

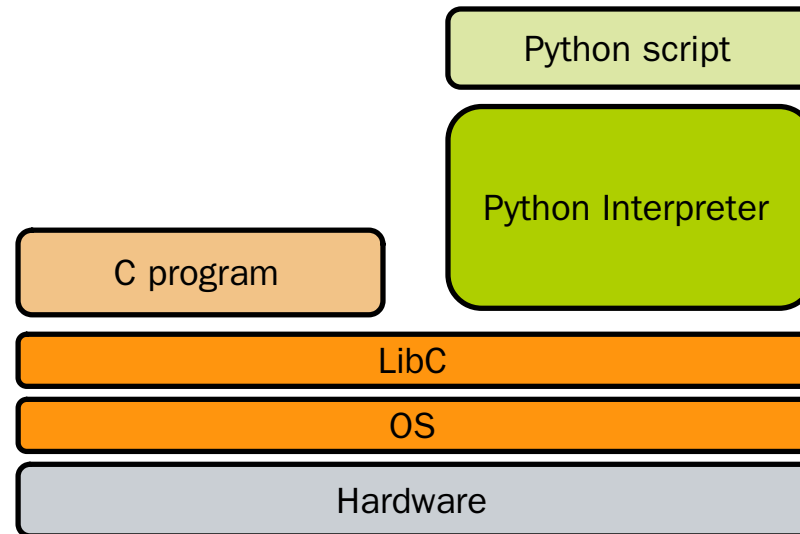
COMP26020 Programming Languages and Paradigms Part 1: C Programming

