The Abdus Salam
International Centre
for Theoretical Physics

Measuring Hardware Performance

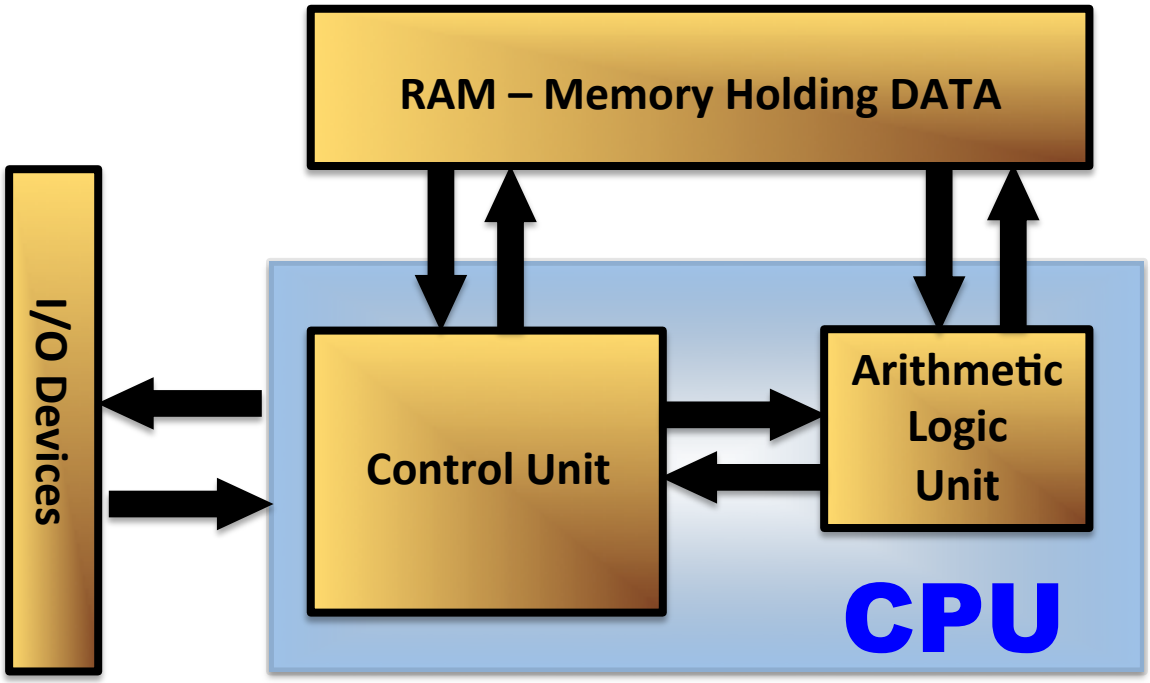
Ivan Girotto – igirotto@ictp.it

Information & Communication Technology Section (ICTS)

International Centre for Theoretical Physics (ICTP)

