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 Goal

Internal messaging may be used as an easy to use replacement to shared variables. Messaging allow to send data from a 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 having 2 tasks that communicate. 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 executed every second by the mean of an alarm and send a data using `outMessage`. When the message is received in `inMessage`, the notification activates task `receiver`. `receiver` receive the message from `inMessage` and print the value.

2 Message broadcasting

OSEK messaging allows the 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 receive the message and prints the value, the second one waits for the event in an infinite loop, receive the message and prints the value. Check the application works as expected.*

3 Filtering

Look at the filtering principles (slides 106 to 109).

Question 2 *Add a filter to activate task* `receiver` *for the 4th 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* $[20, 60]$ *range and activate task* `Emergency` *only if the value is not within the range.*

4 Queued messages

Look at slide 102

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 no error occurred (queue under- or over-flow). If errors occur, propose a modification to correct it*