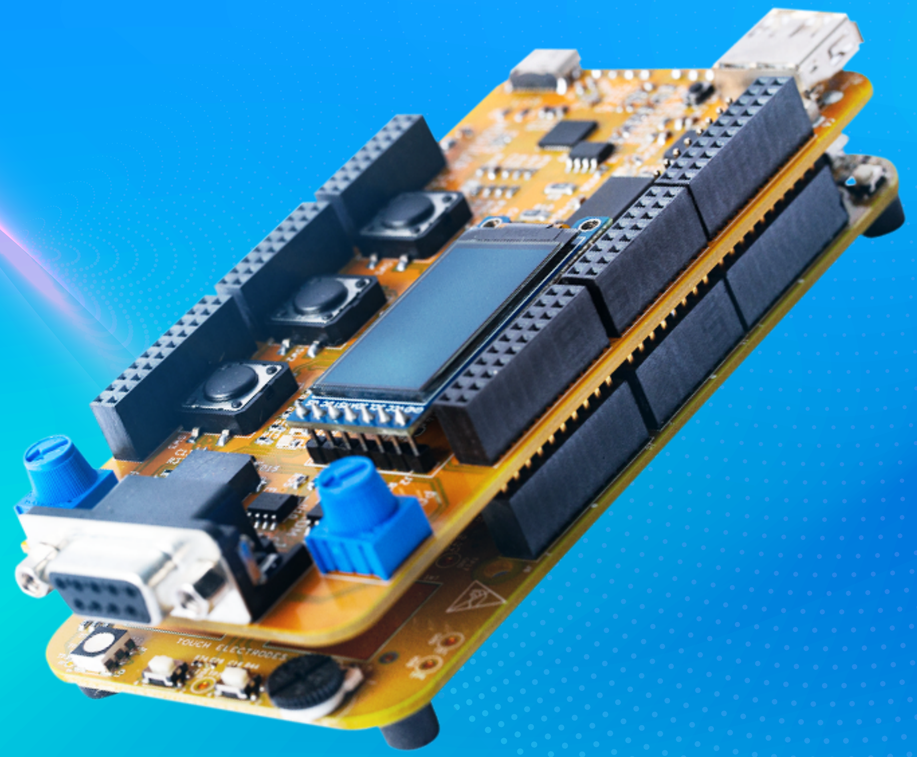
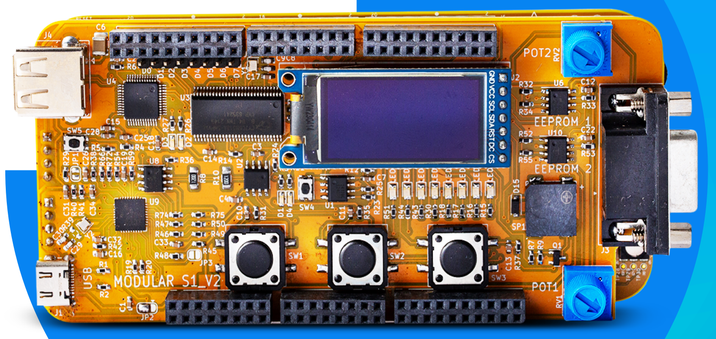
Automotive
Embedded Upskilling

AUTOSAR OSEK/VDX

With NXP S32K1xx and TrampolineRTOS



What is an operating system, and what distinguishes one of the OSEK type?



Terminology of Internal Concepts of an OSEK-type Operating System and How to Design an Application Based on Operating Systems

These are the concepts that participants will learn to understand and address over the course of 5 weeks through practical exercises, reference examples, theory, and a small project they will be required to complete

modular mx

Who is this for?

This training is aimed at Embedded Software Engineers with experience in using the C language for microcontrollers. It is applicable to those who already have experience with operating systems as well as to those who have not previously worked with one, whether it is of the OSEK type or not.

Introduction to OSEK/VDX

Before discussing operating systems, it is necessary to understand the platform we will be using, including both the hardware, which is the NXP S32K144-Q100 board, and the software, which in our case will be the preconfigured RTD drivers and TrampolineRTOS.

