

# How to communicate between processes/threads?

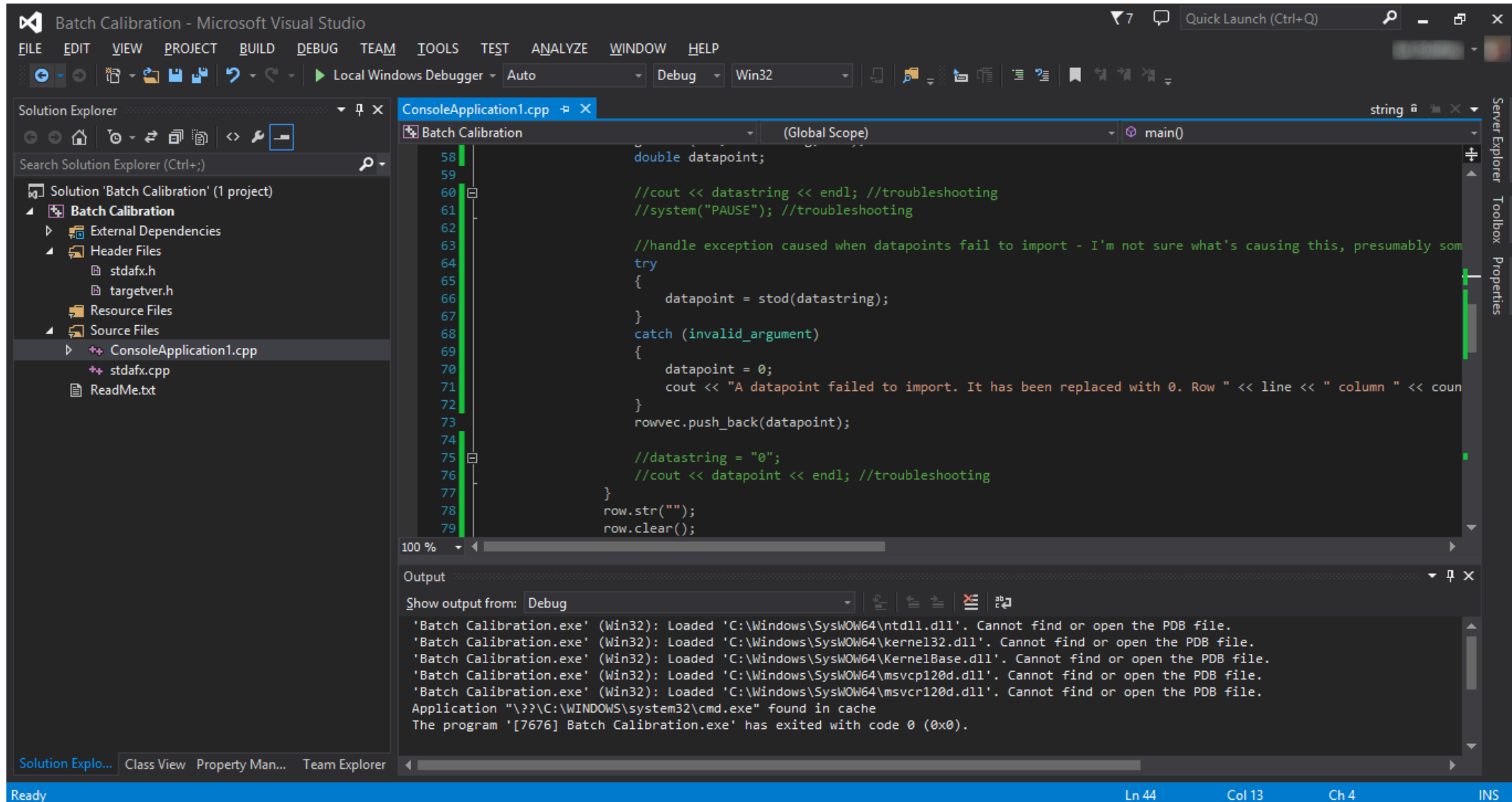
Prof. Yung-Pin Cheng

2017 03

# Timing to communicate between processes and threads?

- In practice, always packing your code (assume large enough) into a monolithic chunk of program (i.e., an .exe) is not a good choice in many cases.

# Visual Studio



# Visual Studio

- You may think Visual Studio is a big .exe
- Actually its code only contains GUI + editor
- When you hit “build” it actually pass the your compiler options/linking options to Cl.exe, link.exe

# Visual Studio

Microsoft Visual Studio 14.0

檔案 常用 共用 檢視

← → ↶ ↷ > 本機 > Windows (C:) > Program Files (x86) > Microsoft Visual Studio 14.0 >

★ 快速存取

- 桌面
- 下載
- 文件
- 圖片
- ADLINK
- ITRI-KORAT
- lab dump file - keyboard war III - 0327
- lab multithreading safe
- slides
- transfer-2017

OneDrive

本機

- 下載
- 文件
- 音樂
- 桌面
- 圖片
- 影片

Windows (C:)

DATA (D:)

Expansion Drive (E:)

Expansion Drive (E:)

名稱	修改日期	類型	大小
Common7	2016/3/31 上午 1...	檔案資料夾	
DesignTools	2016/9/4 下午 09...	檔案資料夾	
DIA SDK	2016/9/4 下午 09...	檔案資料夾	
ImportProjects	2016/3/31 上午 1...	檔案資料夾	
JavaScript	2016/9/4 下午 09...	檔案資料夾	
lib	2016/9/4 下午 09...	檔案資料夾	
Licenses	2016/9/4 下午 09...	檔案資料夾	
PreEmptive Solutions	2016/3/31 上午 1...	檔案資料夾	
professional	2016/9/4 下午 08...	檔案資料夾	
SDK	2016/3/31 上午 1...	檔案資料夾	
Setup	2016/3/31 上午 1...	檔案資料夾	
shell	2016/9/4 下午 09...	檔案資料夾	
Silverlight	2016/3/31 上午 1...	檔案資料夾	
Team Tools	2016/3/31 上午 1...	檔案資料夾	
VB	2016/3/31 上午 1...	檔案資料夾	
VC	2016/9/4 下午 09...	檔案資料夾	
VC#	2016/3/31 上午 1...	檔案資料夾	
Visual Studio Tools for Office	2016/3/31 上午 1...	檔案資料夾	
Web	2016/3/31 上午 1...	檔案資料夾	
Xml	2016/9/4 下午 08...	檔案資料夾	
extensions.configurationchanged	2016/9/22 上午 0...	CONFIGURATIO...	0 KB
Microsoft.VisualStudio.Authentication....	2016/5/25 下午 0...	Microsoft Visual ...	1,456 KB
Microsoft.VisualStudio.ConnectedServ...	2016/5/25 下午 0...	Microsoft Visual ...	13,148 KB
Microsoft.VisualStudio.ConnectedServ...	2016/6/15 下午 0...	Microsoft Visual ...	486 KB
Microsoft.VisualStudio.MobileServices...	2016/5/25 下午 0...	Microsoft Visual ...	9 KB

