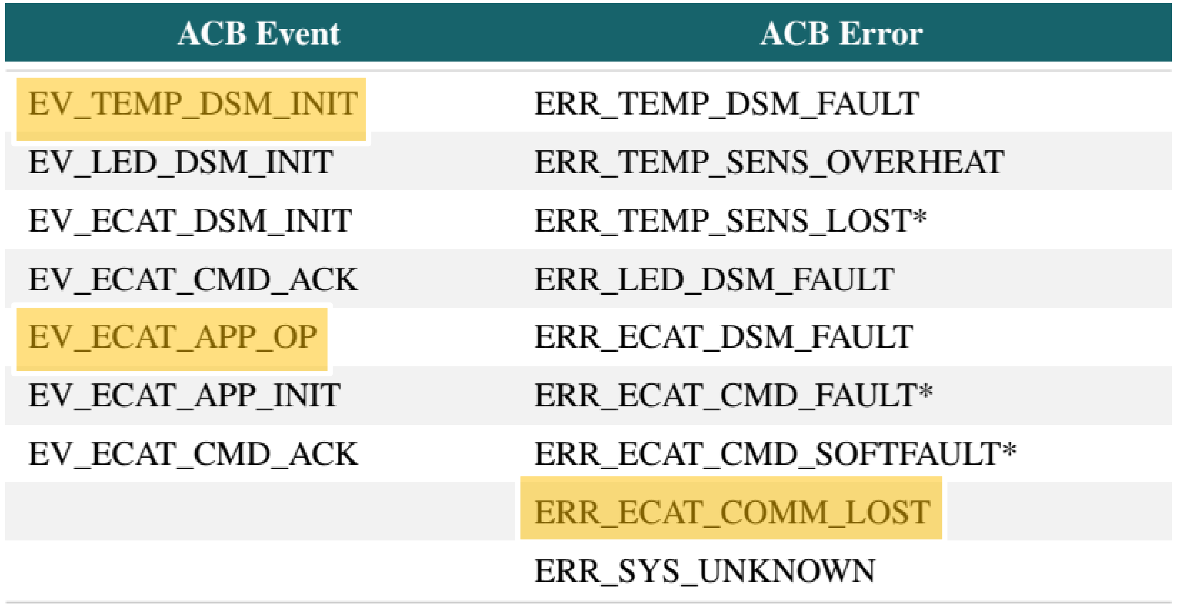


Thread	Priority	Release period
User Task Manager	osPriorityHigh	Event driven
osTimer Daemon Task	osPriorityHigh	Event driven
Event Handler SDM	osPriorityAboveNormal	Event driven
SOES APP SDM	osPriorityAboveNormal	5 – 10 ms
LED SDM	osPriorityNormal	33 ms
ECAT SDM*	osPriorityNormal	100 ms
Temperature SDM	osPriorityNormal	1000 ms

Final priorities' arrangement for main threads. *ECAT SDM is mainly event driven, nevertheless, in the connected state it has a periodic update.

Results: Events and errors handling

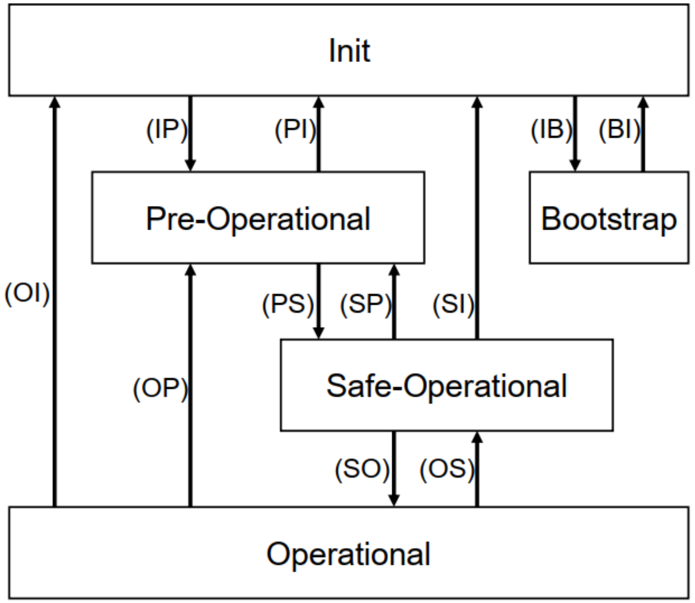
» Events and errors:
Changes are updated within the shared variable through the event handler to notify the LED channels to change the system’s current color

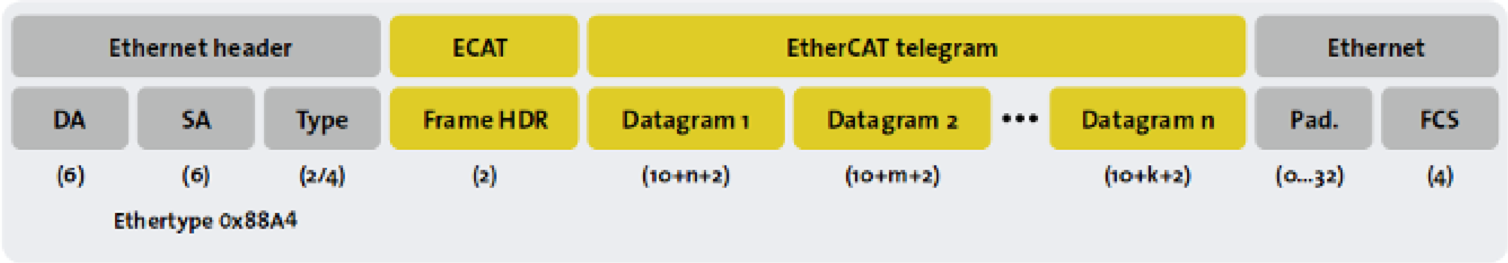


ACB Event	ACB Error
EV_TEMP_DSM_INIT	ERR_TEMP_DSM_FAULT
EV_LED_DSM_INIT	ERR_TEMP_SENS_OVERHEAT
EV_ECAT_DSM_INIT	ERR_TEMP_SENS_LOST*
EV_ECAT_CMD_ACK	ERR_LED_DSM_FAULT
EV_ECAT_APP_OP	ERR_ECAT_DSM_FAULT
EV_ECAT_APP_INIT	ERR_ECAT_CMD_FAULT*
EV_ECAT_CMD_ACK	ERR_ECAT_CMD_SOFTFAULT*
	ERR_ECAT_COMM_LOST
	ERR_SYS_UNKNOWN

