Inter-Process Communication and Process Management

Alexander Einshoj

April 13, 2016

Introduction

This project is divided into two parts; a message passing mechanism, and the other is process management.

Design

The main issue of the message passing part of the assignment, is the bounded buffer, and how to solve it. It's a concept that might be hard to grasp when being explained, but looking at it graphically, it makes a lot of sense. There are only two possibilities with the buffer. Data either goes in, or out.

When one wants to receive data, you have to know whether there is any data to receive. If not, it's not possible. If you want to send data, you need to know the receiving buffer has room for it. If this isn't the case, you'll simply have to wait in line.

Discussion

In this assignment, I have only implemented what was required, without having time left for extra credits of any kind. If I were to implement one or more of the extra credit choices, I would definitely begin with the PS command. This is due to PS being a standard command that I'm familiar with, and I know how the end result is going to look, with a good idea of how to get there. There were some things that could have been done differently in the current implementation to make it look better for further debugging. The though is making the functions look more compact by creating helper functions which each has a small task, to support the existing ones. In the current mailbox implementation I use condition broadcast instead of condition signal. This can be viewed upon as my solution to the consumer producer problem. As there is really no knowledge about the size of the messages or how much space is left in the buffer(could check this, but no point as we don't know the size of the messages), one might as well broadcast the fact that there is free space, giving everyone else the opportunity to send messages. As stated in the assignment itself, condition variables are in part used to solve this issue, and there's not too many ways to do it, but this seemed like the intuitive

choice in my opinion.

The keyboard implementation is based purely on the already implemented mailboxes. One of the only things to be aware of about these functions, is that the correct letter is sent or received, as well as there being enough space in the receiving buffer to send the message.

The process management part was by far the most confusing part to work with, as it included the use of several pre-defined functions from the precode which use weren't necessarily obvious. What took me the most time during the creation of these functions were definitely where to look for the information I wanted, where to put it, and using the pre-defined functions correctly.

Conclusion

Considering the performance of the resulting code, I'd say it works as expected and functions as required. I'm not entirely satisfied with not having done the PS command, as I can imagine it would feel good to implement a command which actually exists and is relatively widely used. The code is well commented, should be easy both to read and understand. As for what I would have done differently, it would be to use the given files more to know what the result should look like. I didn't do this for the same reason I don't look at the answer to a maths question while I'm solving it: It creates more confusion than it helps me find the solution. The lesson learnt being, programming differs from mathematics in this aspect.