---

## Case Study: High Performance Computing

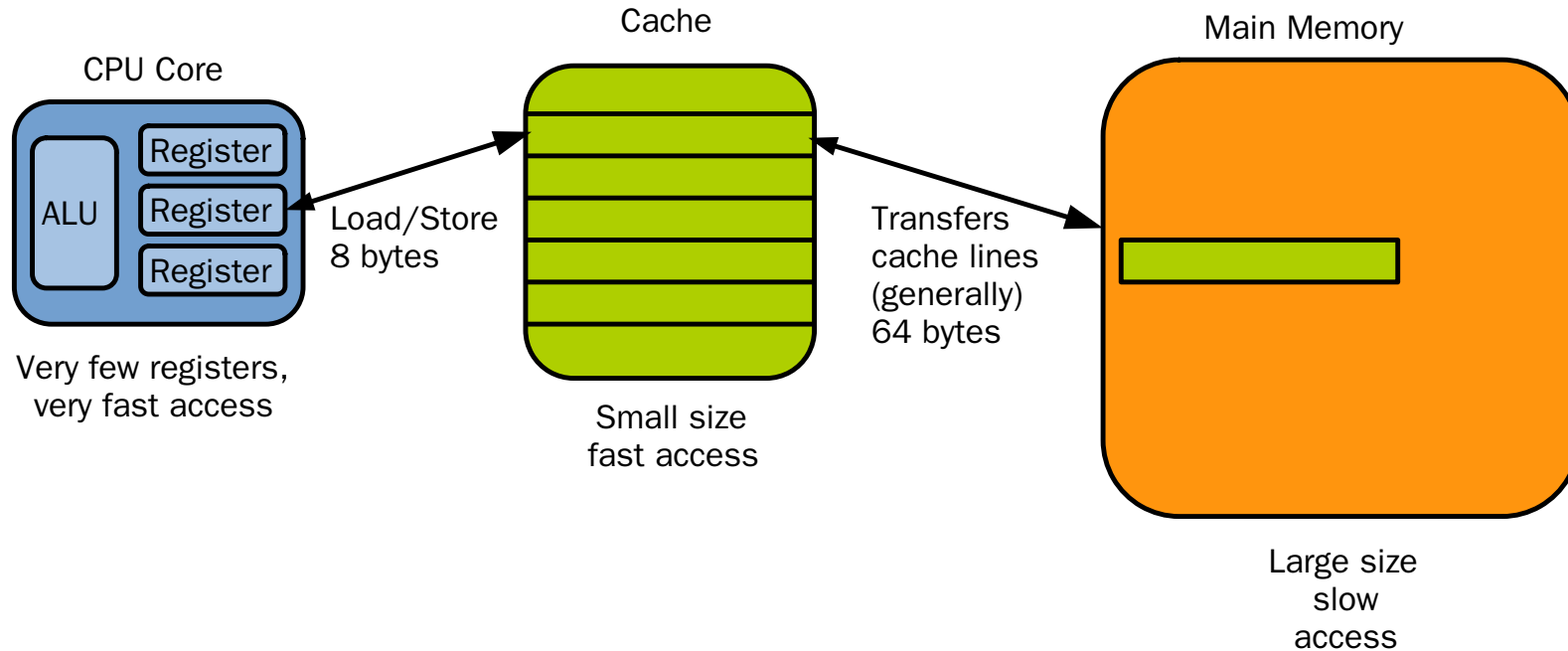
# C/C++ in HPC

C/C++ are extensively used in High Performance Computing (HPC) because of their **speed**



- Due to many reasons, including the fact that they **gives the programmer control over the data memory layout**

# Controlling Memory Layout



- For performance reasons it is very important to fit as much of the data set as possible in the cache

# Controlling Memory Layout

```
typedef struct {
    char c[60];
    int i;
    double d;
} my_struct;

#define N 1000000000
my_struct array[N];

int main(int argc, char **argv) {
    struct timeval start, stop, res;
    my_struct s;

    gettimeofday(&start, NULL);

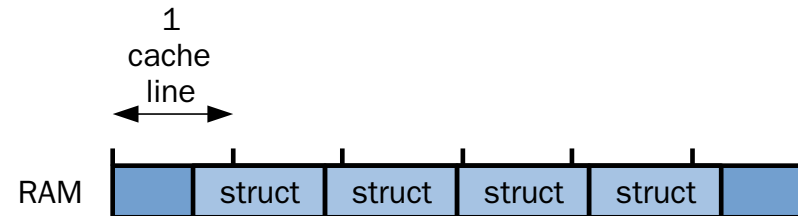
    /* Randomly access N elements */
    for(int i=0; i<N; i++)
        memcpy(&s, &array[rand()%N],
               sizeof(my_struct));

    gettimeofday(&stop, NULL);
    timersub(&stop, &start, &res);
    printf("%ld.%06ld\n", res.tv_sec,
           res.tv_usec);

    return 0; }
```

[20-hpc-case-study/original.c](https://github.com/ericniebler/20-hpc-case-study/blob/master/original.c)

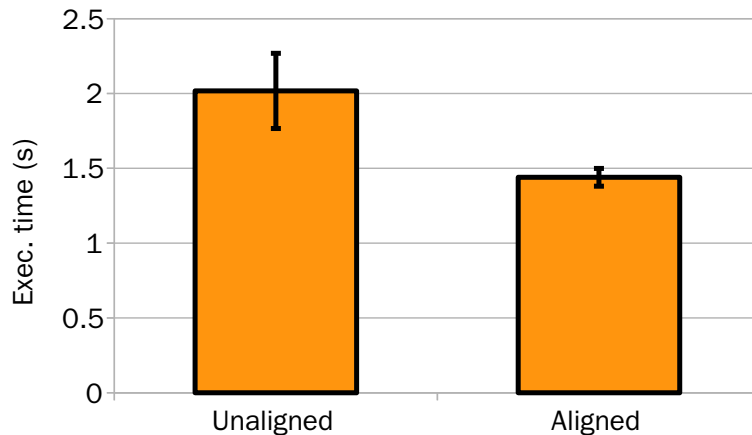
- Struct size:  $60 + 4 \text{ (int)} + 8 \text{ (double)} = 72 \text{ bytes}$ 
  - Larger and not a multiple of the cache line size (64 bytes)
  - Most objects in the array will require to fetch 2 cache lines from main memory



# Controlling Memory Layout

```
typedef struct {  
    char c[52]; // down from 60, we have 52 + 4 + 8 == 64 bytes i.e. a cache line  
    int i;  
    double d;  
} my_struct;  
  
my_struct array[N] __attribute__((aligned(64))); /* force alignment of the array itself */  
  
/* ... */
```

[20-hpc-case-study/optimised.c](#) 



- About 25% faster!
- How much is it on your computer? check out your CPU's cache line size with `cat /sys/devices/system/cpu/cpu0/cache/index0/coherency_line_size`

# Summary

- C/C++ extensively used in HPC because of their speed
    - Run close to the hardware
    - No additional software layers
    - No runtime overhead
    - Integrates well with assembly
    - **Control of the memory layout**
- 

Feedback form: <https://bit.ly/3yz2jzh>