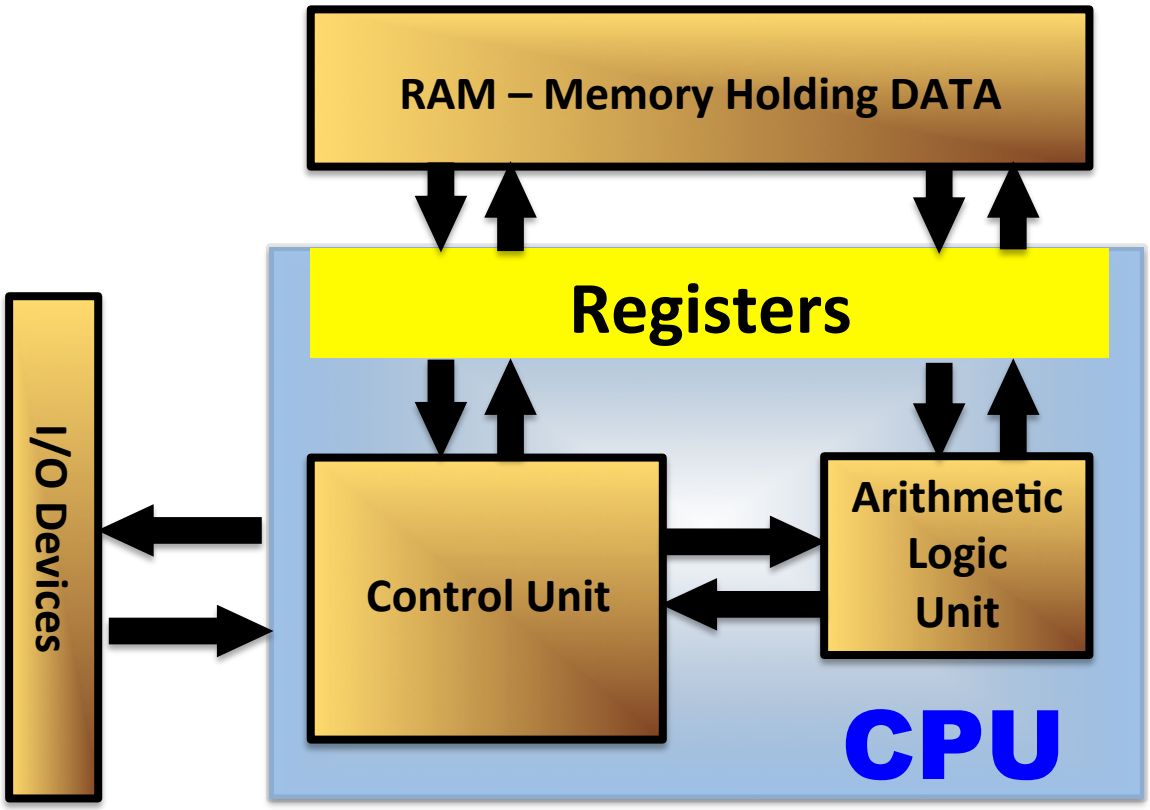
John Von Neumann

The Classical Model



John Von Neumann

The Classical Model



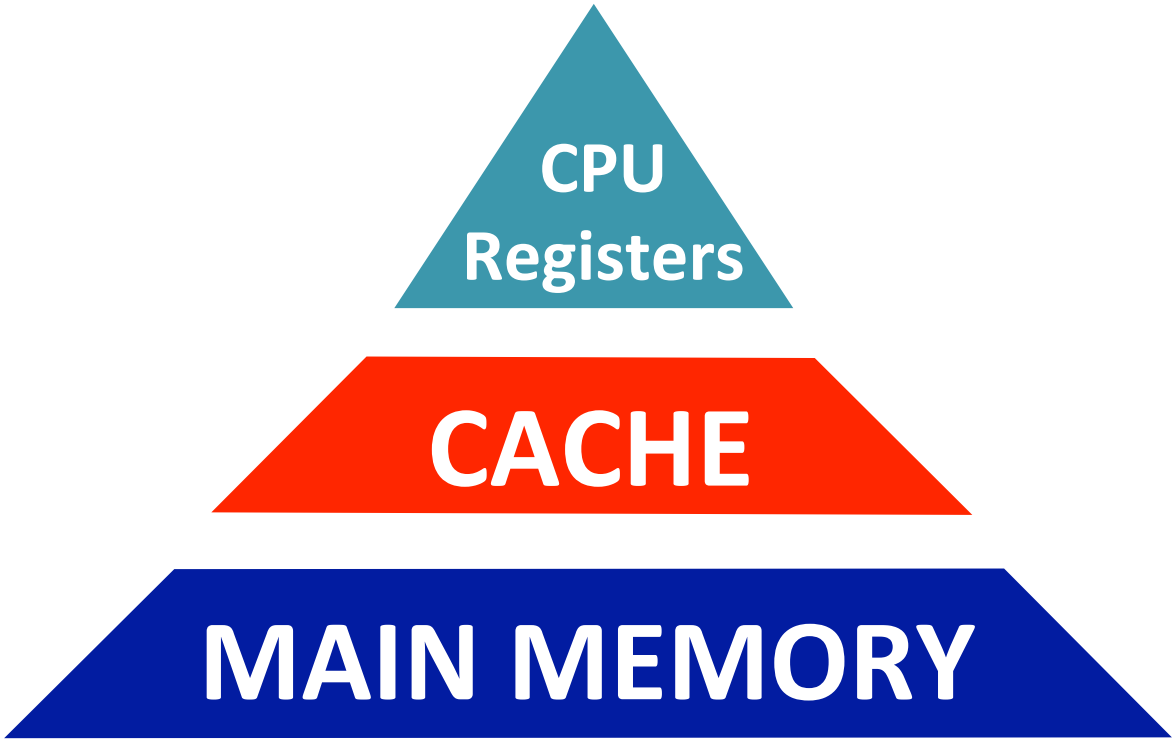
Performance Metrics

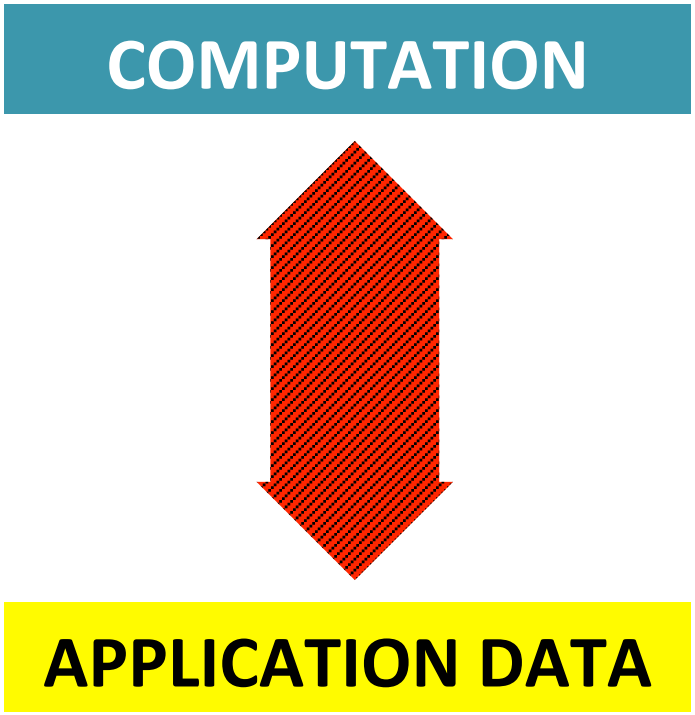
- When all CPU component work at maximum speed that referred *peak of performance*
  - Tech-spec normally describe the theoretical peak
  - Benchmarks measure the real peak
  - Applications show the real performance value
- CPU performance is measured as:
  - Floating point operations per seconds GFLOP/s
- But the real performance is in many cases mostly related to the memory bandwidth (GBytes/s)

How fast is my CPU?!