Results: EtherCAT’s protocol overview

- AL State Machine
- EtherCAT frame within an IEEE 802.3 frame

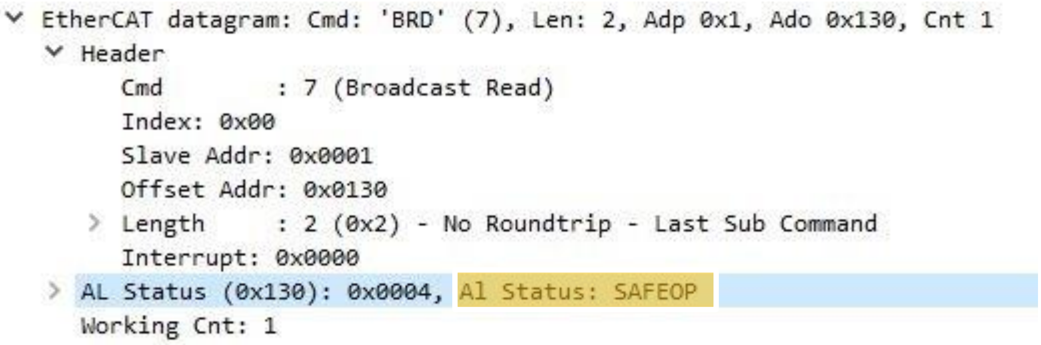




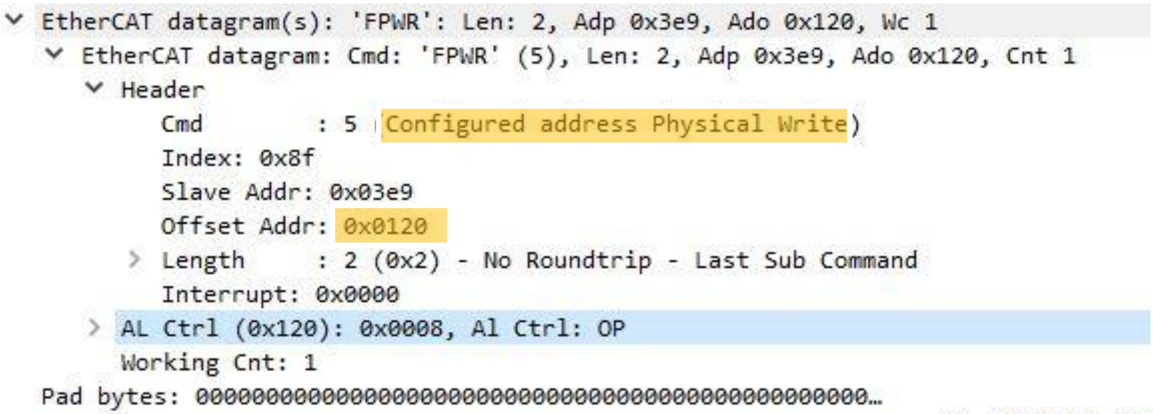
Ethernet header			ECAT	EtherCAT telegram				Ethernet	
DA	SA	Type	Frame HDR	Datagram 1	Datagram 2	•••	Datagram n	Pad.	FCS
(6)	(6)	(2/4)	(2)	(10+n+2)	(10+m+2)		(10+k+2)	(0...32)	(4)
	Ethertype 0x88A4								

Reference from [Bec17b]

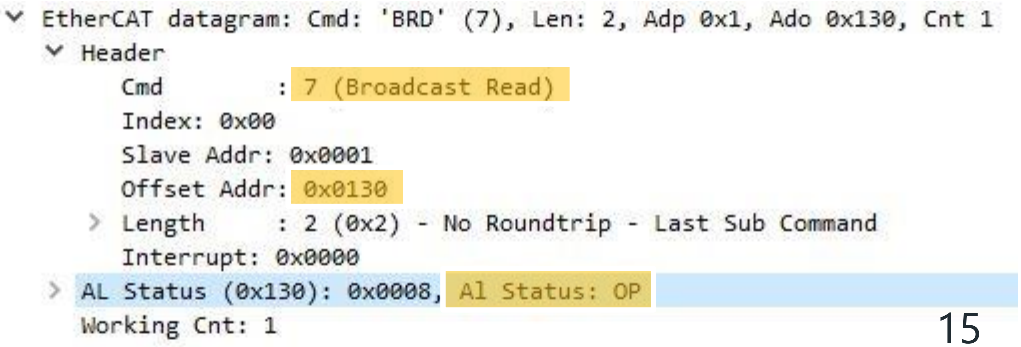
Results: Successful Operational state



```
⌄ EtherCAT datagram: Cmd: 'BRD' (7), Len: 2, Adp 0x1, Ado 0x130, Cnt 1
  ⌄ Header
      Cmd         : 7 (Broadcast Read)
      Index: 0x00
      Slave Addr: 0x0001
      Offset Addr: 0x0130
    > Length      : 2 (0x2) - No Roundtrip - Last Sub Command
      Interrupt: 0x0000
  > AL Status (0x130): 0x0004, Al Status: SAFEOP
    Working Cnt: 1
```



```
⌄ EtherCAT datagram(s): 'FPWR': Len: 2, Adp 0x3e9, Ado 0x120, Wc 1
  ⌄ EtherCAT datagram: Cmd: 'FPWR' (5), Len: 2, Adp 0x3e9, Ado 0x120, Cnt 1
    ⌄ Header
        Cmd         : 5 (Configured address Physical Write)
        Index: 0x8f
        Slave Addr: 0x03e9
        Offset Addr: 0x0120
      > Length      : 2 (0x2) - No Roundtrip - Last Sub Command
        Interrupt: 0x0000
    > AL Ctrl (0x120): 0x0008, Al Ctrl: OP
      Working Cnt: 1
  Pad bytes: 0000000000000000000000000000000000000000000000000…
```

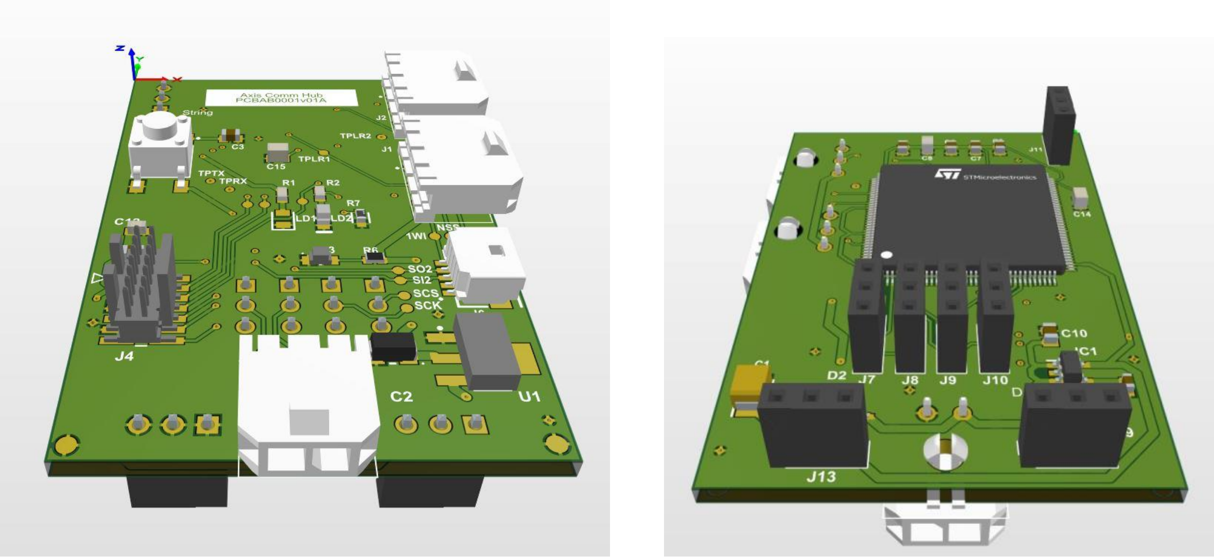


```
⌄ EtherCAT datagram: Cmd: 'BRD' (7), Len: 2, Adp 0x1, Ado 0x130, Cnt 1
  ⌄ Header
      Cmd         : 7 (Broadcast Read)
      Index: 0x00
      Slave Addr: 0x0001
      Offset Addr: 0x0130
    > Length      : 2 (0x2) - No Roundtrip - Last Sub Command
      Interrupt: 0x0000
  > AL Status (0x130): 0x0008, Al Status: OP
    Working Cnt: 1
```