# Visual Studio

檔案 常用 共用 檢視

← → ↕ ↑ > 本機 > Windows (C:) > Program Files (x86) > Microsoft Visual Studio 14.0 > VC > bin >

★ 快速存取

- 桌面
- 下載
- 文件
- 圖片
- ADLINK
- ITRI-KORAT
- lab dump file - keyboard war III - 0327
- lab multithreading safe
- slides
- transfer-2017

OneDrive

本機

- 下載
- 文件
- 音樂
- 桌面
- 圖片
- 影片

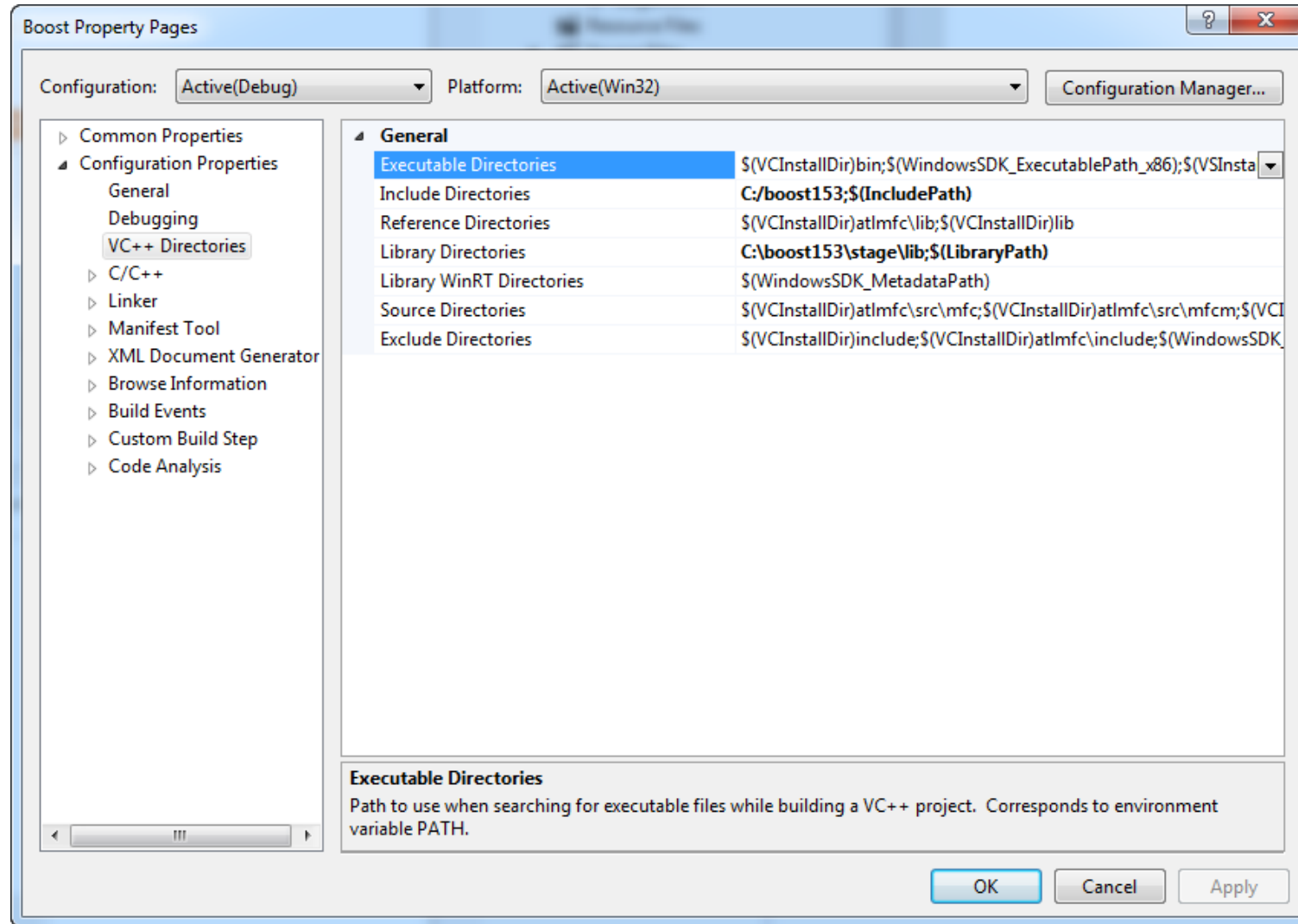
Windows (C:)

- DATA (D:)
- Expansion Drive (E:)
- Expansion Drive (E:)