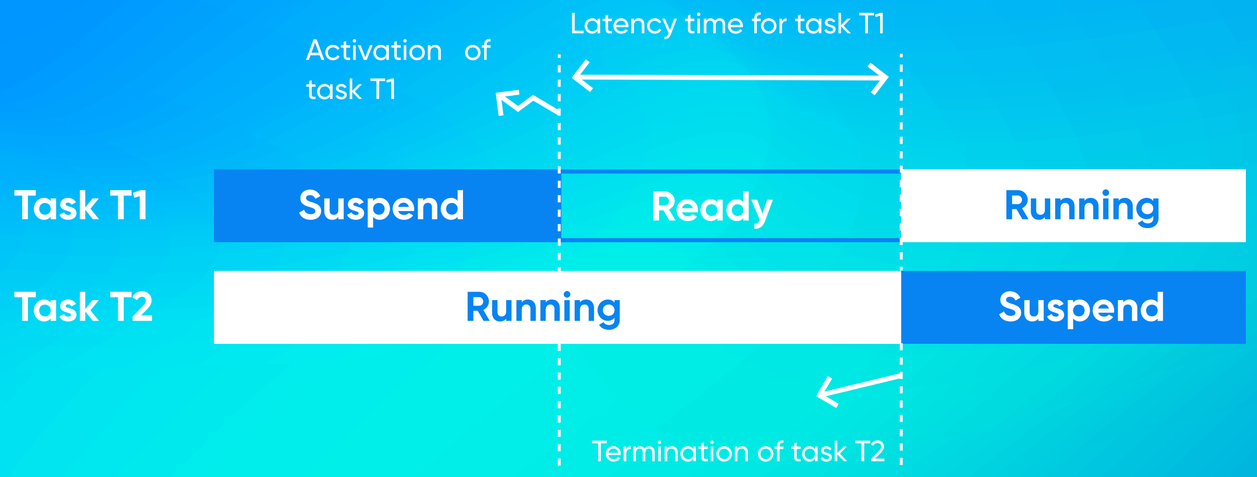
We will learn theoretical concepts such as scheduler, tasks, priorities, preemption, and semaphores. We will also cover the features that distinguish an OSEK/VDX-type operating system and its configuration parameters.

Task and Events

The foundation of any program with operating systems is tasks, and in this second part, we will learn how to define them in the OSEK operating system and configure their attributes such as priority, number of activations, and task type. We will also cover the concept of events and how events help us synchronize tasks.

Examples and initial exercises with the operating system will help reinforce these concepts.

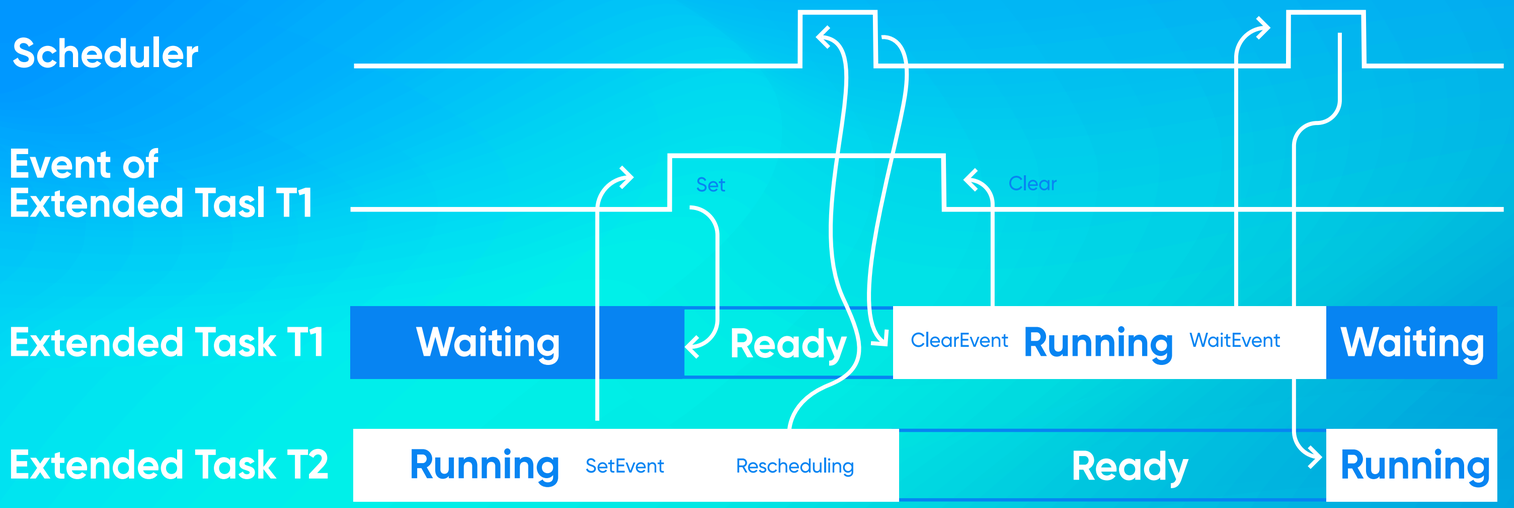
Inter-task Communication



Tasks can be considered as autonomous programs, but an application consists of multiple tasks interacting with each other. This week, we will explore the mechanisms of inter-task communication to coordinate the information exchanged between them