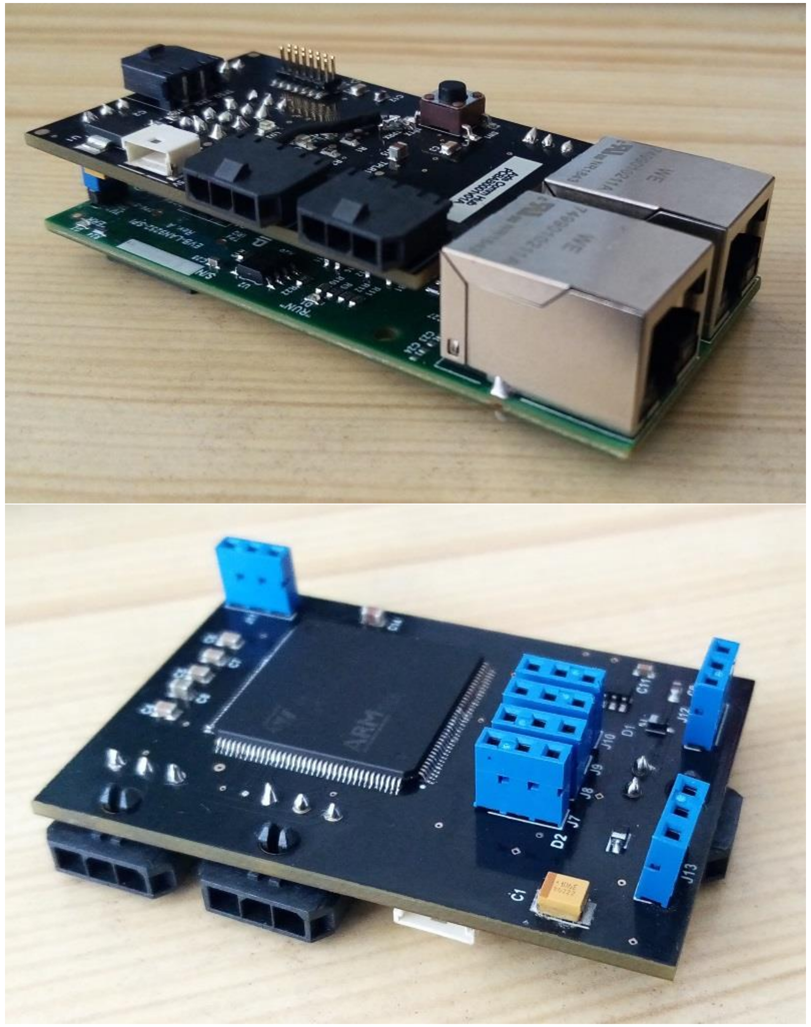
- Data packets obtained through Wireshark running on Master PC (TwinCAT3)

Results: PCB

- Altium Designer
- 4-layered PCB, 38x56mm



3D model



Manufactured PCB

Conclusions and further development

- Firmware based on State Machine abstraction of the logic with a deterministic approach, ease the expansion of features and debug process.
- CMSIS Abstraction layer for RTOS is an open standardized API, so flexible interoperability with other RTOS is granted, e.g. FreeRTOS and RTX Kernel.
- CMSIS package also allows other standardized APIs for example CMSIS NN and CMSIS SVD.
- Open industrial protocol for further development of reliable industrial applications.
- Modular and flexible firmware + experience in working with open source libraries and standardized documentation.

Further development

- FSoE Safety related
  - Multi core embedded systems for redundancy features >> Multi core RTOS

- SoE Hard real time applications 1.5 to 30 us – DC jitter can be reduced to ns.

- Dependability analysis of the Firmware + Hardware

- Software verification of the State Machines with CMSIS tools and optimization of the Hardware resources and execution times for a better and more formal scheduling [Fed17].

References

- » [SKJ18] S. Schriegel, T. Kobzan, and J. Jasperneite. Investigation on a distributed sdn control plane architecture for heterogeneous time sensitive networks. In 2018 14th IEEE International Workshop on Factory Communication Systems (WFCS), pages 1–10, 2018.
- » [MGSR20] M. Metaal, R. Guillaume, R. Steinmetz, and A. Rizk. Integrated industrial ethernet networks: Time-sensitive networking over sdn infrastructure for mixed applications. In 2020 IFIP Networking Conference (Networking), pages 803–808, 2020.
- » [Nor20] M. Norris. A Cannabis Equipment Supplier Turns to Automation to Extract CBD at Commercial Scale. https://www.automationworld.com/home/article/21173096/a-cannabisequipment-supplier-turns-to-automation-to-extract-cbd-at-commercial-scale, August 2020. Last visited: 14/09/2020.
- » [XJY11] H. Xing, H. Jia, and L. Yanqianga. Motion control system using sercos over ethercat Procedia Engineering, 24:749–753, 12 2011.
- » [DC17] R. Delgado and B. Choi. Real-time servo control using ethercat master on real-time embedded linux extensions. International Journal of Applied Engineering Research, 12:1179–1185, 11 2017
- » [MHF+20]C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini. Motion planning for quadrupedal locomotion: Coupled planning, terrain mapping, and whole-body control. IEEE Transactions on Robotics, pages 1–14, 202

References

- [Bec20b] Beckhoff Automation GmbH. EtherCAT - the Ethernet Fieldbus. https://www.ethercat.org/en/technology.html, January 2020. Last visited: 16/09/2020
- [BDL06] G. Behrmann, A. David, and K. Larsen. A tutorial on uppaal 4.0. Department of computer science, Aalborg university, 2006
- [Arm20] Arm Ltd. Real-Time Operating System: API and RTX Reference Implementation. https://www.keil.com/pack/doc/CMSIS/RTOS2/html/index.html, April 2020. Last visited: 20/09/2020.
- [Bec17b] Beckhoff Automation GmbH. ETG.1000.4 Data Link Layer protocol specification. https://www.beckhoff.com/english/downloadfinder, November 2017. Last visited: 16/09/2020.
- [Fed17] D. Fedasyuk, R. Chopey and B. Knysh, "Architecture of a tool for automated testing the worst-case execution time of real-time embedded systems' firmware," 2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), Lviv, 2017, pp. 278-281, doi: 10.1109/CADSM.2017.7916134.
- Reference: https://arm-software.github.io/CMSIS_5/General/html/index.html