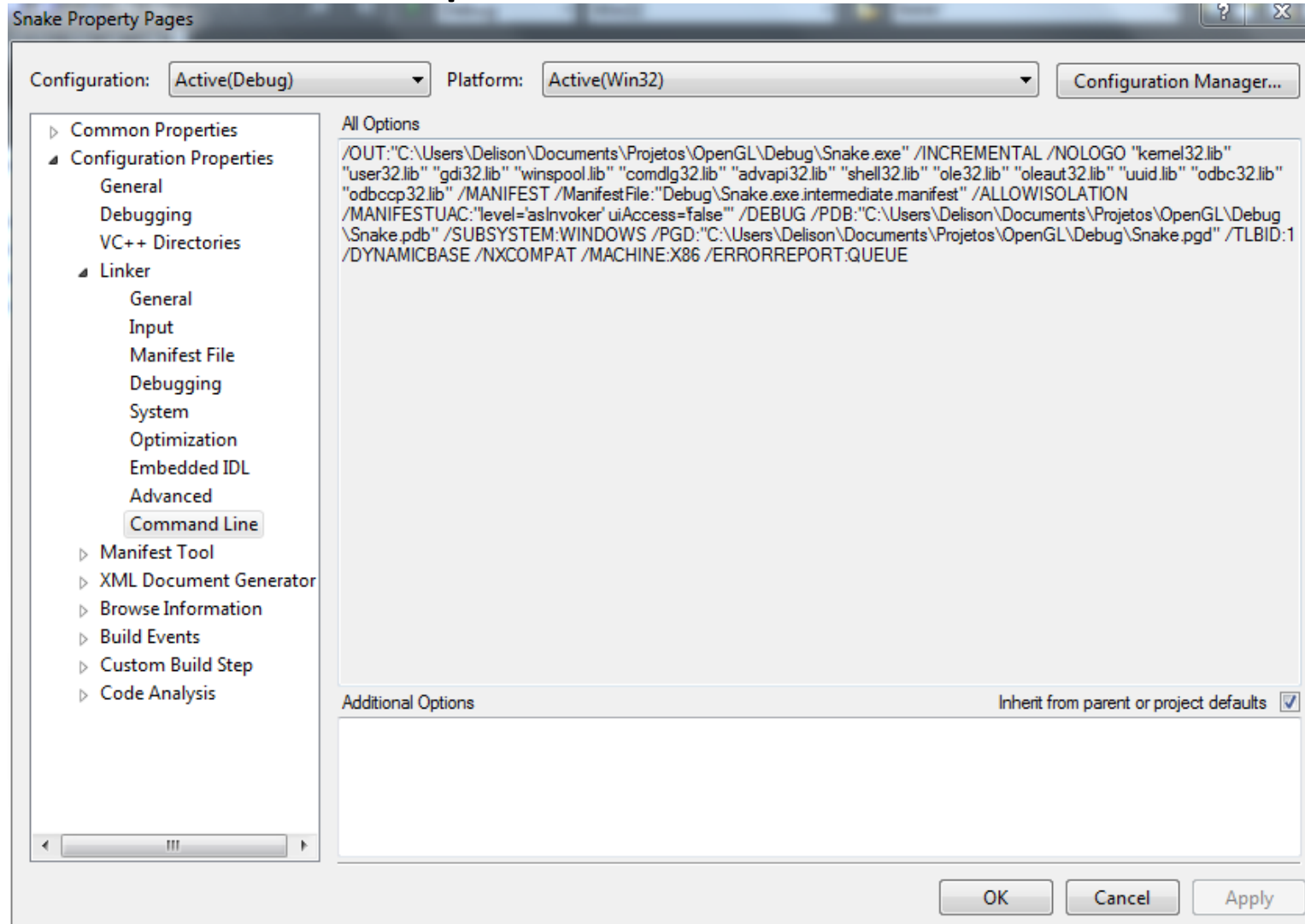
網路

名稱	修改日期	類型	大小
1028	2016/9/4 下午 09...	檔案資料夾	
1033	2016/9/4 下午 09...	檔案資料夾	
amd64	2016/9/4 下午 09...	檔案資料夾	
amd64_arm	2016/9/4 下午 09...	檔案資料夾	
amd64_x86	2016/9/4 下午 09...	檔案資料夾	
arm	2016/9/4 下午 09...	檔案資料夾	
x86_amd64	2016/9/4 下午 09...	檔案資料夾	
x86_arm	2016/9/4 下午 09...	檔案資料夾	
vcvars32.bat	2016/6/9 下午 10...	Windows 批次檔案	7 KB
vcvarsphoneall.bat	2016/6/9 下午 10...	Windows 批次檔案	1 KB
vcvarsphonex86.bat	2016/6/9 下午 10...	Windows 批次檔案	5 KB
cl.exe.config	2016/6/9 下午 10...	XML Configurati...	1 KB
link.exe.config	2016/6/9 下午 10...	XML Configurati...	1 KB
xdcmake.exe.config	2016/6/9 下午 10...	XML Configurati...	1 KB
bscmake.exe	2016/6/9 下午 10...	應用程式	91 KB
cl.exe	2016/7/21 下午 1...	應用程式	187 KB
cvtres.exe	2016/6/9 下午 10...	應用程式	48 KB
dumpbin.exe	2016/6/9 下午 10...	應用程式	27 KB
editbin.exe	2016/6/9 下午 10...	應用程式	27 KB
ifc.exe	2016/6/9 下午 10...	應用程式	163 KB
lib.exe	2016/6/9 下午 10...	應用程式	27 KB
link.exe	2016/7/21 下午 1...	應用程式	972 KB
ml.exe	2016/6/9 下午 10...	應用程式	438 KB
mspdbscmf.exe	2016/6/9 下午 10...	應用程式	965 KB
mspdbsrv.exe	2016/6/9 下午 10...	應用程式	135 KB
nmake.exe	2016/6/9 下午 10...	應用程式	104 KB
pgocvt.exe	2016/6/9 下午 10...	應用程式	62 KB
pgomgr.exe	2016/6/9 下午 10...	應用程式	91 KB
pgosweep.exe	2016/6/9 下午 10...	應用程式	67 KB
undname.exe	2016/6/9 下午 10...	應用程式	30 KB
vctip.exe	2016/6/9 下午 10...	應用程式	879 KB
xdcmake.exe	2016/6/9 下午 10...	應用程式	48 KB

