Master-Slave State Synchronization Using freeRTOS

Contents

[1. Prerequisites 1](#_Toc185764715)

[ A toolchain for compiling and editing a project is realized with help of 1](#_Toc185764716)

[2. Solution short description 2](#_Toc185764717)

[3. Sequence diagram of Master/Slave 3](#_Toc185764718)

[4. ESAT\_MasterSlave files description 4](#_Toc185764719)

[5. Additional resources 4](#_Toc185764720)

[ ENDUROSAT Technical Task version 1.0.0 – Master-Slave State Synchronization Using freeRTOS 4](#_Toc185764721)

[ This task solution has two resources. 4](#_Toc185764722)

[6. How to run demo 4](#_Toc185764723)

# Prerequisites

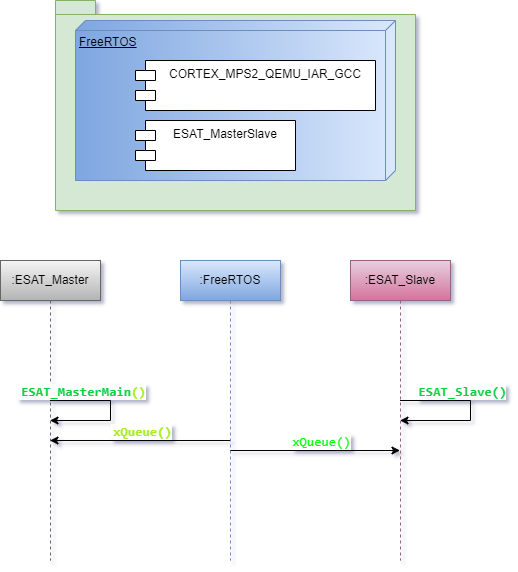
Project is realized over Windows host system

## A toolchain for compiling and editing a project is realized with help of

* + - MinGW
    - Arm-none-eabi (gcc-arm-none-eabi-10.3-2021.10-win32)
    - Embedded OS version - FreeRTOSv202212.01 (DEMO CORTEX\_MPS2\_QEMU\_IAR\_GCC
    - QEMU simulation program for ARM (qemu-w64-setup-20241124.exe)
    - Project is realized over Eclipse IDE and FreeRTOSv202212.01
    - Git Bash support

To run a demo, without IDE and debugger, only QEMU and Git Bash are required.

# Solution short description



At picture 1 a delivery diagram and simplified sequence diagram is shown.

Task solution consist of two parts:

* ESAT\_MasterMain(Device A)
* ESAT\_Slave(Device B)

Master (ESAT\_MasterMain) frequently requests states of Slave (ESAT\_Slave). According to current Slave state, Master prints status on console, or request additional details when slave is in ACTIVE state.

If Slave face ERROR, it logs it and Master send command to reset Slave condition.

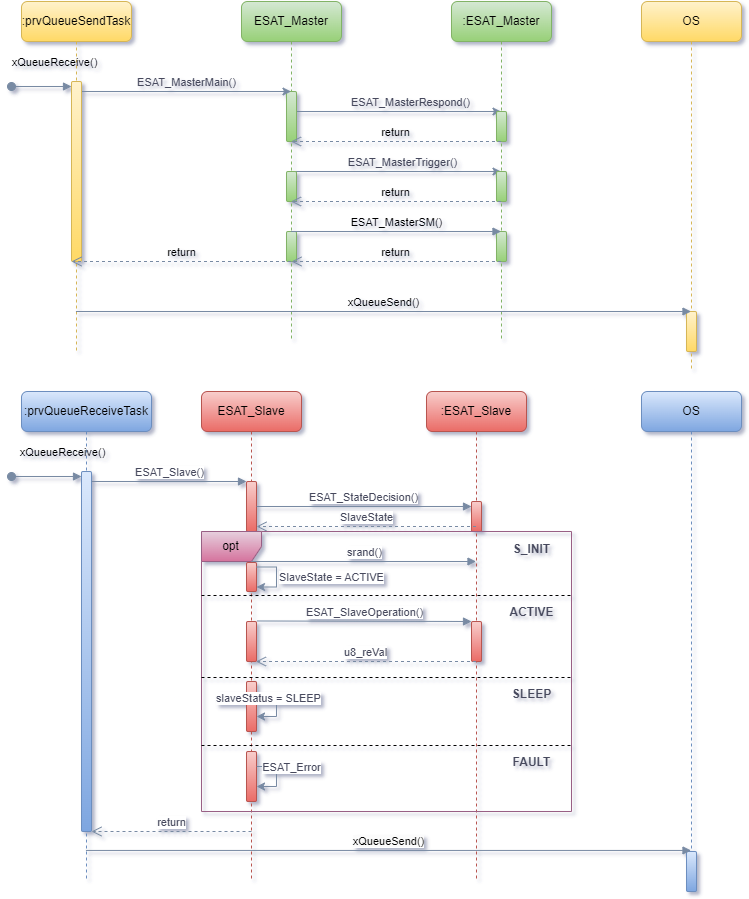
For changing Master/Slave states a pseudo-random generator is used.

Two tasks are configured in FreeRTOS and communication between Device A/Device B is realized across freeRTOS Queue mechanism.

This task is based on QEMU simulation, and NO additional hardware required.

# Sequence diagram of Master/Slave

Sequence diagram of Master and Slave components show internal work sequence of each side.



At picture 2 a detailed sequence diagram is shown. Master component is hosted on freeRTOS Send Task and Slave component operate in freeRTOS Receive Task.

# ESAT\_MasterSlave files description

* + ESAT\_Master.c/ ESAT\_Master.h – describe Device A (Master) part.
  + ESAT\_Slave.c/ ESAT\_Slave.h – describe Device B (Slave) part.
  + ESAT\_MS\_Types.h – describe used types and actions used from devices.

# Additional resources

## ENDUROSAT Technical Task version 1.0.0 – Master-Slave State Synchronization Using freeRTOS

## This task solution has two resources.

* + - * + Mail based – mail details are sent to author of the task.
        + Git based additional information could be found at GitHub ( <https://github.com/denimlm/FreeRTOS_ESAT_MasterSlave.git> )

# How to run demo

* + - * Git resources required to be downloaded. Additional information is provided in repo ( <https://github.com/denimlm/FreeRTOS_ESAT_MasterSlave.git> )
      * Mail demo consist from zip that contain precompiled output file, bash starting script and QEMU folder (QEMU installation should be placed tin this folder to start demo).
        + To start Demo, just unzip zip file, go to unzipped folder (ESAT\_MasterSlave), copy QEMU installation in folder qemu/, open Git Bash console from location of StartDemo file and type command (./StartDemo run ) in it.