HAN'S ROBOT

WE SERVE HUMANITY

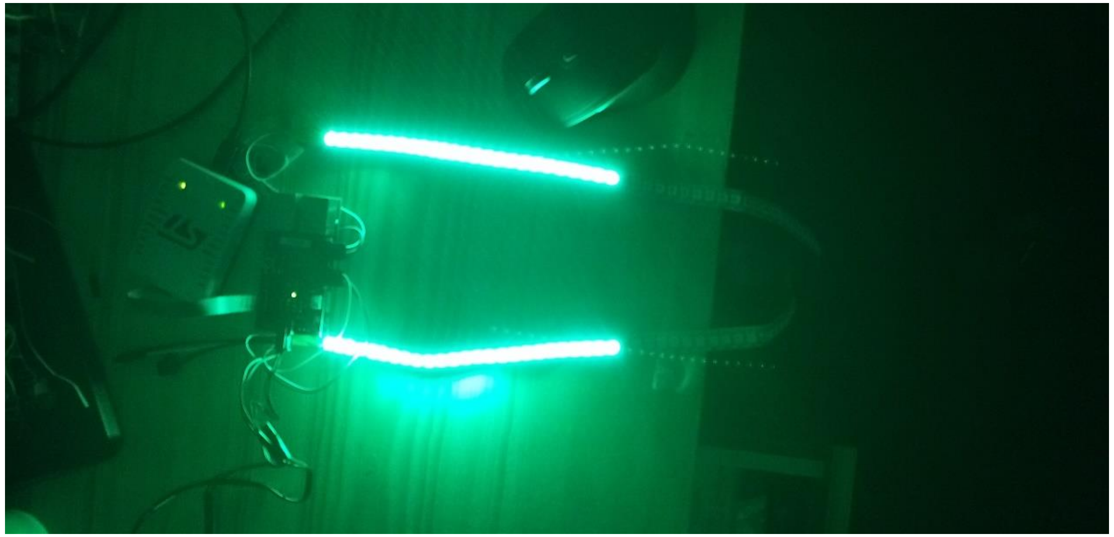
Questions

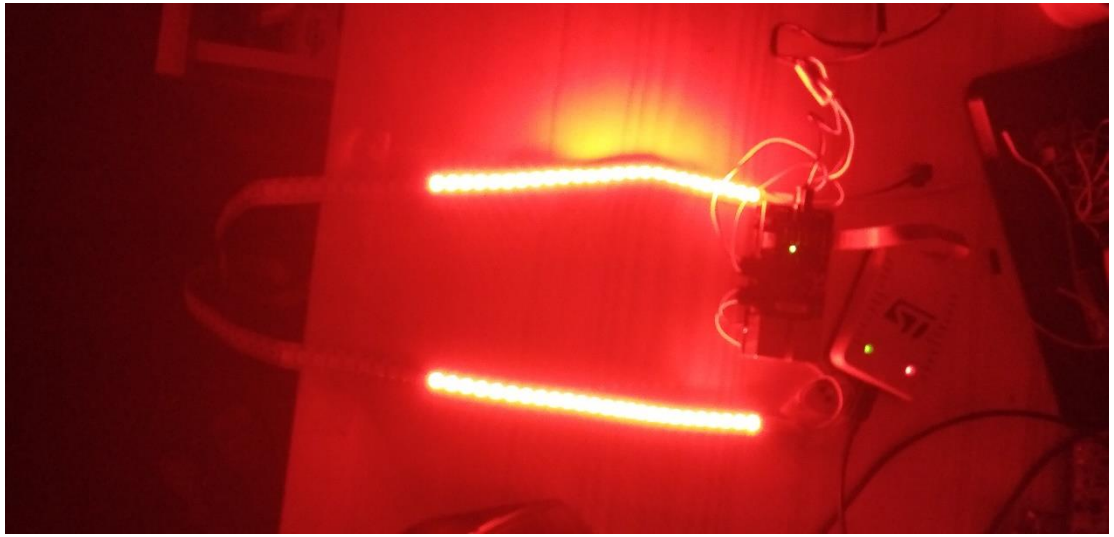
Dankeschön für Ihre Aufmerksamkeit!