# Compiler/Linker Options in Visual Studio



# Command Line options





# Why?

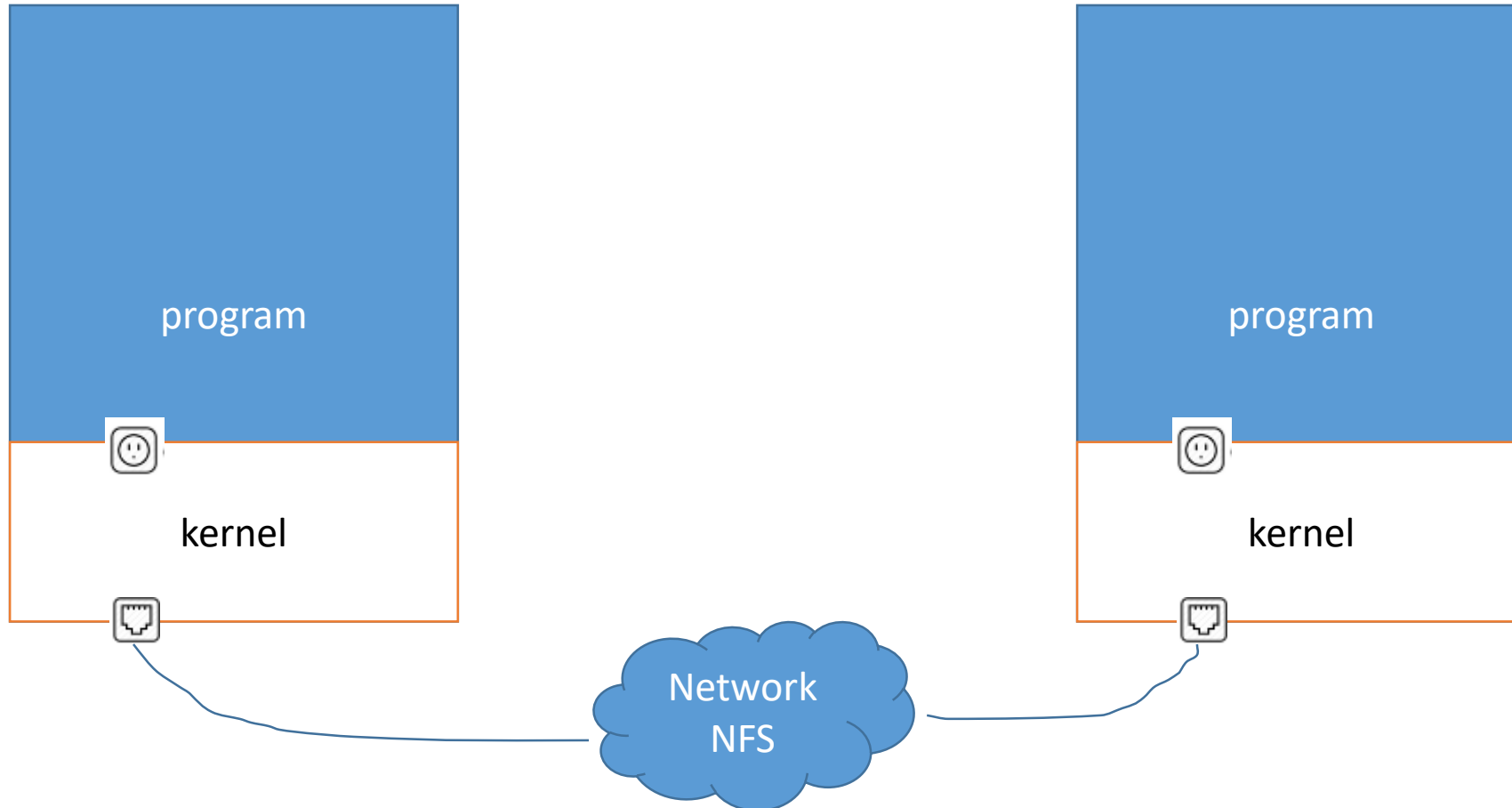
- A large program is usually hard to maintain and extend.
- If you include C compiler code in Visual Studio, every time C compiler code change, all Visual Studio code (million lines) need to be recompiled, debugged, and tested.
- So, divide a large-scale program into several small programs is a common design when performance issues are not compromised. This is called decoupling.
- Each program is self-contained, smaller, and more manageable.

To break or not to break !!  
That is the question.

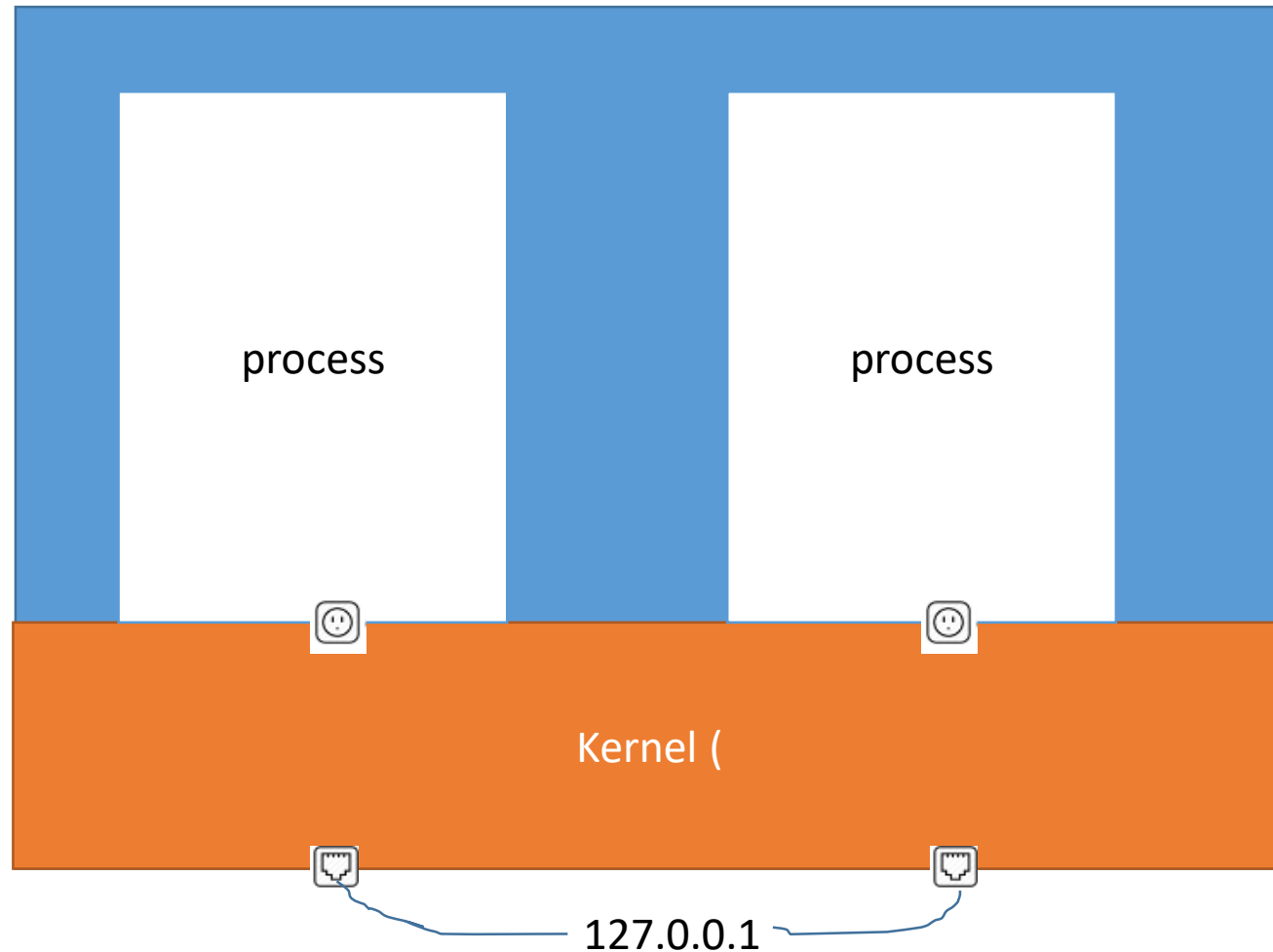
# Break it !!

