PikeOS: End-to-End predictability for networked applications on multi-core avionics platforms

Sergey Tverdyshev, SYSGO AG
Don Kuzhiyelil, SYSGO AG
Overview

- PikeOS Introduction
- ARINC 653 Time Partitioning Scheduling
- Coupling of Time-Triggered Network and PikeOS Time Partitioning
  - Scheduling synchronization between node and network
PikeOS: Focus on Critical Systems

• Sicherheit [ˈʃɪçəhaɪt], noun
  1. as in „safety“: the condition of being safe from undergoing or causing hurt, injury, or loss; a device designed to prevent inadvertent or hazardous operation
  2. as in “security”: something that secures or protects; measures taken to guard against espionage or sabotage, crime, attack, or escape

• Certification for safety & security
  • DO-178B, EN50128, IEC61508, CC’s EAL, MILS ...
PikeOS in a Nutshell

• Hard Real Time
  • PikeOS is a hard real time operating system

• Safe And Secure Virtualization
  • PikeOS is a virtualization platform for safety and security critical systems

• Mixed Criticality
  • Applications with different safety and security levels can run on the same hardware, protected from each other by means of software partitioning

• Multiple Guest Operating Systems
  • Virtualization enables multiple Personalities (OS environments, APIs, run-time environments)

• Highly Portable
  • Supports all important CPU Architectures like x86, PowerPC, ARM, MIPS and Sparc

• Certifiable
  • Certifiable according to Highest Safety and Security Standards
  • Modular certification Kit for Safety Critical Avionics, Industrial Automation and Transportation Applications
Main Design Principles

• **Micro-Kernel Approach**
  • Limit the amount of code which runs in CPU Supervisor Mode

• **Strict Time and Space Partitioning to support “Mixed Criticality”**
  • All platform and operating system resources are assigned to Software Partitions based on a static configuration

• **Preemptive Design**
  • Kernel and System Software are fully preemptive to guarantee fast response time and simplify Worst Case Timing analysis

• **Support of different Guest Operating Systems (aka “Personalities”)**
  • Provide specialized services to support an efficient Guest OS implementation
  • Allow Guest Operating Systems to access all partition resources like memory, communication ports, files, interrupts, shared memory and I/O devices
  • Allow guest operating systems to implement their own devices drivers (e.g. Linux)
Multi-Functionality OS

Application Layer
- Standard Partitions based on different API

PikeOS Hypervisor
- Configuration
- Partition Management
- Partition Communication
- Health Monitoring
- Device Drivers
- OS primitives
- Platform Support Package
- Low Level Drivers
- Real-time

PikeOS System Software
- Driver (File API)
- Driver (Port API)
- Kernel Driver
- PikeOS PSP
- Boot Loader

PikeOS microkernel

Custom Application
- Linux
- Autosar
- Android

System Partition
- CBIT Health Mon. Logging

Hardware Platform
Mixed-Criticality OS

Application Layer
- Multiple Independent Levels of Security/Criticality

PikeOS Hypervisor
- Certified for Safety and Security

PikeOS – Mixed-Criticality OS

- Custom Application
  - Linux
- Custom Application
  - Autosar
- Custom Application
  - Android
- System Partition
  - CBIT
  - Health Mon.
  - Logging

PikeOS System Software

- Driver (File API)
- Driver (Port API)

PikeOS microkernel

- Low Level Driver
- PikeOS PSP

Boot Loader

Hardware Platform
Sharing Challenges

Challenge: Resources sharing

- Resources
  - CPUs
  - Memory, IO memory
  - Flies, drivers, devices, buses

- Safety
  - Integrity, availability
  - Isolation, application errors containment

- Security
  - Integrity, availability, confidentiality
  - Possible side channels via shared resources
  - Resources and API are attack surface

PikeOS Solution
Resource Partitioning

Challenge: Time sharing

- Time
  - CPU cycles
  - Time effects of accessing shared resources, e.g. buses

- Safety
  - Availability, deterministic behavior, meeting deadlines
  - Right balance between time- and event-triggered tasks

- Security
  - Availability, confidentiality
  - Possible timing side channels via shared resources, e.g. caches, busses
  - Time is the attack surface

PikeOS Solution
Time Partitioning
PikeOS Resource Partitioning

- Static allocation of all system resources
- Application has guaranteed access to assigned resources
- Applications cannot access resources of other partitions if not explicitly configured otherwise
- No error propagation throughout other partitions
- Memory protection enforcement using Hardware (MMU)
- All partitions execute in user mode
PikeOS Time Partitioning and Time-Triggered Networks: Synchronisation
PikeOS Time Partitioning