- CPU power is measured in FLOPS
  - number of floating point operations x second
  - FLOPS = #cores x clock x $\frac{\text{FLOP}}{\text{cycle}}$
- FLOP/cycle is the number of multiply-add (FMA) performed per cycle
  - architectural limit
  - depend also by the instruction set supported

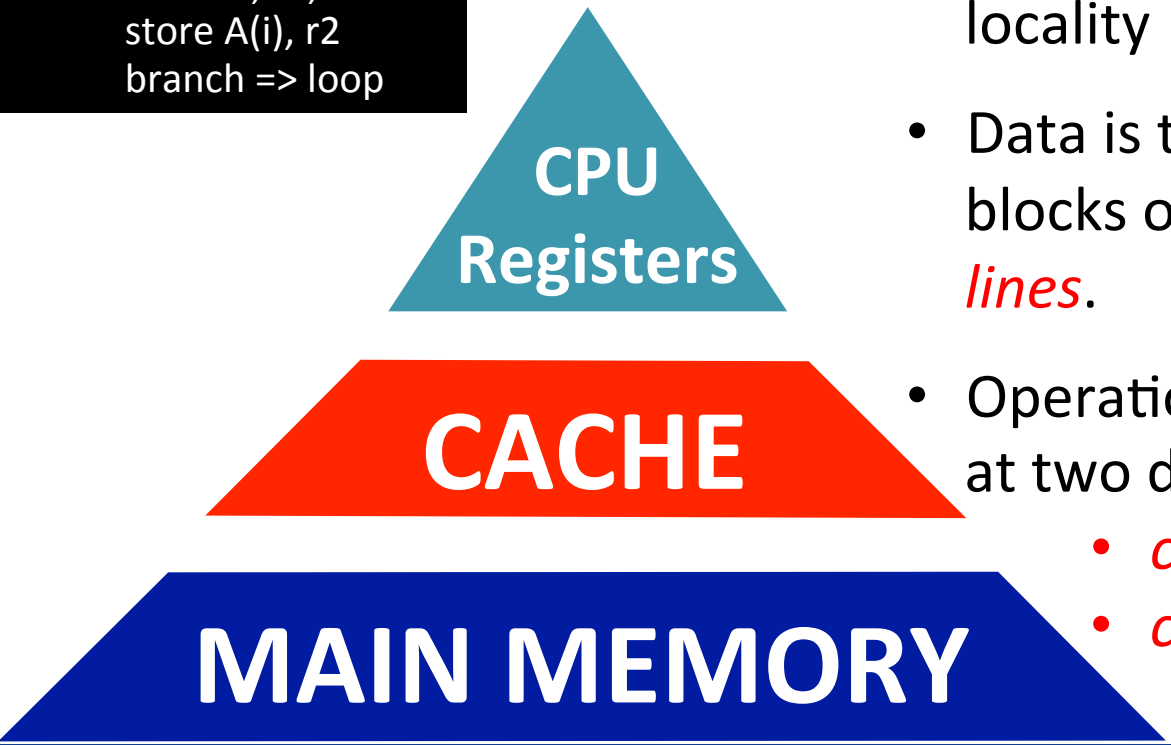
The CPU Memory Hierarchy





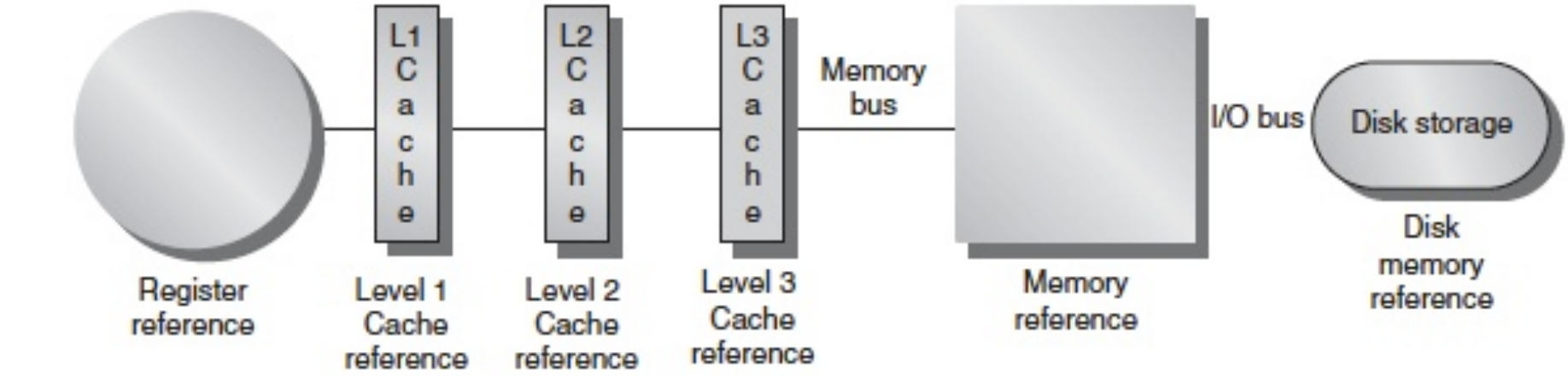
Cache Memory

```
Loop: load r1, A(i)
      load r2, s
      mult r3, r2, r1
      store A(i), r2
      branch => loop
```