- In case you break it. You will find you often need to make two programs (processes) communicate.

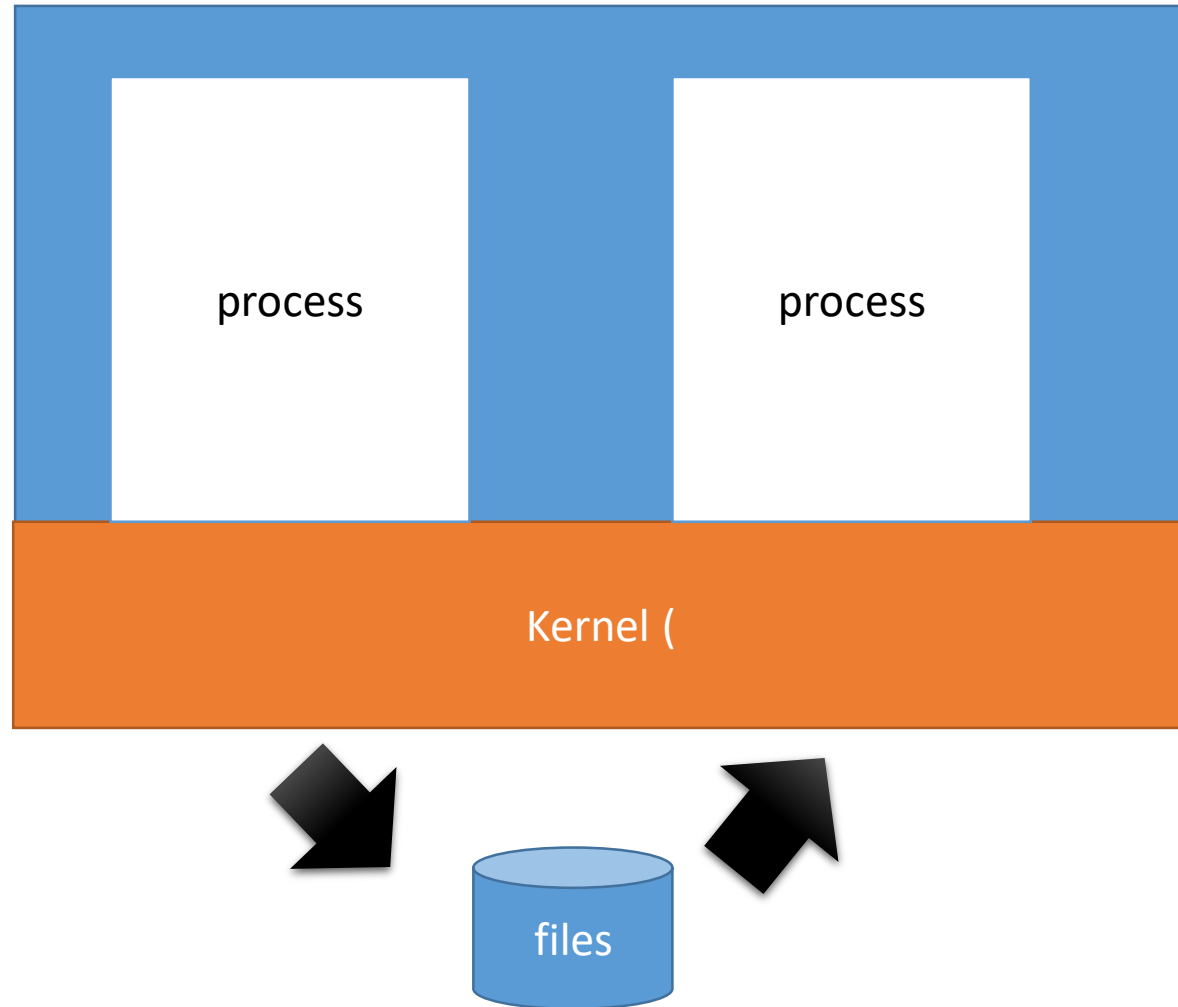
# Cross Computer Communication - Socket



# Programs in the same computer - socket



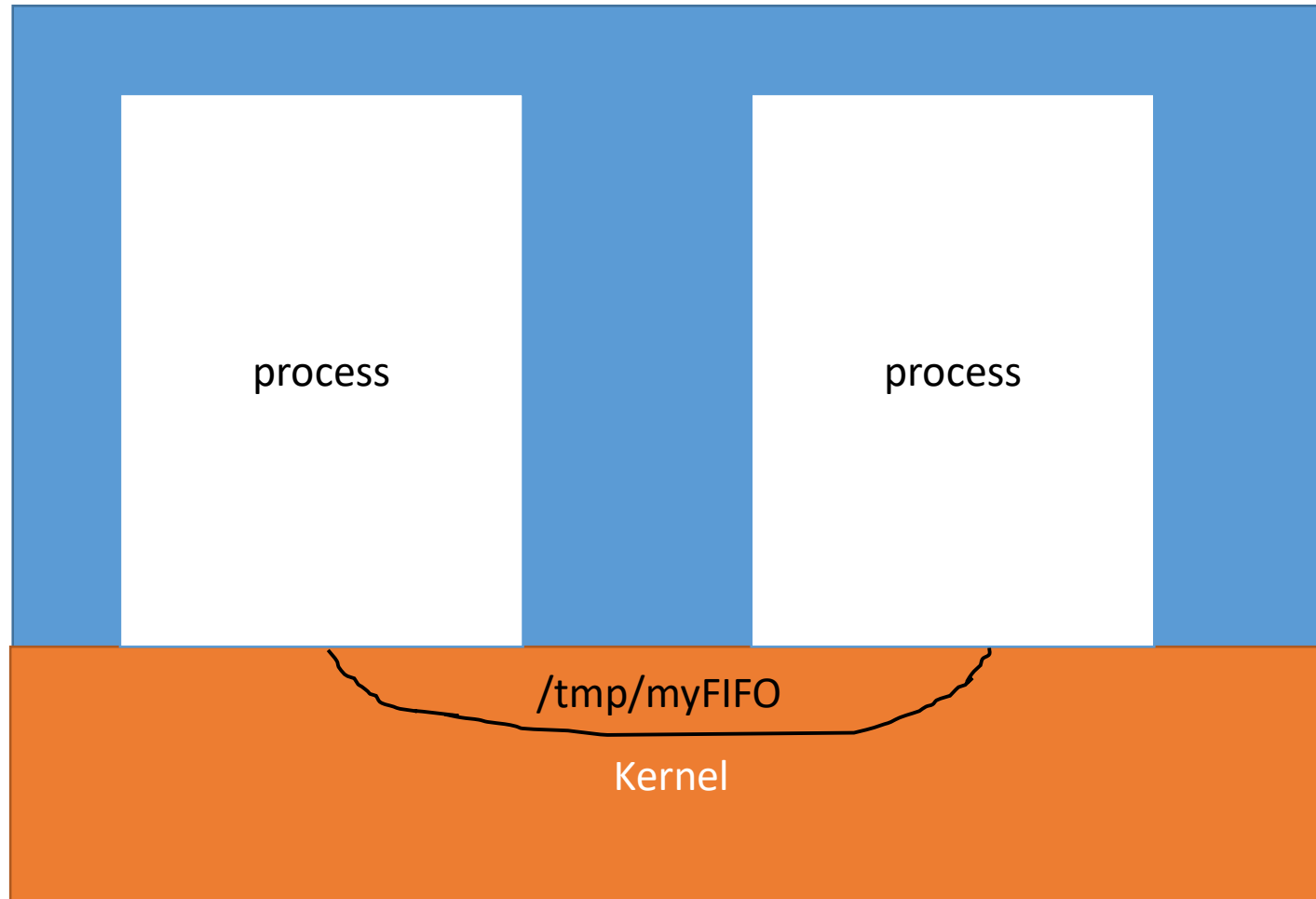
# Programs in the same computer – files



# Synchronization by files

- You may want to have a program read the file at then have a program to write (append?) at the same time. Message is passed by file.
- Don't try it !! However, you are welcome to try it to understand why
- If you really want to pass info by files
  - Use other communication technique to notify another program that the file is ready for reading or ready for writing
  - Make sure you flush the file before doing so

# Programs in the same computer – named pipes





# Linux named pipe

## writer.c

```
