Lab #4
⋆
Internal communication

November 13, 2012

Note: All the software and documents are stored at `http://www.irccyn.ec-nantes.fr/~bechenne/trampoline`

1 Overview

Internal messaging may be used as an easy to use replacement for shared variables. Messaging allows to send data from one task to one or many task. This lab will show the different ways to communicate in OSEK. Related course spans from slides 96 to 109. Go into the `labs/lab4` directory. It contains an application with 2 communicating tasks. Examine the lab4.oil and the lab4.c files. The OIL file declare 2 tasks: `sender` and `receiver` and 2 messages: `outMessage` and `inMessage`. `inMessage` is connected to `outMessage`. Task `sender` uses `outMessage` and task `receiver` uses `inMessage`. Task `sender` is activated every second by the mean of an alarm and sends data to `outMessage`. When the data is received in `inMessage`, the notification mechanism activates task `receiver`. `receiver` reads the data from `inMessage` and prints the value.

2 Message broadcasting

OSEK messaging supports *one-to-many* communication. This is done by having more than one receiving messages connected to the same sending message.

Question 1 *Add 2 other receiving messages connected to* `outMessage`*. The first message activates the task* `Emergency` *(add the corresponding task), the second message sets an event to task* `NormalOperation` *(add the corresponding extended tasks too). Write the C code of the tasks you added. The first one receives the message and prints the value; the second one waits for the event in an infinite loop, receives the message and prints the value. Check the application works as expected.*

3 Filtering

Look at the filtering principles (slides 93 to 95).

Question 2 *Add a filter to activate task* `receiver` *upon the 4th message and then every 3 values (ie 4th, 7th, 11st and so on).*

Question 3 *Modify task* `sender` *to send a random value ranging from 0 to 100. Using filtering, set the event to task* `NormalOperation` *only if the value is within the range* $[20, 60]$*; activate task* `Emergency` *only if the value is not within the range.*

4 Queued messages

Look at slide 89

Question 4 *Restarting with the Messaging application, modify the* `inMessage` *to make it a queued message with a size of 4 and remove the notification. Add an alarm to activate task receiver every 4 seconds. Check the return value of* `ReceiveMessage` *and verify that no error occurred (queue under- or over-flow). If errors occur, correct the application.*