

```
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
```

```
#define MAXPAROLA 30
#define MAXRIGA 80
```

```
int main(int argc, char *argv[])
{
    int freq[MAXPAROLA]; /* vettore di contatori
delle frequenze delle lunghezze delle parole */
    char riga[MAXRIGA];
    int i, inizio, lunghezza;
    FILE *f;
```

```
for(i=0; i<MAXPAROLA; i++)
    freq[i]=0;
```

```
if(argc != 2)
```

```
{
    printf(stderr, "ERRORE, serve un parametro con il nome del file\n");
    exit(1);
}
```

```
f = fopen(argv[1], "r");
if(f==NULL)
```

```
{
    printf(stderr, "ERRORE, impossibile aprire il file %s\n", argv[1]);
    exit(1);
}
```

```
while( fgets( riga, MAXRIGA, f ) != NULL )
```



# Multi-Threading

## Introduction to Multi-Threading

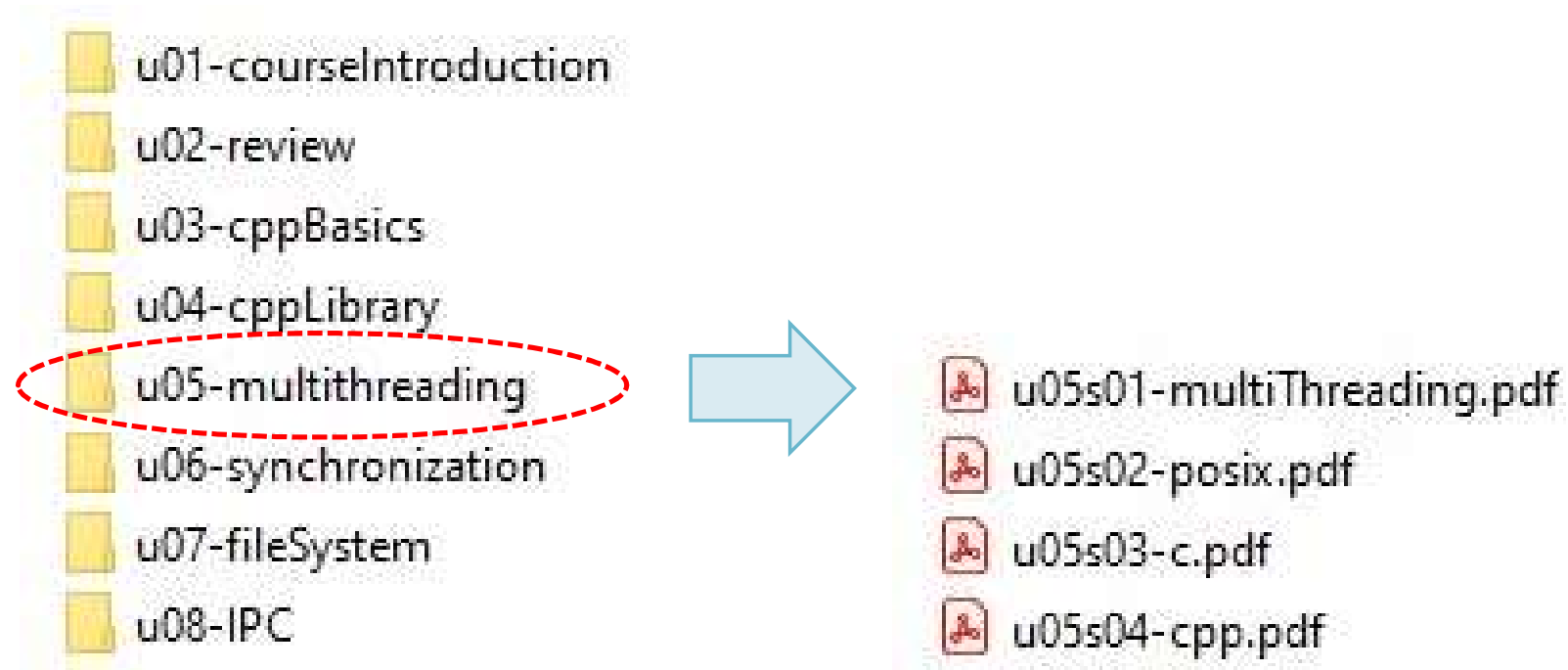
Stefano Quer

Dipartimento di Automatica e Informatica

Politecnico di Torino

# Premises

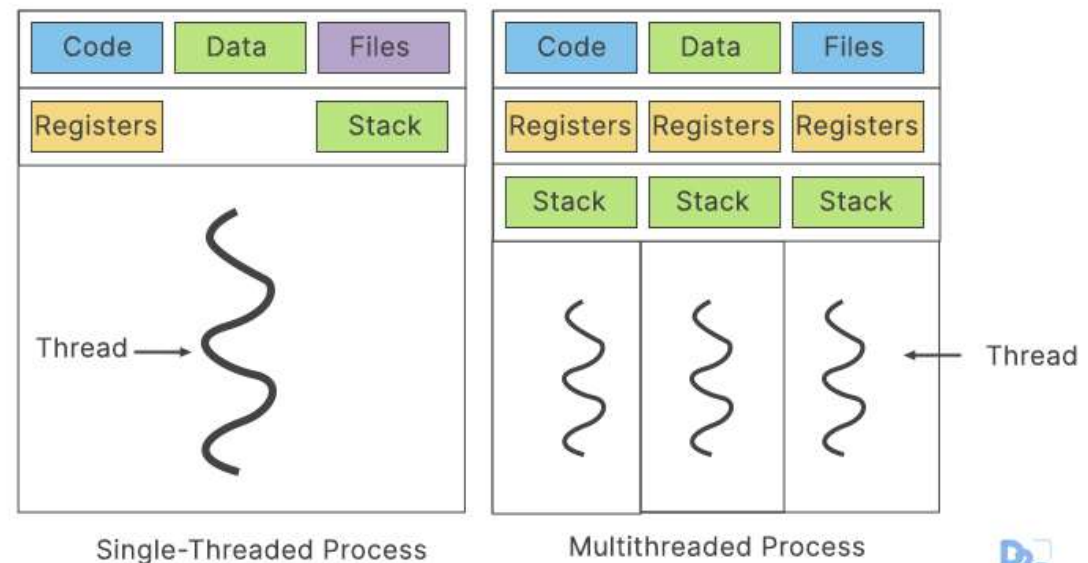
## ❖ Where are we?



# Introduction

- ❖ C and C++ can run multiple threads in a program
  - The thread model allows a program to control multiple different flows of operations (**scheduled and executed independently**) that overlap in time
  - Each flow of operations is referred to as a **thread**

Processes group resources  
Threads are units for the  
scheduling of the CPU