#include <fcntl.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

int main()
{
    int fd;
    char * myfifo = "/tmp/myfifo";

    /* create the FIFO (named pipe) */
    mkfifo(myfifo, 0666);

    /* write "Hi" to the FIFO */
    fd = open(myfifo, O_WRONLY);
    write(fd, "Hi", sizeof("Hi"));
    close(fd);

    /* remove the FIFO */
    unlink(myfifo);

    return 0;
}
```

## reader.c

```
#include <fcntl.h>
#include <stdio.h>
#include <sys/stat.h>
#include <unistd.h>

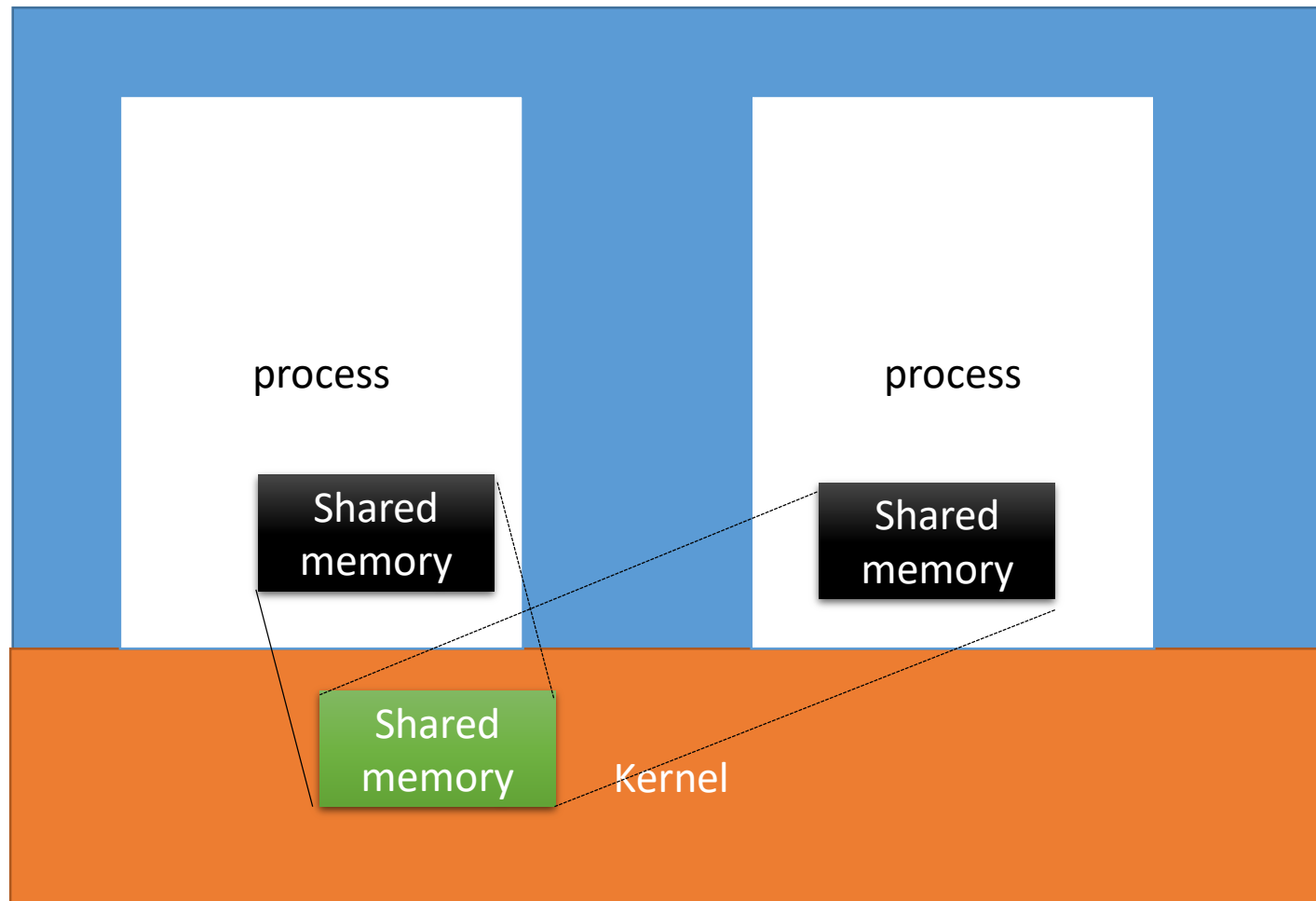
#define MAX_BUF 1024

int main()
{
    int fd;
    char * myfifo = "/tmp/myfifo";
    char buf[MAX_BUF];

    /* open, read, and display the message from the FIFO */
    fd = open(myfifo, O_RDONLY);
    read(fd, buf, MAX_BUF);
    printf("Received: %s\n", buf);
    close(fd);

    return 0;
}
```