- Notion of Time Partitioning inherited from ARINC-653 standard
  - Simple time triggered scheduling method
  - Execution order and duration of applications can be statically configured
- Time (like other resources) is statically partitioned at configuration time
  - Repetition period of the time schedule is defined which is known as major time frame
  - Major time frame is divided into time partitions of varying length
  - All threads in the system are assigned to one such time partition
- Scheduling is done in two phases
  - Time partition scheduler makes a time partition active according to the defined time partition schedule
  - Within the active time partition, threads are schedules based on Priority based FIFO scheduling
PikeOS Time Partitioning

Traditional ARINC 653 Scheduling

- 5 ms for Part A
- 5 ms for Part B
- 7 ms for Part C
- 8 ms for Part D

Scheduling with Background Time Partition

- 9 ms for Part A
- 9 ms for Part B
- 7 ms for Part C

- 4 ms additional buffer for critical applications
- 6 ms additional buffer for non-critical partition in the normal case
Time-Triggered Network

- Applicable for networks that require to provide Quality of Service guarantees
- Network bus bandwidth is partitioned between participating nodes according to a static schedule
- Global supervising scheduler enforces this schedule on the network

- Similar to PikeOS time partitioning
  - Instead of threads, the network schedules network packets
  - Instead of allocating CPU time, network scheduler allocates network bandwidth to nodes
PikeOS and Time-Triggered Network

Usage of PikeOS with Time Triggered Networks:

1. ACROSS Project
   - Multicore platform in a Network On Chip (NoC) Architecture
   - Cores with no shared memory; communicates over the time triggered internal network (TTNoC)
   - No temporal interferences between PikeOS instances running on different cores

2. INTERESTED Project
   - PikeOS nodes in a TTEthernet
PikeOS and Time-Triggered Network

Similar concepts for Time-triggered network and PikeOS time partitioning

→ Synchronize the two time schedules

→ In a synchronized operation between nodes and network, PikeOS threads could be scheduled exactly after the packet arrival and just before packet sending

→ Efficient and easy system design

→ Ideally even without blocking on network interrupt
Coupling of Time-Triggered Network and Time Partitioning

- Harmonise the lengths of the network schedule and the major time frame of the time partitioning

- Synchronise the underlying clocks controlling the respective schedule, so that the scheduling in the two systems does not drift apart

- Ensure that the phases of the schedules are in synchronisation, i.e., establish a relationship between switching times of time partitions and transmission times of packets on the network.
Time-Triggered Networks: Two Kinds

1. Fully Hardware Synchronised NoC
2. External Time-Triggered Networks
Coupling of Time-Triggered Network and Time Partitioning

1. Fully Hardware Synchronised NoC
   - One clock for system and network
     → system and network are in perfect sync
     → guaranteed by hardware

   - Different cores in the system have different startup times
     → Phase correction is required to establish synchronous operation

   - Once synchronized, all cores run in sync with the network schedule
     → no resynchronization is required.
Perfect Sync Possible (NoC): Sync Once

- after boot, block and wait for sync
- immediately in sync
- keeps in sync
- same clock, no jitter

Operating system's time partitioning
Coupling of Time-Triggered Network and Time Partitioning

2. External Time-Triggered Networks

- Nodes start independently without waiting for the network
- OS clock different from network clock
  - phase synchronization is required between OS time partition schedule and network schedule
  - perfect synchronous operation cannot be reached
  - synchronous operation with a tolerable jitter
  - Perform resync when the jitter is above a threshold
- Network provides the master clock
  - OS time schedule shall synchronize with network clock
- Possibility of network break down
  - Resynchronization after the network is up again
External Clock: Permanent Resync

network schedule

send
receive

out of sync
allowable jitter
still out of sync
in sync!
in sync

TP1 TP 2 TP 3 TP 1 TP 2 TP 3 TP 1 TP 2 TP 3

operating system's
time partitioning

variable length time partition used for synchronisation

last stretch to sync
continuously kept in sync
Summary: ARINC Node and Network synchronisation

- PikeOS provides a safe and secure platform for building mixed-criticality systems
- PikeOS time partitioning can be synchronised to external clocks
- PikeOS time partitioning and time-triggered networks share a similar concept and therefore a synchronous operation leads to an easy and efficient system deployment
Summary: PikeOS Safety Multi-Core Certification

High-assurance certification on multicore hardware is possible

- PikeOS has been certified for EN 50128 SIL4 on a x86 multicore
  - SIL4 is the highest level
- EN 50128 SIL4 is comparable with ECSS-E-ST-40C / DO-178 Level A
  - For example, it requires MC/DC
Questions?