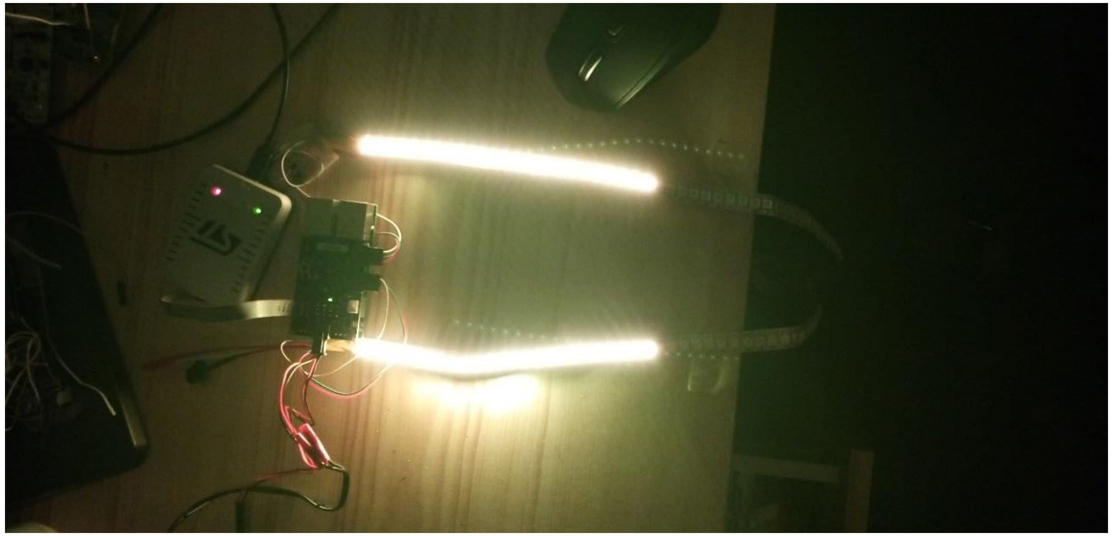
HAN'S ROBOT
WE SERVE HUMANITY

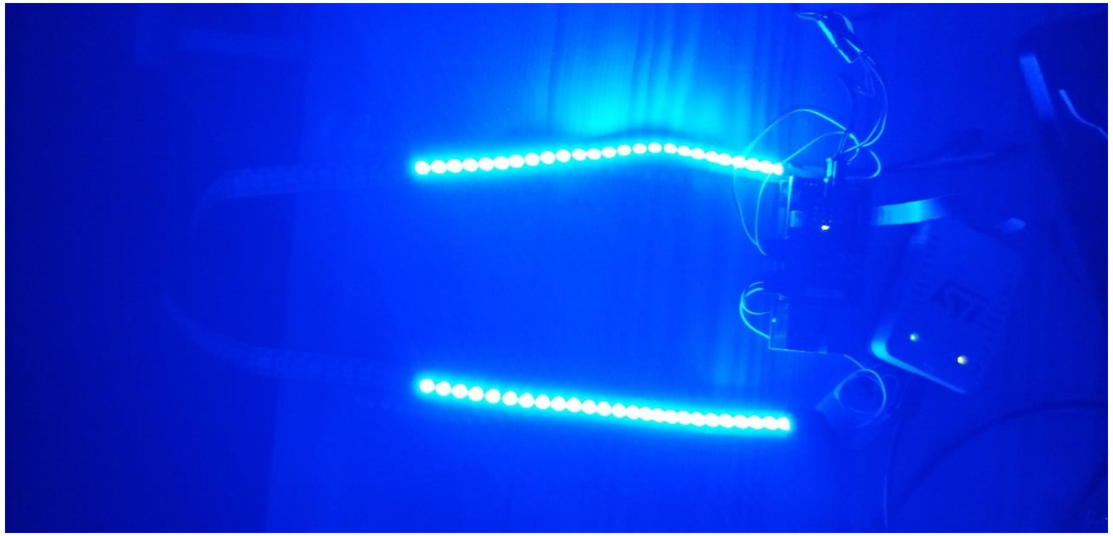
Extra information

Photos



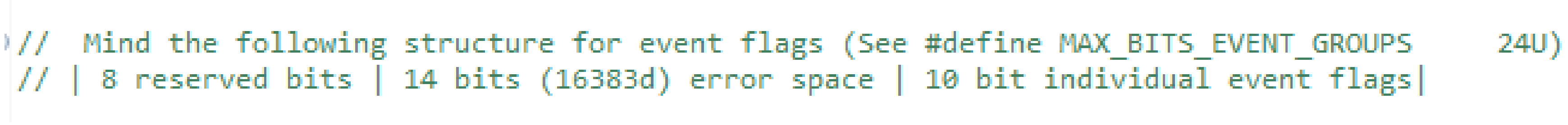




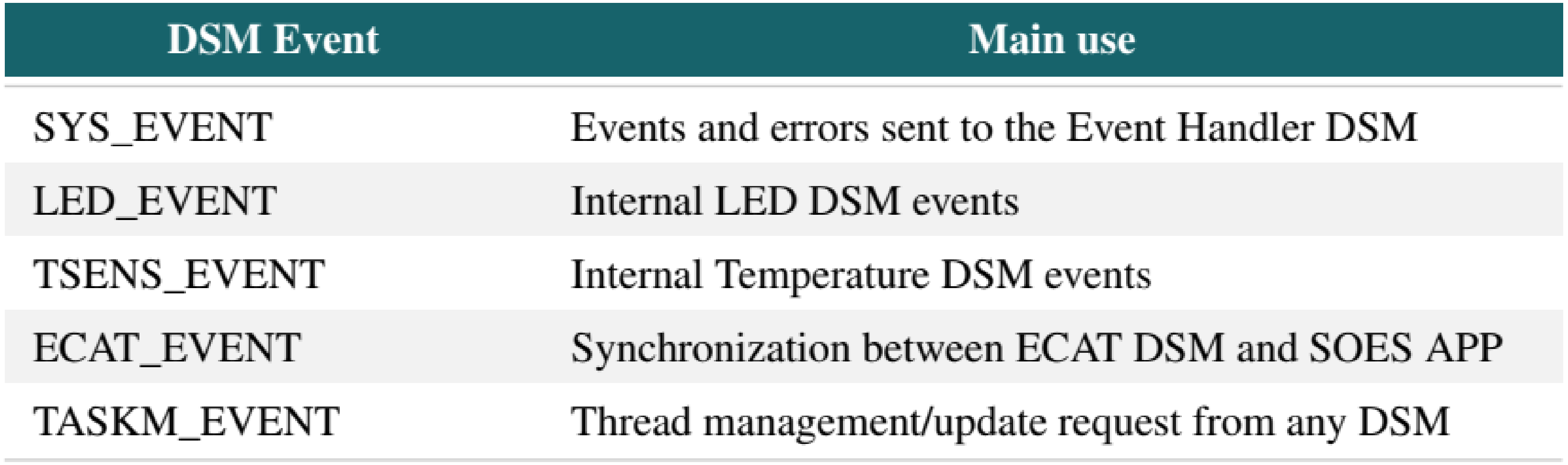


Synchronization with CMSIS-RTOS

- Synchronization methods:
  - EventFlags (extension of ThreadFlags)
- Semaphores:
  - signalling,
  - multiplex,
  - rendezvous,
  - barrier turnstile,
  - semaphore barrier.



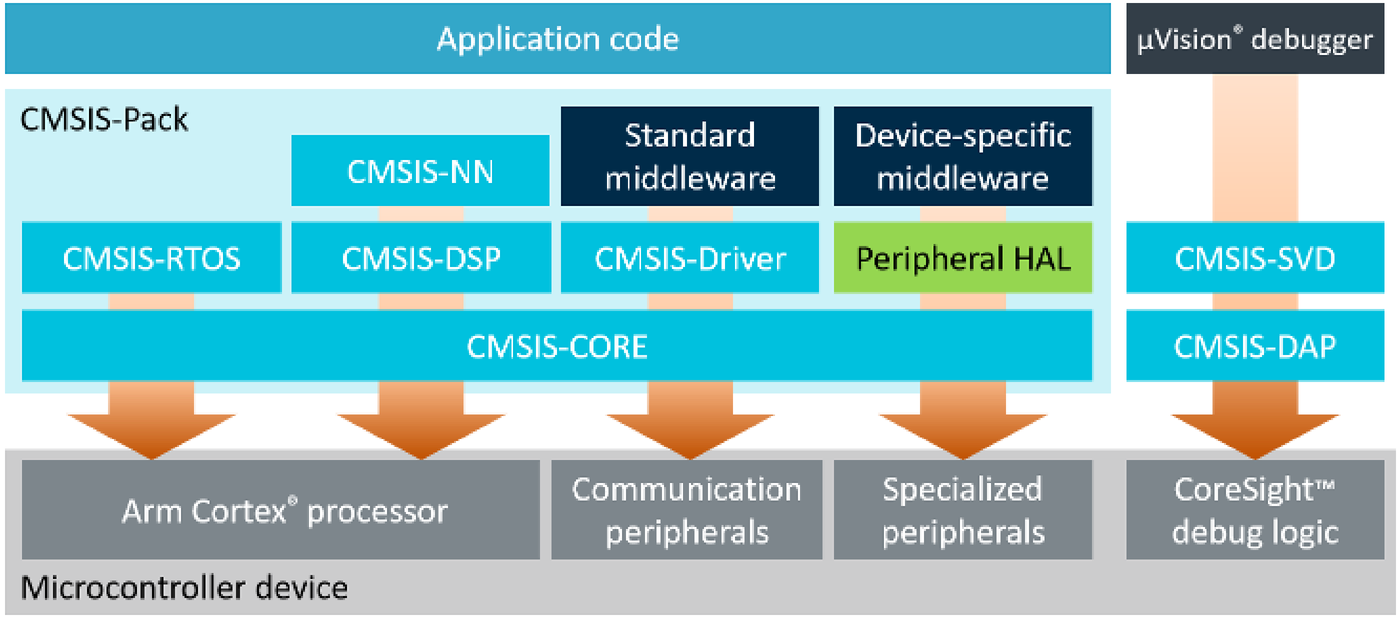
```
//  Mind the following structure for event flags (See #define MAX_BITS_EVENT_GROUPS     24U)
// | 8 reserved bits | 14 bits (16383d) error space | 10 bit individual event flags|
```



DSM Event	Main use
SYS_EVENT	Events and errors sent to the Event Handler DSM
LED_EVENT	Internal LED DSM events
TSENS_EVENT	Internal Temperature DSM events
ECAT_EVENT	Synchronization between ECAT DSM and SOES APP
TASKM_EVENT	Thread management/update request from any DSM

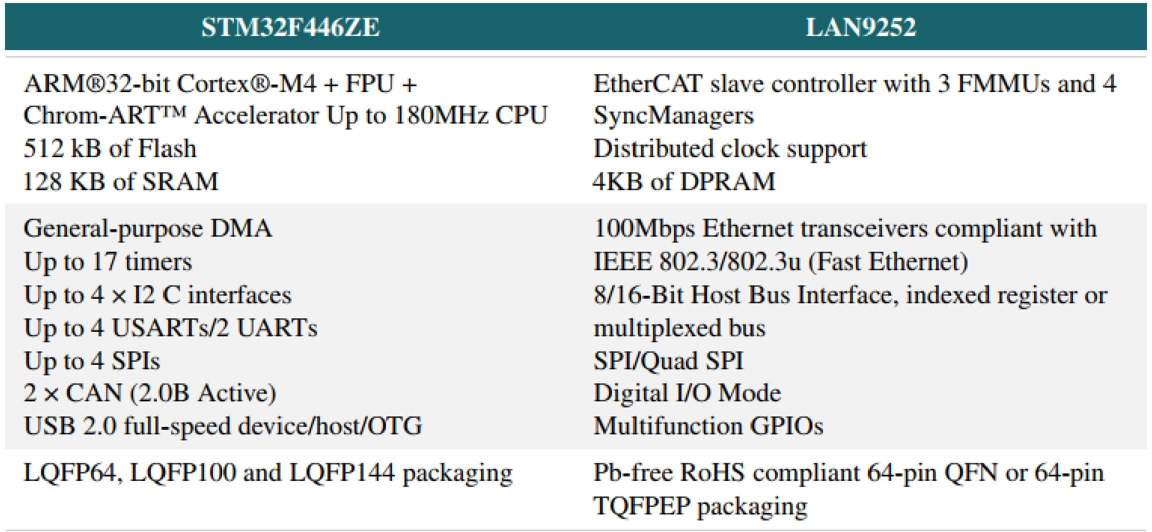
CMSIS Abstraction layer

- Simple software interfaces to the processor and peripherals, software re-use, reducing the time to market for new devices
- Common approach for RTOS, peripheral and middleware components.
- **CMSIS-Zone for configuration of multiple processors****