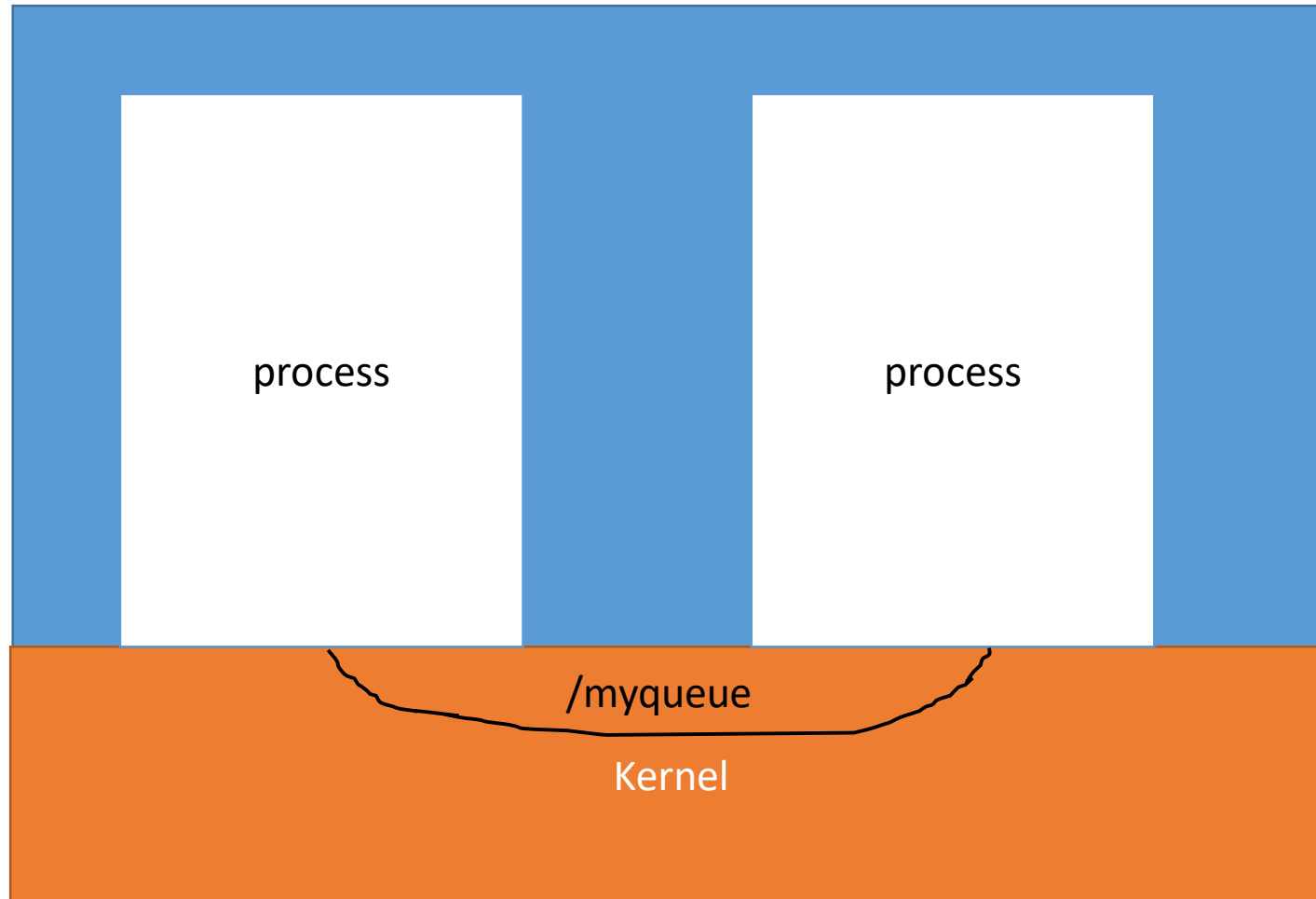
# Programs in the same computer – shared memory



# Shared memory

- In time-sharing multitasking, process address space are protected by hardware to prevent memory from messed up by another process
- However, you can request an area of shared memory from O.S.
- O.S. setup the physical frame and make page table of p1 and p2 point to the same physical frame
- Once you do so, you begins to risk yourself in race