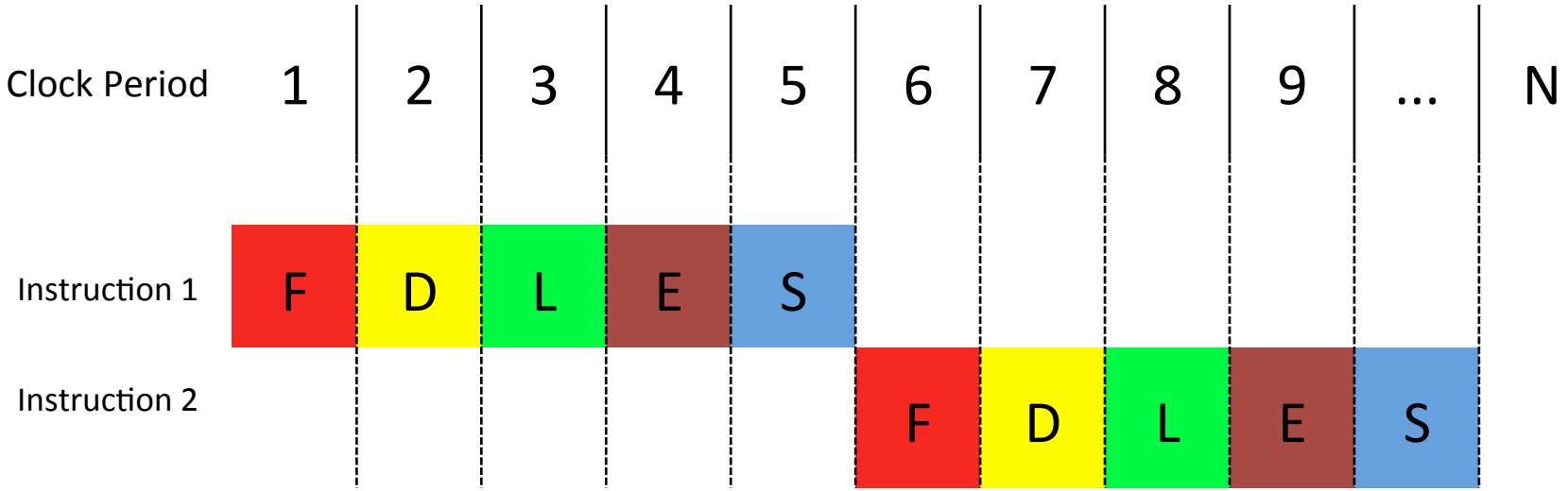
- Designed for temporal/spatial locality
- Data is transferred to cache in blocks of fixed size, called *cache lines*.
- Operation of LOAD/STORE can lead at two different scenario:
  - *cache hit*
  - *cache miss*