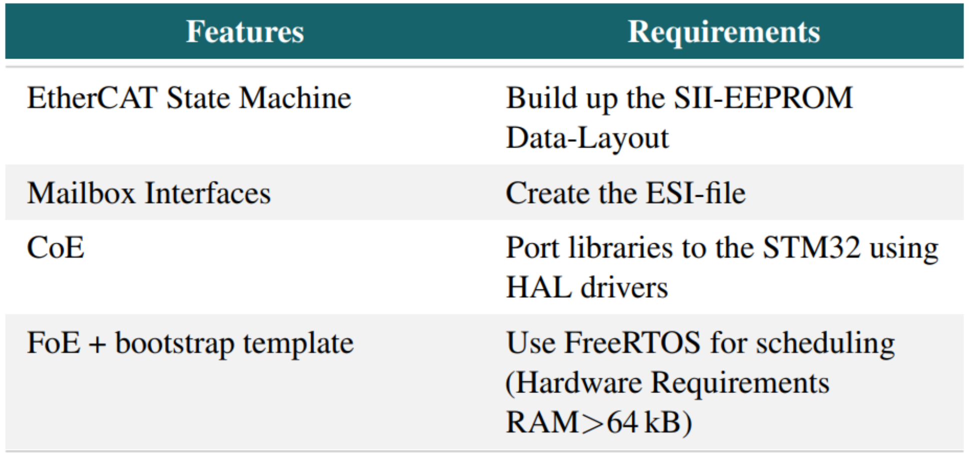


- Common API for real-time operating systems along with a reference implementation based on RTX.
- Compliant with ANSI C (C99) and C++ (C++03).
- Provided free of charge by Arm under the Apache 2.0 License

Technical specifications of the hardware



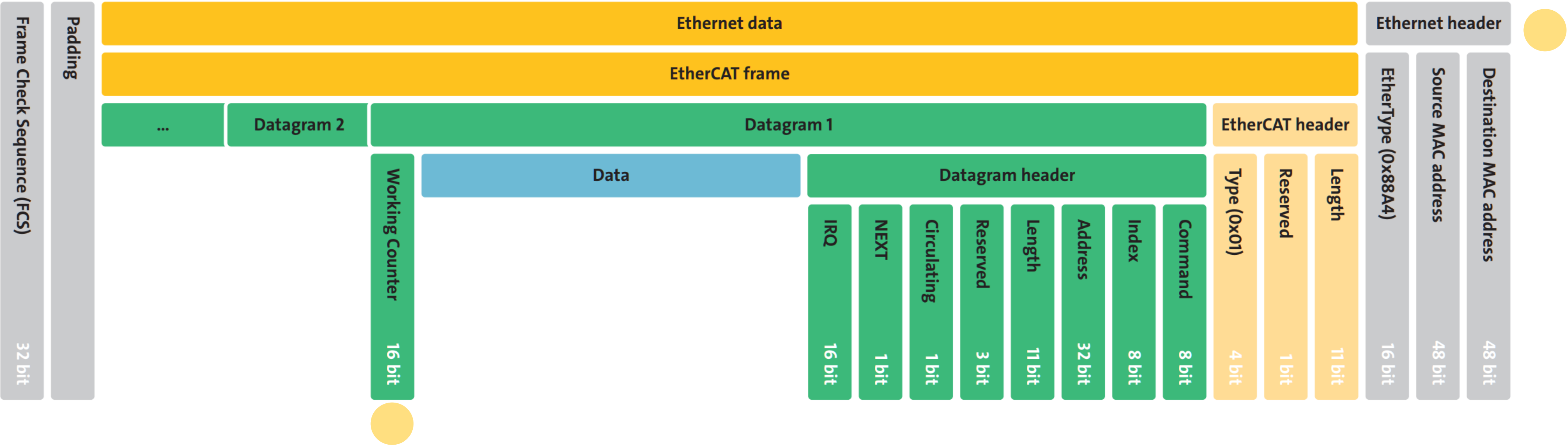
STM32F446ZE	LAN9252
ARM®32-bit Cortex®-M4 + FPU + Chrom-ART™ Accelerator Up to 180MHz CPU 512 kB of Flash 128 KB of SRAM	EtherCAT slave controller with 3 FMMUs and 4 SyncManagers Distributed clock support 4KB of DPRAM
General-purpose DMA Up to 17 timers Up to 4 × I2 C interfaces Up to 4 USARTs/2 UARTs Up to 4 SPIs 2 × CAN (2.0B Active) USB 2.0 full-speed device/host/OTG	100Mbps Ethernet transceivers compliant with IEEE 802.3/802.3u (Fast Ethernet) 8/16-Bit Host Bus Interface, indexed register or multiplexed bus SPI/Quad SPI Digital I/O Mode Multifunction GPIOs
LQFP64, LQFP100 and LQFP144 packaging	Pb-free RoHS compliant 64-pin QFN or 64-pin TQFPEP packaging

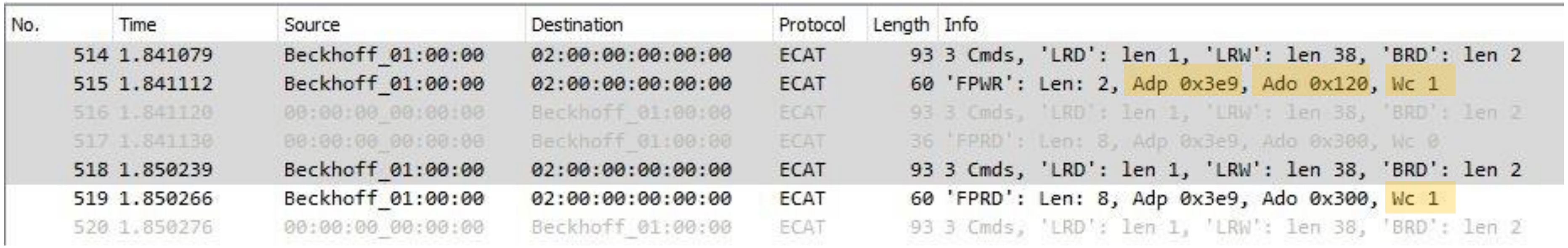


Features	Requirements
EtherCAT State Machine	Build up the SII-EEPROM Data-Layout
Mailbox Interfaces	Create the ESI-file
CoE	Port libraries to the STM32 using HAL drivers
FoE + bootstrap template	Use FreeRTOS for scheduling (Hardware Requirements RAM>64 kB)

