Secure Timeout System NXP S32K3X8EVB

Beamer for the CAOS Project

Andrea Botticella

Fabrizio Emanuel

Elia Innocenti

Renato Mignone

Simone Romano

January 24, 2025

Politecnico di Torino

Table of Contents

1 Project Overview

Project Overview

- Implement a secure timeout system application on the NXP S32K3X8EVB board using FreeRTOS, emulated with QEMU.
- Divided into several parts, each focusing on different aspects of the development process.

Table of Contents

2 Projects' Goal

Projects' Goal

- Learn how to use different technologies:
  - QEMU to emulate a heterogeneous hardware architecture.
  - FreeRTOS for real-time operating system functionalities.
- Learn how to use Git to manage a team project:
  - Efficient collaboration and version control.
- Learn how to present your work:
  - Documenting and presenting the project effectively.

Table of Contents

3 Part 1 - QEMU Board Emulation

Custom QEMU Version

3 Part 1 - QEMU Board Emulation

- Emulate the NXP S32K3X8EVB board, which is not natively supported by QEMU.
- Ensure proper emulation of the CPU, memory map, and peripherals.

Resources Used

3 Part 1 - QEMU Board Emulation

- Adding a new architecture to QEMU.
- Previous projects and repositories for reference.

Implementation Details

3 Part 1 - QEMU Board Emulation

- Board Initialization and Configuration:
  - Implement functions to load firmware, initialize memory regions, and handle hardware components.
  - Set up and configure hardware components like NVIC, LPUART, and PIT timers.
  - Manage system clocks and interrupts.

Memory Regions Initialization

3 Part 1 - QEMU Board Emulation

- Flash Memory:
  - Block0: Base Address: 0x00400000, Size: 2 MB
  - Block1: Base Address: 0x00600000, Size: 2 MB
  - Block2: Base Address: 0x00800000, Size: 2 MB
  - Block3: Base Address: 0x00AD0000, Size: 2 MB
  - Block4: Base Address: 0x10000000, Size: 128 KB
  - Utest: Base Address: 0x18000000, Size: 8 KB
- SRAM Memory:
  - Block0: Base Address: 0x20400000, Size: 256 KB
  - Block1: Base Address: 0x20440000, Size: 256 KB
  - Block2: Base Address: 0x20480000, Size: 256 KB
- DRAM Memory:
  - Base Address: 0x30000000, Size: 1 MB

Hardware Components Setup

3 Part 1 - QEMU Board Emulation

- NVIC (Nested Vectored Interrupt Controller):
  - Configured with 32 IRQs and 4 priority bits.
  - Connected to system clock and reference clock.
- LPUART (Low Power UART):
  - Base Address: 0x4006A000
  - Connected to NVIC and system clock.
- PIT Timers (Periodic Interrupt Timer):
  - Timer1: Base Address: 0x40037000
  - Timer2: Base Address: 0x40038000
  - Connected to NVIC and system clock.

System Clocks and Interrupts

3 Part 1 - QEMU Board Emulation

- System Clock:
  - Created clock object with 7.14ns period (140MHz frequency).
- Interrupt Handling:
  - Configured NVIC to handle interrupts.
  - Linked NVIC's memory access to system memory.

Firmware Loading

- Function: `s32k3x8_load_firmware`
- Parameters:
  - `cpu`: The ARM CPU instance.
  - `ms`: The machine state.
  - `flash`: The memory region representing the flash memory.
  - `firmware_filename`: The filename of the firmware to be loaded.
- Functionality:
  - Reads the firmware file and loads its contents into the specified flash memory region.

Initialization Functions

- `s32k3x8_initialize_memory_regions`:
  - Initializes flash, SRAM, and DRAM memory regions.
- `s32k3x8_init`:
  - Initializes the system, including memory regions, NVIC, LPUART, and PIT timers.

Table of Contents

4 Part 2 - FreeRTOS Porting

FreeRTOS on Emulated Board

4 Part 2 - FreeRTOS Porting

- Port FreeRTOS to run on the emulated NXP S32K3X8EVB board.
- Ensure compatibility and functionality of FreeRTOS on the emulated hardware.

Implementation Details

4 Part 2 - FreeRTOS Porting

- Kernel Configuration:
  - Configure FreeRTOS kernel settings in FreeRTOSConfig.h.
  - Define task priorities, stack sizes, and heap sizes.
  - Enable necessary FreeRTOS features like mutexes, semaphores, and task notifications.

Table of Contents

Secure Timeout System Application

5 Part 3 - Write a Simple Application

- Implement a simple application with multiple tasks to demonstrate the setup.
- Tasks include monitoring user activity, handling alerts, and simulating events.
- Use hardware timers for periodic operations.

Implementation Details

5 Part 3 - Write a Simple Application

- Task Implementation:
  - Monitor Task: Detects user activity and logs it.
  - Alert Task: Detects suspicious activity and logs it.
  - Event Task: Simulates user and suspicious activities periodically.
- Hardware Timer Initialization:
  - Initialize hardware timers to generate periodic interrupts.
  - Implement interrupt handlers to detect activities.

Table of Contents

6 Conclusion

Conclusion

6 Conclusion

- The s32k3x8evb_board.c file plays a crucial role in the emulation of the NXP S32K3X8EVB board within QEMU.
- It provides the necessary functions to load firmware, initialize memory regions, set up hardware components, and manage system clocks and interrupts.
- This detailed analysis highlights the key functionalities and their implementations, providing a comprehensive understanding of the board initialization and configuration process.

Thank you for listening!
Any questions?