The CPU Memory Hierarchy

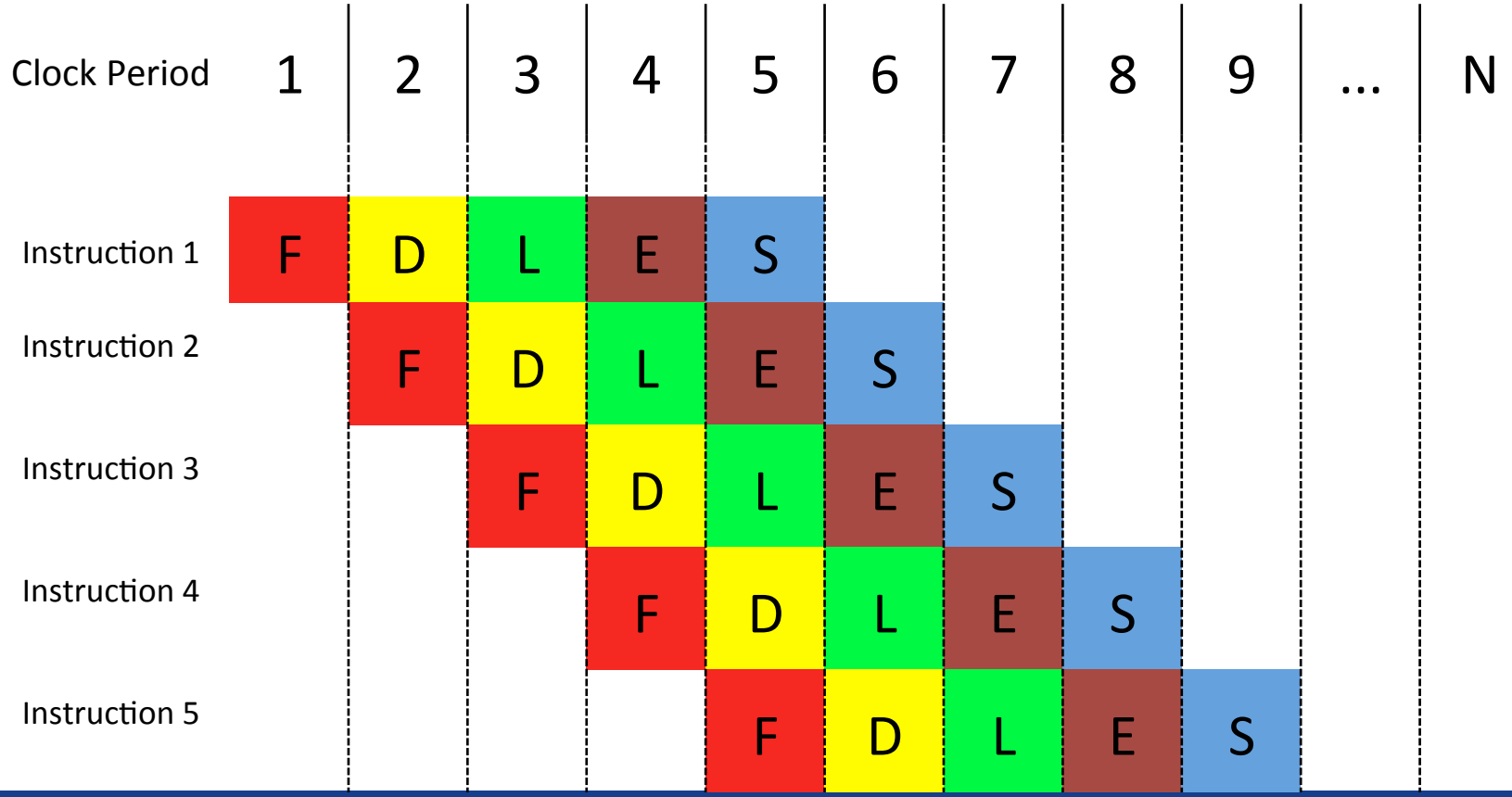


(a) Memory hierarchy for server

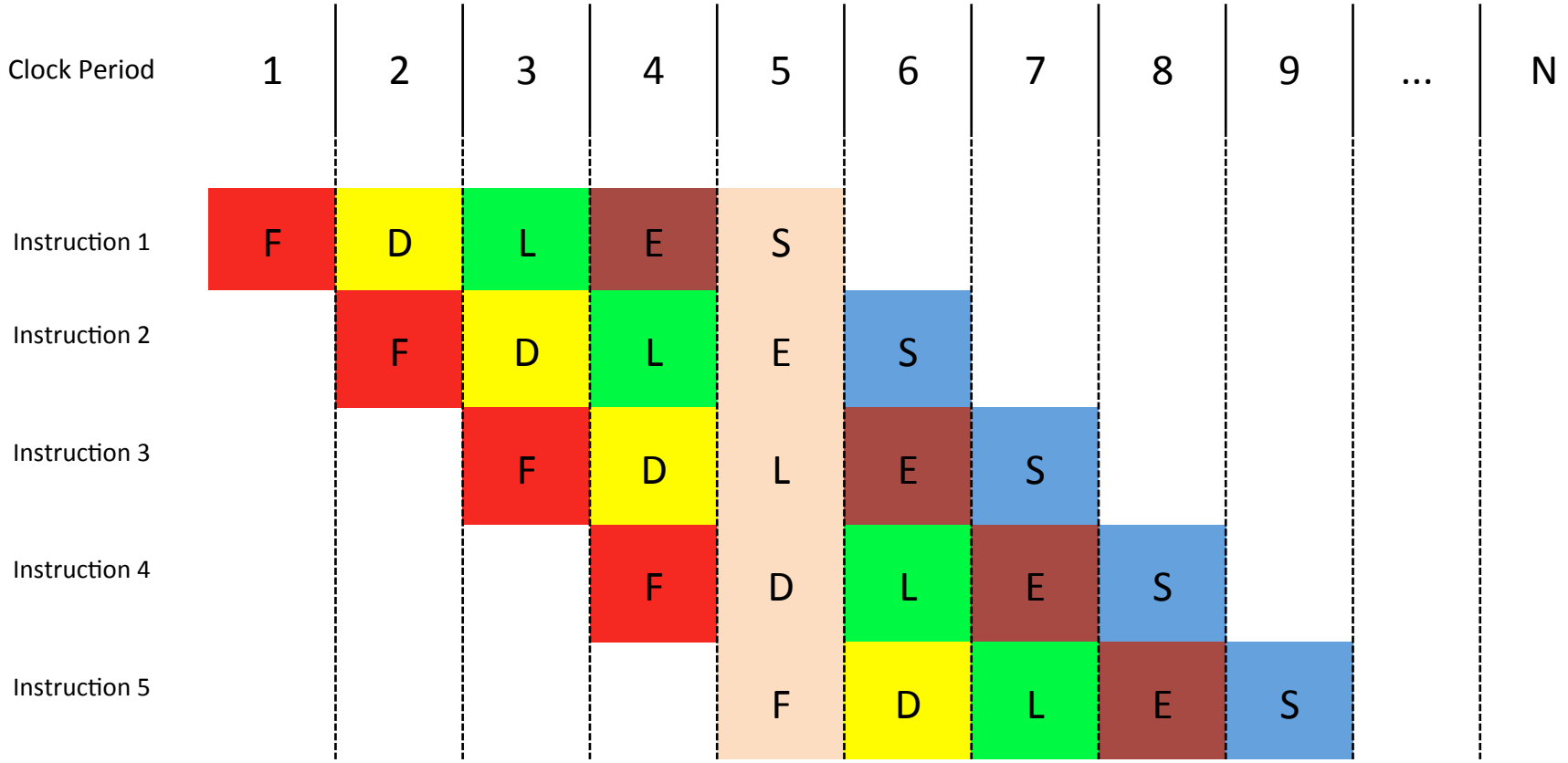
Sequential Processing



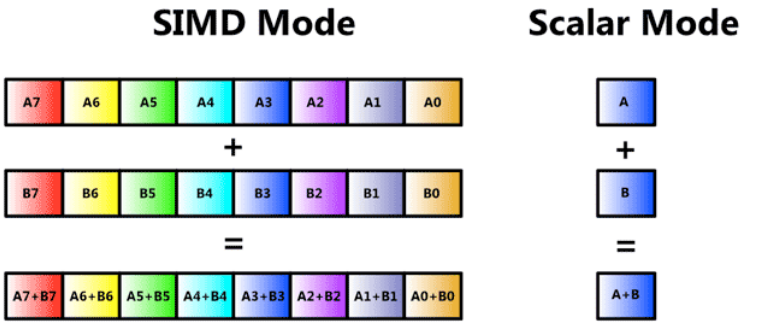
Pipelining


Clock Period
1
2
3
4
5
6
7
8
9
...
N
Instruction 1
Instruction 2
Instruction 3
Instruction 4
Instruction 5
F
D
L
E
S