Results: Data frame

» Transaction



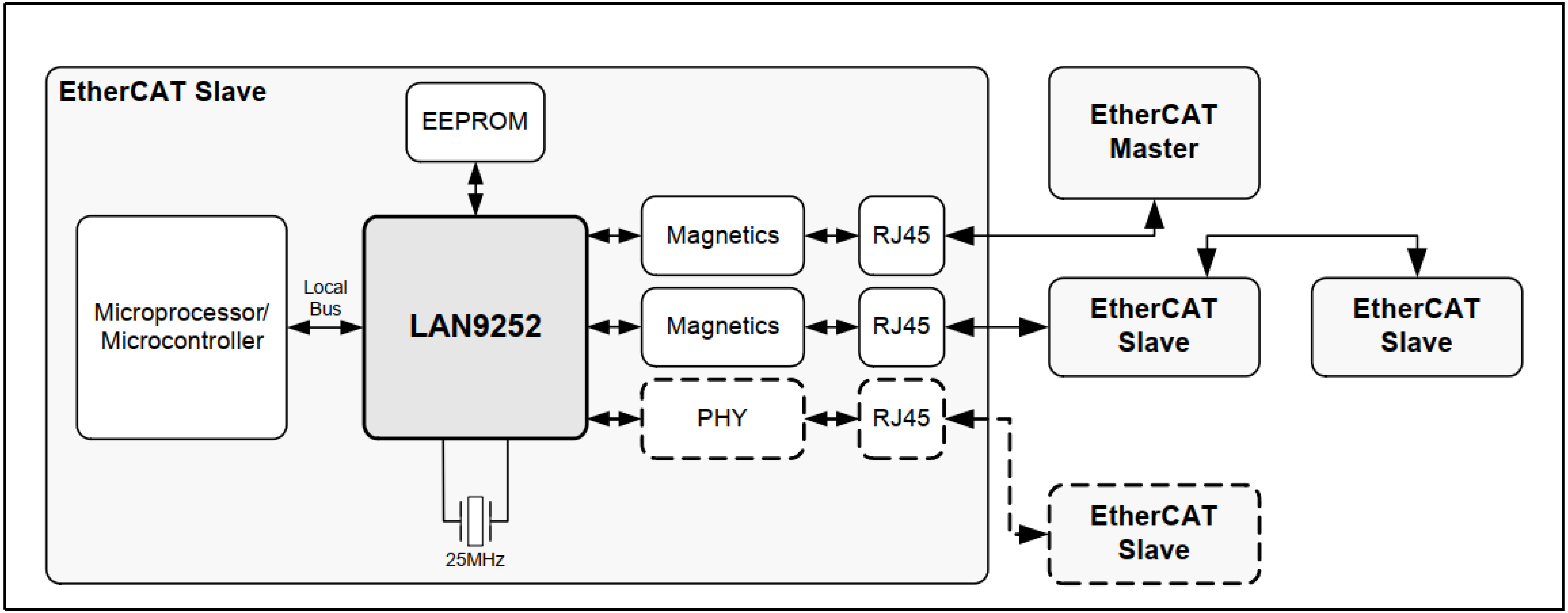


```
No.     Time      Source             Destination        Protocol  Length Info
    514 1.841079  Beckhoff_01:00:00  02:00:00:00:00:00  ECAT          93 3 Cmds, 'LRD': len 1, 'LRW': len 38, 'BRD': len 2
    515 1.841112  Beckhoff_01:00:00  02:00:00:00:00:00  ECAT          60 'FPWR': Len: 2, Adp 0x3e9, Ado 0x120, Wc 1
    516 1.841120  00:00:00_00:00:00  Beckhoff_01:00:00  ECAT          93 3 Cmds, 'LRD': len 1, 'LRW': len 38, 'BRD': len 2
    517 1.841130  00:00:00_00:00:00  Beckhoff_01:00:00  ECAT          36 'FPRD': Len: 8, Adp 0x3e9, Ado 0x300, Wc 0
    518 1.850239  Beckhoff_01:00:00  02:00:00:00:00:00  ECAT          93 3 Cmds, 'LRD': len 1, 'LRW': len 38, 'BRD': len 2
    519 1.850266  Beckhoff_01:00:00  02:00:00:00:00:00  ECAT          60 'FPRD': Len: 8, Adp 0x3e9, Ado 0x300, Wc 1
    520 1.850276  00:00:00_00:00:00  Beckhoff_01:00:00  ECAT          93 3 Cmds, 'LRD': len 1, 'LRW': len 38, 'BRD': len 2
```

Ethernet frame (IEEE802.3): 64-1518 Bytes (up to 1522 Bytes if VLAN is used)

ESC

- System Block Diagram: LAN9252 Structure, reference from datasheet



Other relevant tasks within the development

- Understanding the EtherCAT protocol specification
- Writing the ESI file with snippets from other applications (ET1100 and PIC32 Development boards)
- The overall consulting of MCU’s datasheet to proper usage of peripherals (PWM+DMA)
- Better understanding of the HAL specification provided by STM32
- Constant consulting of CMSIS-RTOS/KEIL project
- Understanding the data sheet of LAN9252 to proper configuration and integration with MCU
- Integration of the OW open-source library
- Relation of

Advantages

- From procedural-based 'C' code on small 8-/16-bit microcontrollers to inherently fostering structured code development which is enforced by the RTOS application programming interface (API).
- 500 bytes of RAM and 5k bytes of code, Flash memory is 512 k and 128 k RAM
- The RTOS itself consists of a scheduler which supports round-robin, pre-emptive and co-operative multitasking of program threads, as well as time and memory management services.
- Debugging by thread isolation of faults.
- pre-emptive priority-based scheduling

Approach for avoiding errors

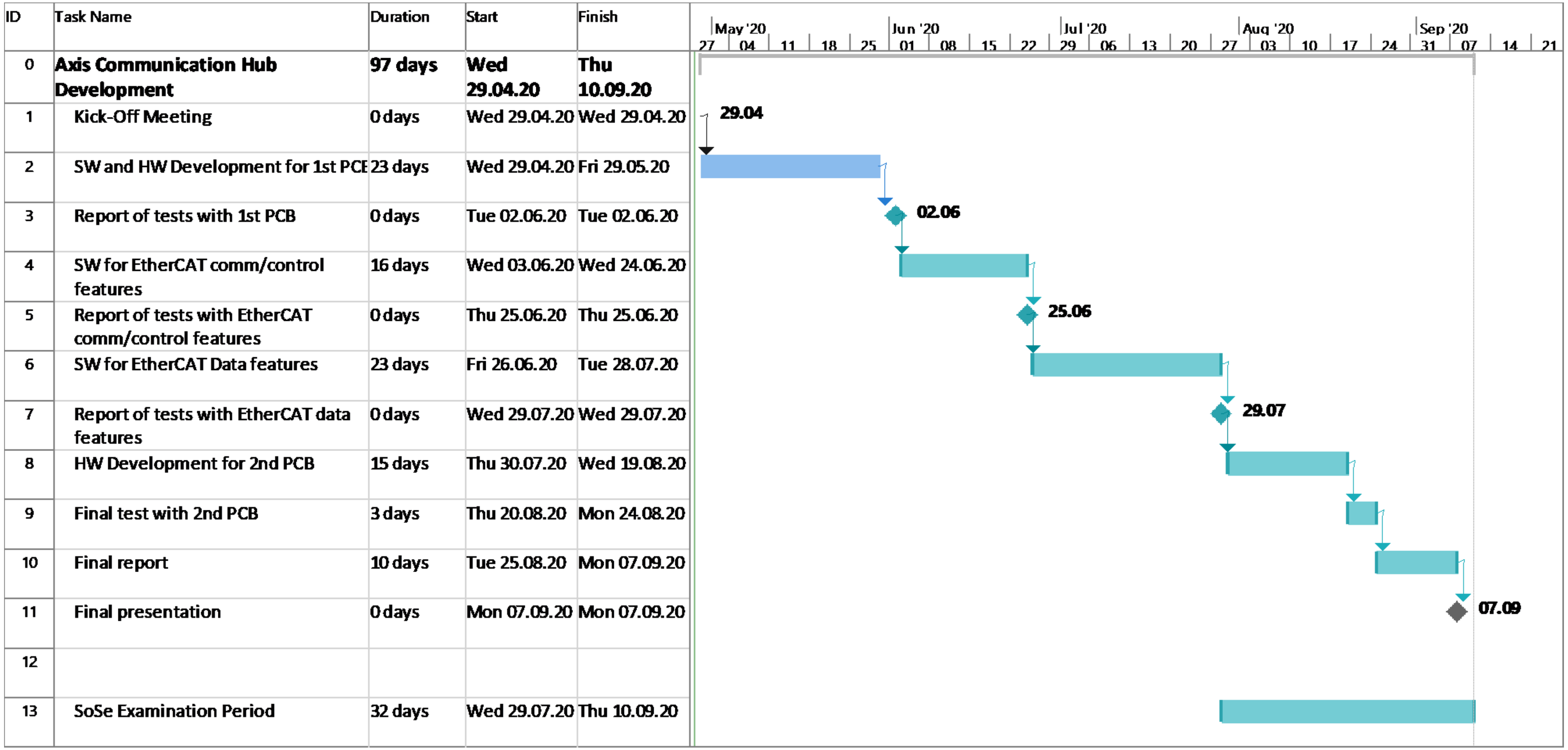
- Avoid dynamic allocation of osObjects as they could lead to memory fragmentation.
- Multiple instances of the same code differentiated by arguments during their call, for example, the callback functions of the ostimeouts
- Usage of virtual timers
- To avoid priority inversion there is only one flag that sets if the temperature is accessible or not.
- Nevertheless there are Data exchange objects that can be allocated for a more formal asynchronous method of communication. Message queue and mail queue. Memory pool is the transmission of a pointer. (Zero Copy Mailbox)