# Programs in the same computer – message queue



```

#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <mqqueue.h>

#include "common.h"

int main(int argc, char **argv)
{
    mqd_t mq;
    char buffer[MAX_SIZE];

    /* open the mail queue */
    mq = mq_open(Queue_NAME, O_WRONLY);
    CHECK((mqd_t)-1 != mq);

    printf("Send to server (enter \"exit\" to stop it):\n");

    do {
        printf("> ");
        fflush(stdout);

        memset(buffer, 0, MAX_SIZE);
        fgets(buffer, MAX_SIZE, stdin);

        /* send the message */
        CHECK(0 <= mq_send(mq, buffer, MAX_SIZE, 0));

    } while (strcmp(buffer, MSG_STOP, strlen(MSG_STOP)));

    /* cleanup */
    CHECK((mqd_t)-1 != mq_close(mq));

```

```

int main(int argc, char **argv)
{
    mqd_t mq;
    struct mq_attr attr;
    char buffer[MAX_SIZE + 1];
    int must_stop = 0;

    /* initialize the queue attributes */
    attr.mq_flags = 0;
    attr.mq_maxmsg = 10;
    attr.mq_msgsize = MAX_SIZE;
    attr.mq_curmsgs = 0;

    /* create the message queue */
    mq = mq_open(Queue_NAME, O_CREAT | O_RDONLY, 0644, &attr);
    CHECK((mqd_t)-1 != mq);

    do {
        ssize_t bytes_read;

        /* receive the message */
        bytes_read = mq_receive(mq, buffer, MAX_SIZE, NULL);
        CHECK(bytes_read >= 0);

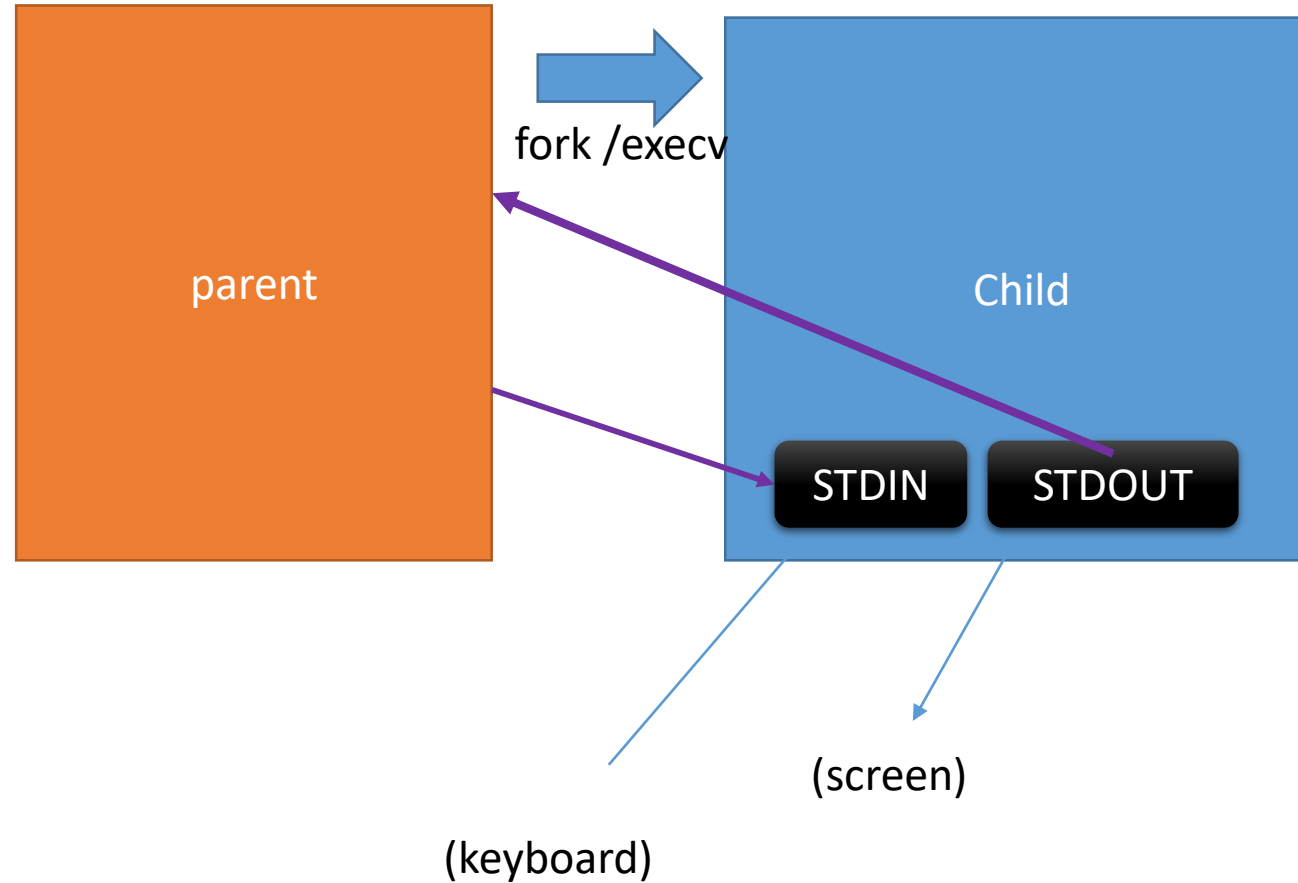
        buffer[bytes_read] = '\0';
        if (! strcmp(buffer, MSG_STOP, strlen(MSG_STOP)))
        {
            must_stop = 1;
        }
        else
        {
            printf("Received: %s\n", buffer);
        }
    } while (!must_stop);

```

# Pipe vs Message Queue

- Pipes are flat, much like a stream, to impose a message structure you would have to implement a protocol on both sides, message queues are message oriented already, no care has to be taken to get, say, the fifth message in the queue.
- Pipes aren't limited in size, message queues are.
- Pipes can be integrated in systems using file descriptors, message queues have their own set of functions, though linux supports `select()`, `poll()`, `epoll()` and friends on the `mqd_t`.
- Pipes, once closed, require some amount of cooperation on both sides to reestablish them, message queues can be closed and reopened on either side without the cooperation of the other side.

# Programs in the same computer – I/O redirect



# Thread to Thread -

- As described in previous slide
- Threads can share static variables in data
- Threads can share referenced object in heap