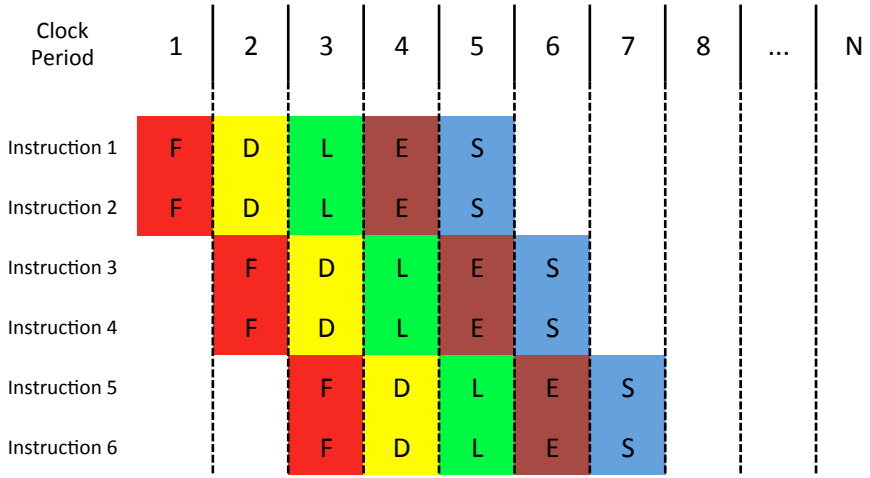
Pipelining


Clock Period
1
2
3
4
5
6
7
8
9
...
N
Instruction 1
F D L E S
Instruction 2
F D L E S
Instruction 3
F D L E S
Instruction 4
F D L E S
Instruction 5
F D L E S

