Intertask Communication

I had to update the example slightly to make it work for Xenomai 3, so use what's in this folder to start off with.

Part A

Part 1

Here's my output:

```
xenomai@ieu:~/exercises/intertask_comm$ ./ex05a_1
start task  : 1
taskOne sent message to mailbox
taskOne received message: Message from taskOne
with length 21
start task  : 2
taskTwo sent message to mailbox
taskTwo received message: Message from taskTwo
with length 21

Type CTRL-C to end this program
```

Here you can see that each task got its own message. This is because task 1 and task 2 write a message then immediately read from the mailbox, and as you can see from the output, task 1 starts and executes before task 2 starts and executes. I believe this is related to how the mailbox was created:

```
rt_queue_create(&myqueue,"myqueue",QUEUE_SIZE,10,Q_FIFO);
```

Q_FIFO makes tasks pend in FIFO order on the queue for consuming messages. Since both tasks are using the same mailbox, then first in, first out for task 1 and 2.

Part 2

Here's my output:

```
xenomai@ieu:~/exercises/intertask_comm$ ./ex05a_2
start task  : 1
taskOne sent message to mailbox
start task  : 2
taskTwo sent message to mailbox
taskTwo received message: Message from taskOne
with length 21
taskOne received message: Message from taskTwo
with length 21

Type CTRL-C to end this program
```

Here, each task got the message intended for it. Once task 1 sends its message to task 2's mailbox and waits to receive, task 2 starts and sends its message to task 1's mailbox. Task 2 then reads its mailbox, gets the message, and finishes executing. Task one then resumes and reads its mailbox.

Part 3

I used two mailboxes for this.

```
xenomai@ieu:~/exercises/intertask_comm$ ./ex05a_3
start task  : 1
taskOne sent message to mailbox
taskOne sent message to mailbox
taskOne sent message to mailbox
start task  : 2
taskTwo received message: Message from taskOne
with length 21
taskTwo received message: Message from taskOne
with length 21
taskTwo received message: Message from taskOne
with length 21
taskTwo sent message to mailbox
taskTwo sent message to mailbox
taskTwo sent message to mailbox
taskOne received message: Message from taskTwo
with length 21
taskOne received message: Message from taskTwo
with length 21
taskOne received message: Message from taskTwo
with length 21
```

```
Type CTRL-C to end this program
```

I found that I can send all the messages I wanted to a mailbox before they were read.

Part B

Part 1

Here's my output

```
xenomai@ieu:~/exercises/intertask_comm$ ./ex05b_1
start task  : 1
start task  : 2
taskTwo received message: Message from taskOne
with length 21
taskOne sent message

Type CTRL-C to end this program
```

In order for this to work, send in task 1 needs to have a timeout of TM_INFINITE while receive can have either a timeout of TM_INFINITE or TM_NONBLOCK. Task 1 is started first and must become blocked so task 2 can start and be blocked as well (blocked as in ready to receive the message from task 1). If send has TM_NONBLOCK, it returns EWOULDBLOCK meaning that the recipient task (task 2) is not blocked (as in not waiting for message).

The output is a little misleading with "taskOne sent message" following "task Two receiver …". The message is still sent first before its read, obviously, it's just the confirmation for task 1 is delayed slightly by task 2. Everything pretty much happens instantaneously.

Part 2

You don’t really have to try to know that you cannot send multiple different messages before reading them with this method. This method relies on a task waiting for the other to either receive or send message between the tw. Means you can’t pile up messages in a “mailbox” like the queue method.

I put things in for loops and made it so that a different message is sent to task 2 each iteration. Here’s my output:

```
xenomai@ieu:~/exercises/intertask_comm$ ./ex05b_2
start task  : 1
start task  : 2
taskTwo received message: Message from TaskOne (0)
with length 30
taskTwo received message: Message from TaskOne (0)
with length 30
taskTwo received message: Message from TaskOne (0)
with length 30
taskOne sent message
sleep error: -43
taskOne sent message

Type CTRL-C to end this program
```

So task 1 sends a message and gets blocked waiting for task 2 to read. Task 2 starts, receives the same message 3 times, and finishes executing. Task 1 then resumes, confirms that the first message was sent then tries again. This time, however, send functions returns EIDRM which means that the recipient task has been deleted waiting for a reply.

Part C

Part 1

My output is

```
xenomai@ieu:~/exercises/intertask_comm$ ./ex05c_1_read
Message received: Message from real-time task
```

If you cannot get any rtp nodes to open with open(), then make sure /dev/rtp* are under “xenomai” group and their permissions are 760

Part 2

I’m not doing it because it is possible. Essentially, you just get a backlog of messages like the message queue method. Just keep reading fd and eventually you’ll clear out the queue.

Part D

Message queues are great if you need to send data to another task without feedback. So, I would say a good application would be for data sent to the task responsible for creating telemetry packets, writing files, etc…

Synchronous messaging is good if you are trying to synchronize tasks to accomplish a goal because it’s only one message and one of the tasks is depended on the other. A possible application could be command execution?

Message pipes will be useful for integrating image processing software. I see a task passing image data to the ips software and getting a probability as a return message. Then, it decides whether to pass along the image to get put into telemetry packets. Something like that?