# Introduction

❖ A thread can **share** its **address space** with other threads

➤ **Shared data**

- Code section
- Data section (variables, file descriptors, etc.)
- Operating system resources (e.g., signals)

➤ **Private data**

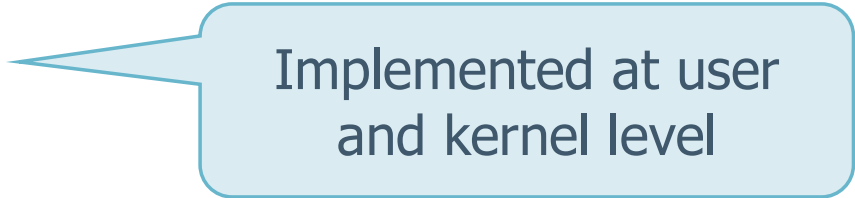
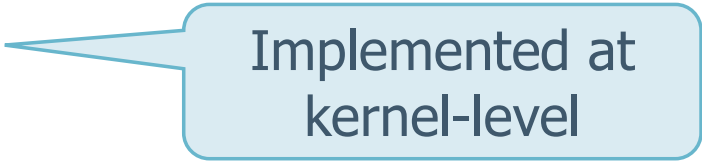
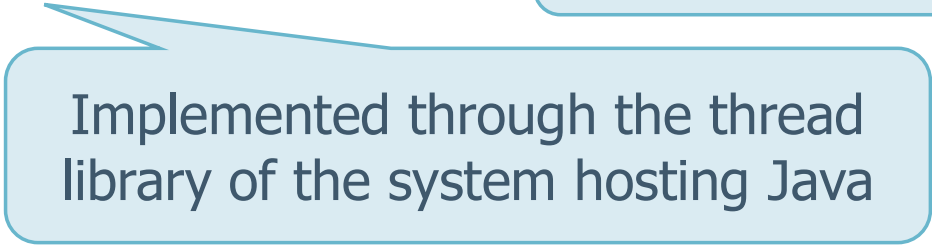
- Program counter and hardware registers
- Stack, i.e., local variables and execution history

Obvious, since a thread implies its own flow of execution (within the same process)

# Introduction

- ❖ Threads allows
  - Shorter response time
  - Shared resources
  - Lower costs for resource management
  - Increased scalability
- ❖ Threads have no implicit data protection
  - They are executed in the same address space and the operating system protection is impossible
  - If the threads are not synchronized, access to shared data is **not thread safe**

## Thread libraries

- ❖ A thread library provides the programmer with the interface to use threads
- ❖ The management can be done
  - At user-level (by functions)
  - At kernel-level (by system calls)
- ❖ The most used thread libraries are
  - POSIX threads (Pthreads)  Implemented at user and kernel level
  - C and C++
  - Windows 32/64  Implemented at kernel-level
  - Java  Implemented through the thread library of the system hosting Java