To the Extreme - Parallel Inside

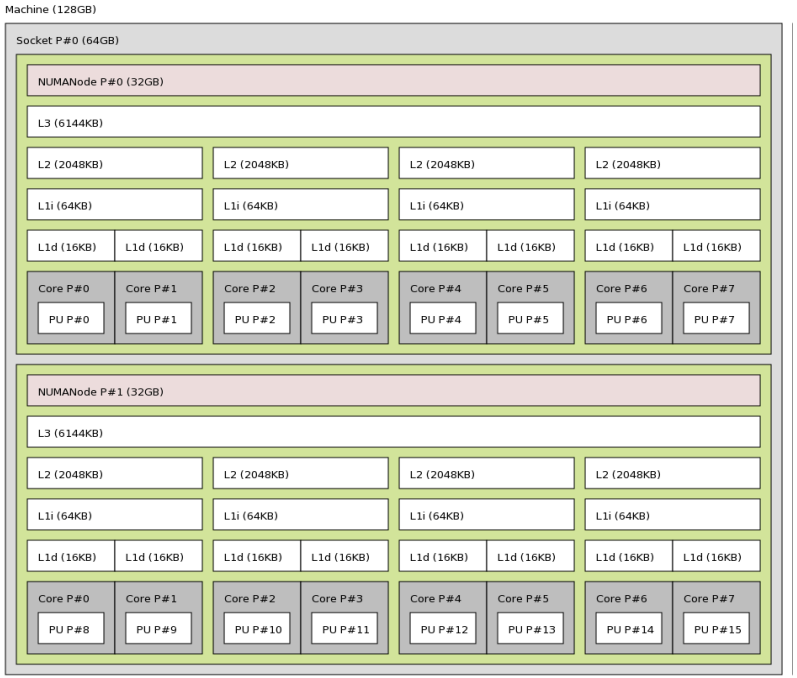


Vector Units for processing multiple data in //

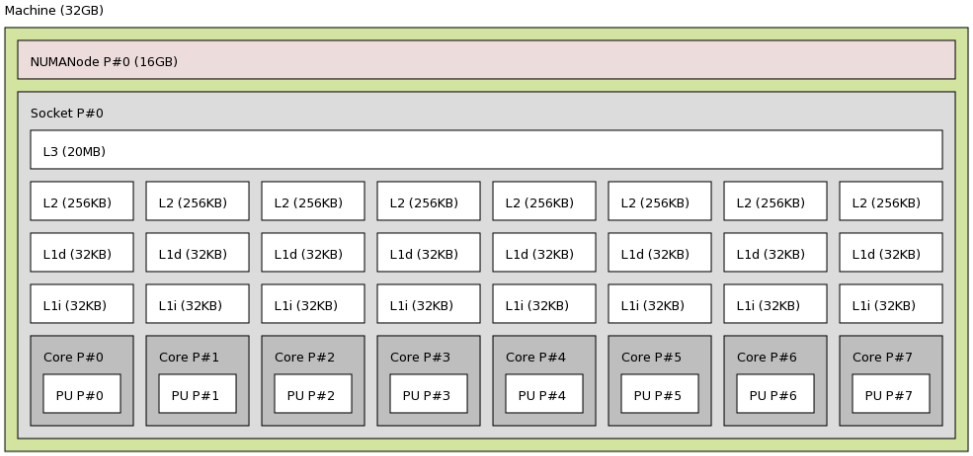


Pipelined/Superscalar design: multiple functional units operate concurrently

Modern CPUs Models

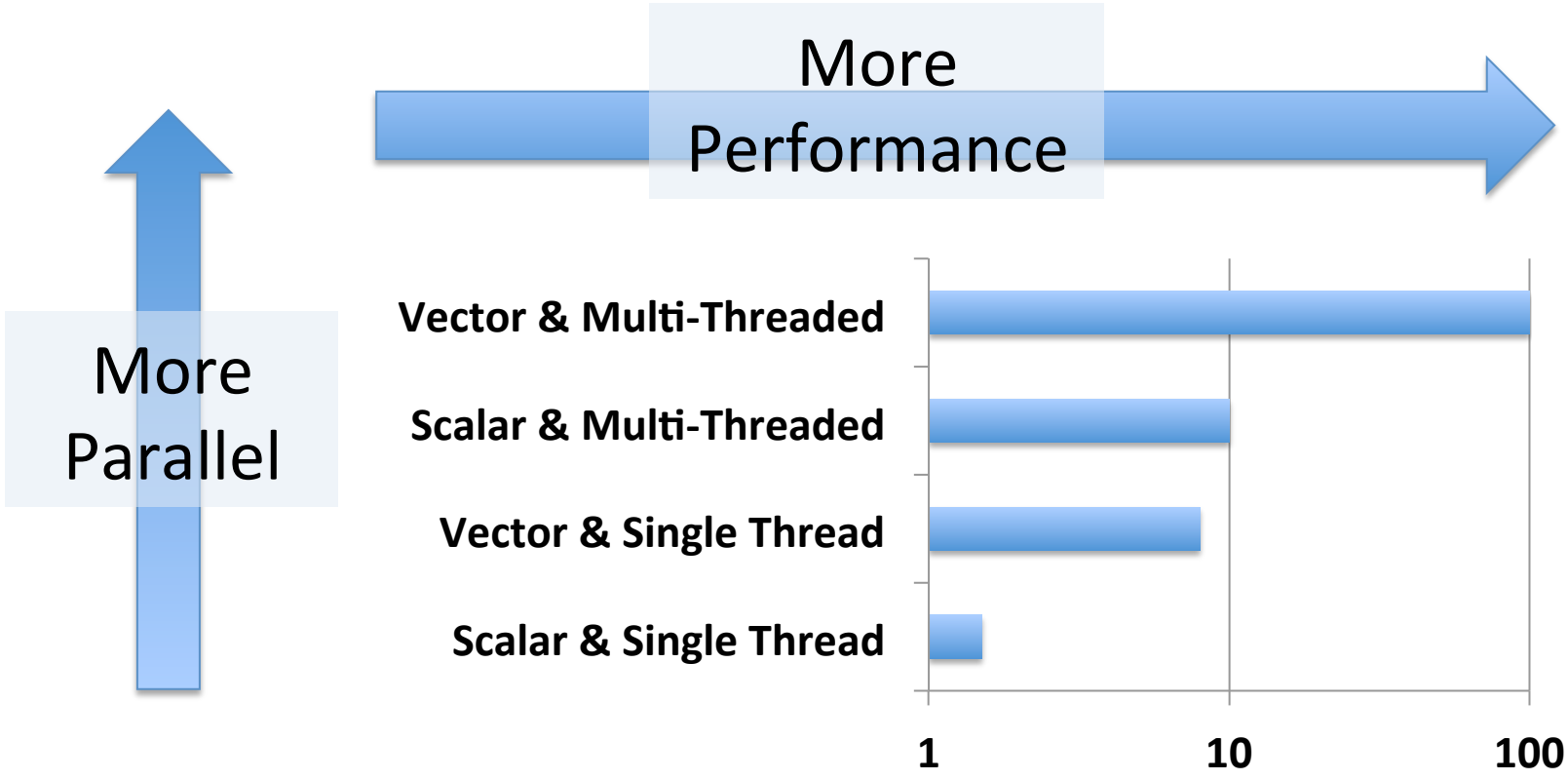


The AMD Opteron 6380 Abu Dhabi 2.5GHz



The Intel Xeon E5-2665 Sandy Bridge-EP 2.4GHz

Threading and Vectorization



DGEMM

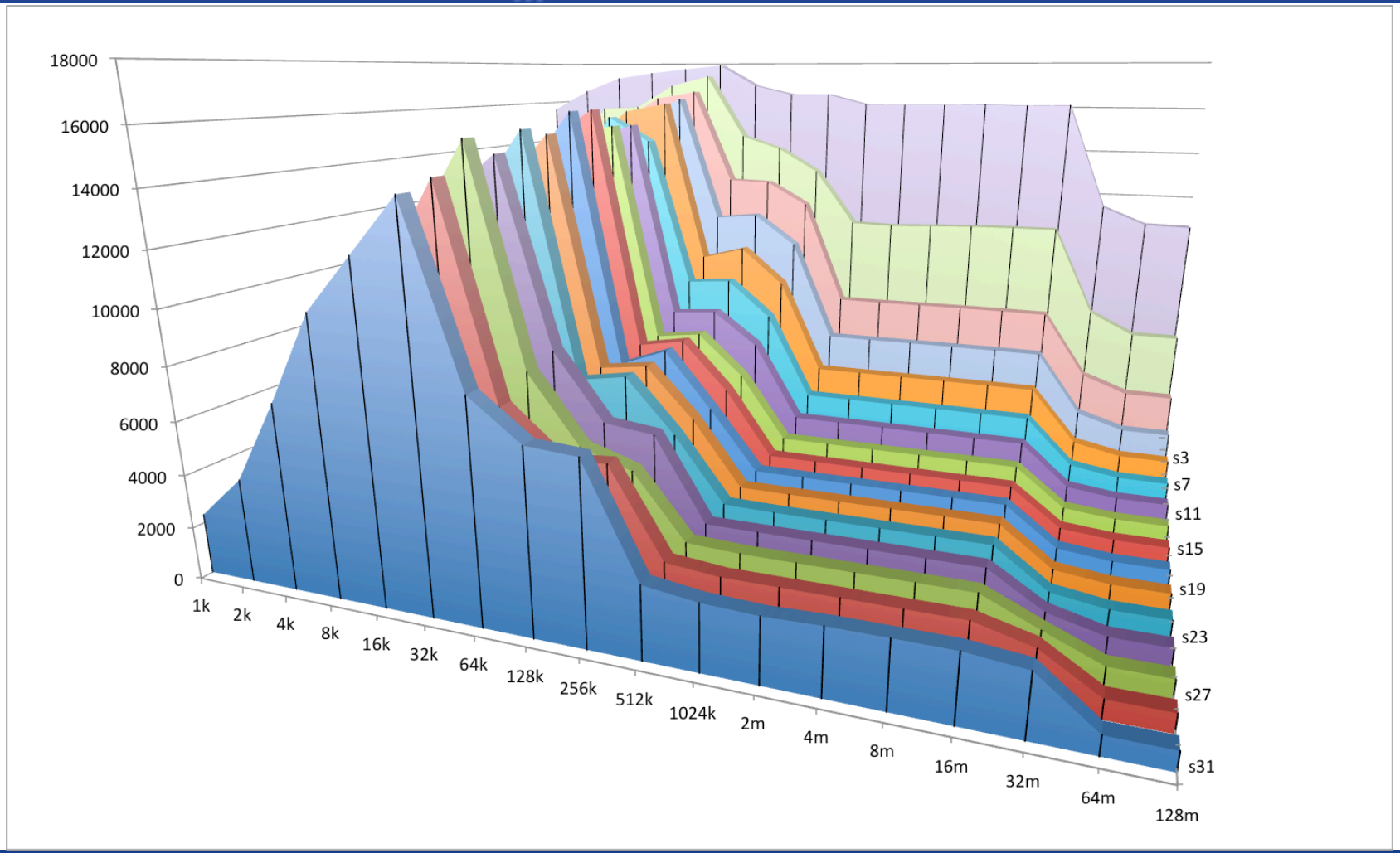
- Matrix Multiplication (MM) is used for measuring HW performance in terms of "compute power"
- The DGEMM is a Level 3 BLAS operation
- For MM the complexity ($2 \times N^3$) is relevant compared to the transferred data ($3 \times N^2$) => compute-bound
- FLOP/s are obtained dividing the time to solution for a DGEMM multiplication by the number of operations

DAXPY

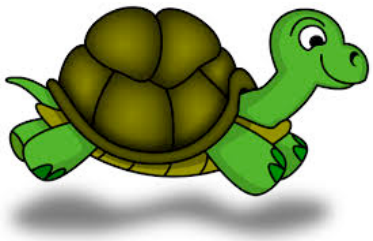
- It is a linear combination of vectors
  - $y = \alpha x + y$
- The DAXPY is a Level 1 BLAS operation
- For DAXPY the complexity (2 x N) is equal to the transferred data (2 x N) => memory-bound