Debugging methods

- Debugging Methods
- It is also possible to monitor the maximum stack memory usage during run time. If you check the "Stack Usage Watermark" option, a pattern (0xCC) is written into each stack space. During runtime, this watermark is used to calculate the maximum memory usage. In Arm Keil MDK, this figure is reported in the threads section of the View - Watch Window - RTX RTOS window.
- This section also allows us to select whether the threads are running in privileged or unprivileged mode. The last option allows us to define the processor operating mode for the user threads. If you want an easy life, leave this set to "privileged mode" and you will have full access to all the processor features. However, if you are writing a safety critical or secure application then "unprivileged mode" can be used to prevent thread access to critical processor registers limiting run time errors or attempts at intrusion.
- `__WEAK uint32_t osRtxErrorNotify`

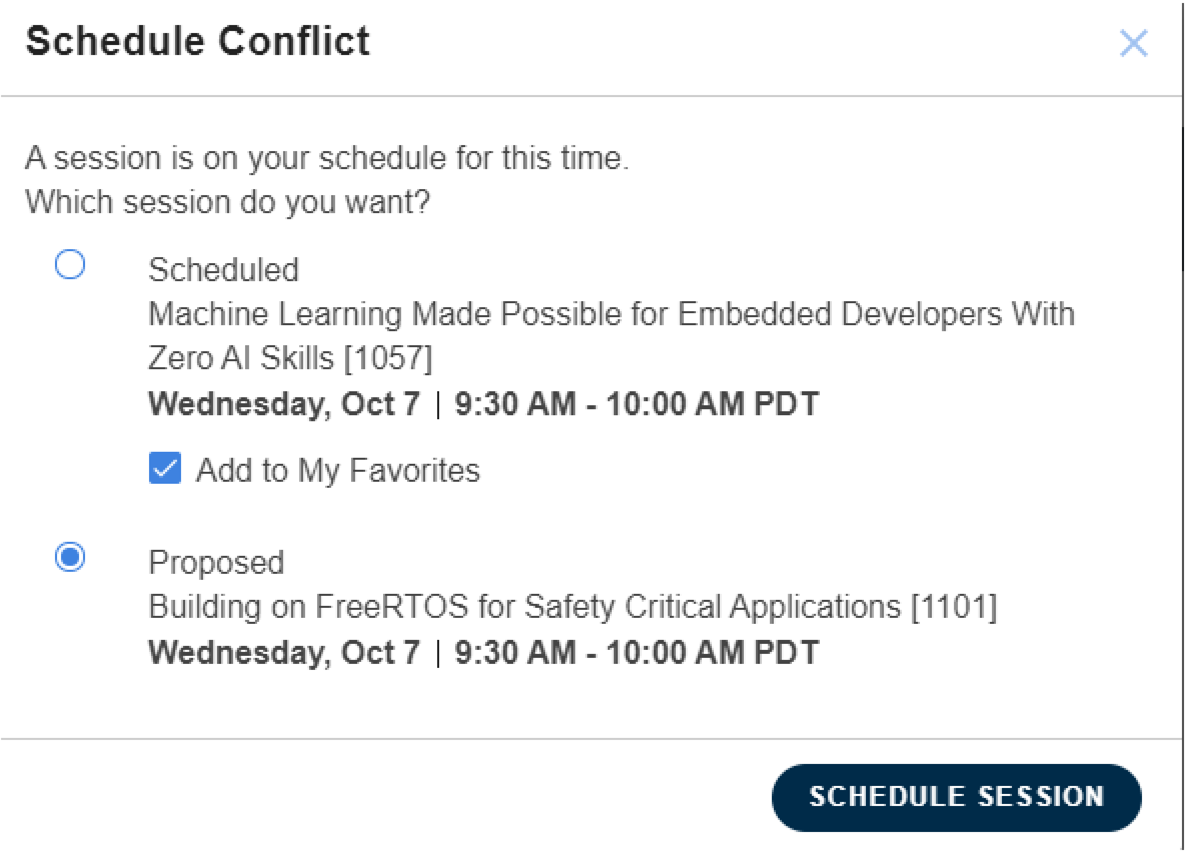
Gantt Chart

- Duration: ~4 Months
- Official start: 29.04 Final Presentation: 07.09 (Proposal)



ID	Task Name	Duration	Start	Finish
0	**Axis Communication Hub Development**	**97 days**	**Wed 29.04.20**	**Thu 10.09.20**
1	Kick-Off Meeting	0 days	Wed 29.04.20	Wed 29.04.20
2	SW and HW Development for 1st PCE	23 days	Wed 29.04.20	Fri 29.05.20
3	Report of tests with 1st PCB	0 days	Tue 02.06.20	Tue 02.06.20
4	SW for EtherCAT comm/control features	16 days	Wed 03.06.20	Wed 24.06.20
5	Report of tests with EtherCAT comm/control features	0 days	Thu 25.06.20	Thu 25.06.20
6	SW for EtherCAT Data features	23 days	Fri 26.06.20	Tue 28.07.20
7	Report of tests with EtherCAT data features	0 days	Wed 29.07.20	Wed 29.07.20
8	HW Development for 2nd PCB	15 days	Thu 30.07.20	Wed 19.08.20
9	Final test with 2nd PCB	3 days	Thu 20.08.20	Mon 24.08.20
10	Final report	10 days	Tue 25.08.20	Mon 07.09.20
11	Final presentation	0 days	Mon 07.09.20	Mon 07.09.20
12				
13	SoSe Examination Period	32 days	Wed 29.07.20	Thu 10.09.20

Extra



Schedule Conflict

A session is on your schedule for this time.
Which session do you want?

○ Scheduled
Machine Learning Made Possible for Embedded Developers With Zero AI Skills [1057]
Wednesday, Oct 7 | 9:30 AM - 10:00 AM PDT

☑ Add to My Favorites

● Proposed
Building on FreeRTOS for Safety Critical Applications [1101]
Wednesday, Oct 7 | 9:30 AM - 10:00 AM PDT