OSEK defines the concept of "message objects" for both receiving and transmitting between tasks. These are also used as notification mechanisms to establish events, invoke callbacks, and activate tasks

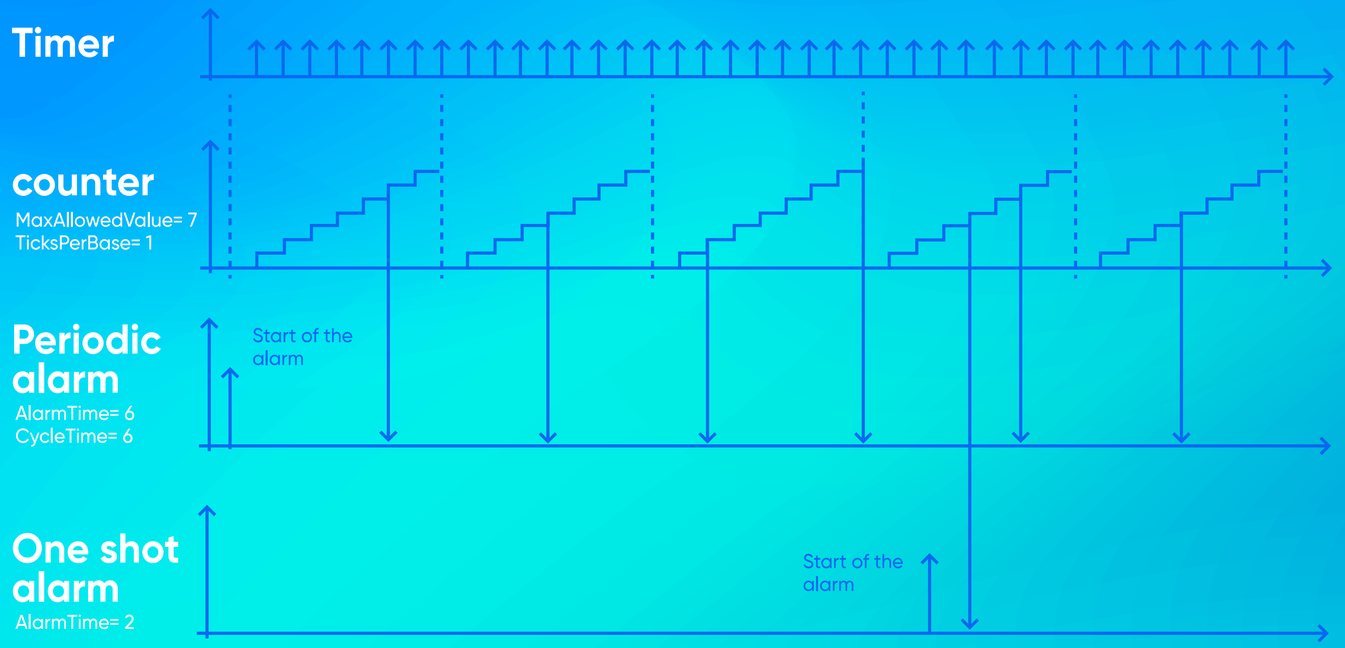
Tick Counter and Alarms



System tick configuration (Main Counter) to establish the time base for the entire system

OSEK defines the concept of timers as alarms, which are mechanisms used to determine task execution, establish events, or invoke callback functions. Additionally, we will explore how these alarms can be modified at runtime.

Resource Management and Interrupts



In a multitasking system, there are potential risks if more than one task tries to access the same resource. In this final week, we will learn how to manage system resources efficiently to prevent data corruption issues and how the OSEK Priority Ceiling protocol works for ensuring resource access control.

No system can operate without using interrupts for handling real-time external stimuli. We will also explore how to establish communication between interrupts and tasks that are waiting for messages.

Requirements and Materials

All sessions are virtual, so it doesn't matter where you are in the country or the world, as long as you have internet.

Computer with Windows
Internet Connection
Advanced Programming in C
Microcontrollers
Basic AUTOSAR

Material

2 remote sessions of 90 minutes each per week
24/7 access to material on Confluence, Tresos, and NXP + Modular-S1 card
Slack group with participants and alumni.

modular mx