- It used to measure the memory bandwidth of a system and how it impacts on performances

```
/* $begin mountainmain */
for (size = MINBYTES; size <= MAXBYTES; size <<= 1)
    for (stride = MAXSTRIDE - 1; stride >= 1; stride -= STRIDESTRIDE) {
        printf("%.0f\t", run(size, stride, Mhz));
    }
    printf("\n");
}
```

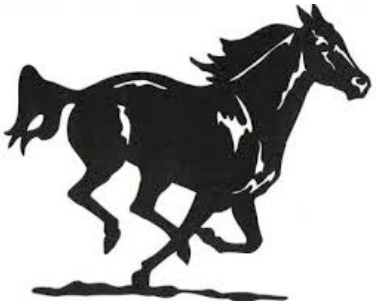
```
/* $begin mountainfuns */
void test(int elems, int stride) /* The test function */
{
    for (int i = 0; i < elems; i += stride)
        int result += data[i];
}

/* Run test(elems, stride) and return read throughput (MB/s) */
double run(int size, int stride, double Mhz)
{
    double cycles;
    int elems = size / sizeof(int);

    /* call test(elems,stride) */
```


18000
16000
14000
12000
10000
8000
6000
4000
2000
0
1k
2k
4k
8k
16k
32k
64k
128k
256k
512k
1024k
2m
4m
8m
16m
32m
64m
128m
s3
s7
s11
s15
s19
s23
s27
s31

Libraries



Highly-portable (included into some Linux distributions)



Freely available Open Source Optimized

Third-Party Highly-Optimized

Thanks for your attention!!