NUMA EFFECT

Setup

```
# GCC 8.1
$ gcc -O3 -march=native -fopenmp streams.c -lnuma -o stream

# Intel compiler 2018.3
$ icc -O3 -march=native -qopenmp streams.c -lnuma -o stream
```

- 209 715 200 doubles in each array (total 4.7 GiB)
- No CPU pinning

Single threaded (Gb/s)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC

Single threaded (Gb/s)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	11.6	18.9	12.5	12.7

Single threaded (Gb/s)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	11.6	18.9	12.5	12.7
numactl -l	11.6	18.9	12.5	12.7

Single threaded (Gb/s)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	11.6	18.9	12.5	12.7
numactl -l	11.6	18.9	12.5	12.7
cpubind=0 membind=0	11.7	18.9	12.6	12.7

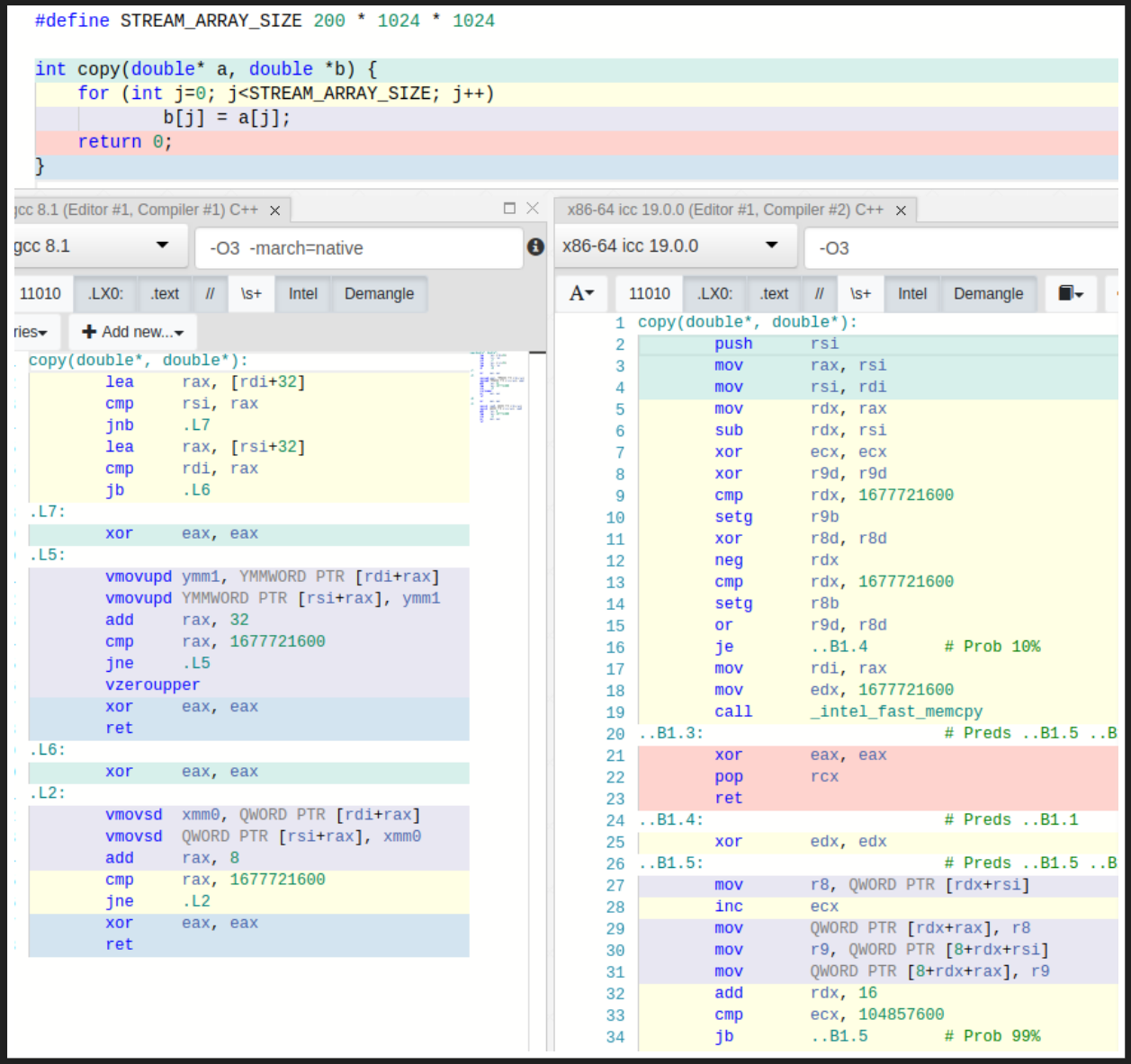
Single threaded (Gb/s)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	11.6	18.9	12.5	12.7
numactl -l	11.6	18.9	12.5	12.7
cpubind=0 membind=0	11.7	18.9	12.6	12.7
cpubind=0 membind=1	8.9	11.8	9.6	9.8

Single threaded (Gb/s)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	11.6	18.9	12.5	12.7
numactl -l	11.6	18.9	12.5	12.7
cpubind=0 membind=0	11.7	18.9	12.6	12.7
cpubind=0 membind=1	8.9	11.8	9.6	9.8
interleave=all	10.2	15.1	11.4	11.5

Why the difference?



vmovupd vs _intel_fast_memcpy

How to check?

How to check?

```
// get NUMA node of page
move_pages(0, 1, &addr, NULL, &node, 0);

// get NUMA node of current thread
numa_node_of_cpu(sched_getcpu());
```

How to check?

```c
// get NUMA node of page
move_pages(0, 1, &addr, NULL, &node, 0);

// get NUMA node of current thread
numa_node_of_cpu(sched_getcpu());
```

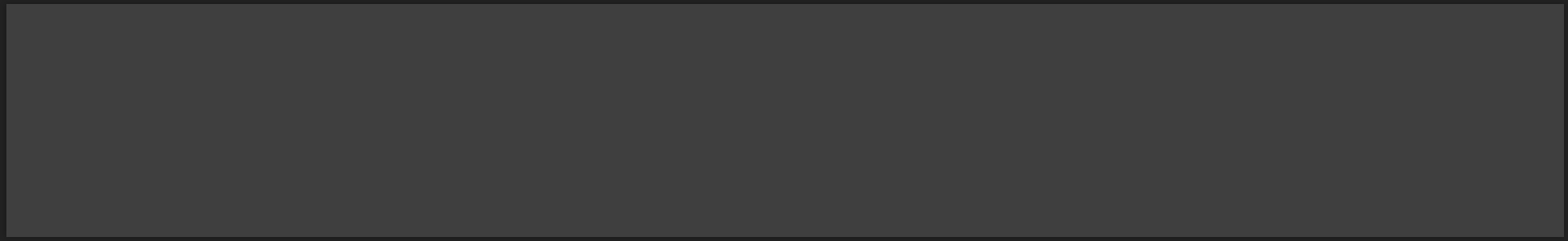
```
$ numastat
                           node0           node1
numa_hit               5926843663      6971552565
numa_miss                27077349         1705662
numa_foreign              1705662        27077349
interleave_hit            2501176         2501190
local_node             5925602884      6965325516
other_node               28318128         7932711
```

```
$ numastat > stats1.txt
$ ./stream
$ numastat > stats2.txt
$ diff stats1.txt stats2.txt
```

```
$ numastat > stats1.txt
$ ./stream
$ numastat > stats2.txt
$ diff stats1.txt stats2.txt

< local_node           5926865901      6966782363
---
> local_node           5927482327      6966789151
# ~600k local_node hits, 300M doubles total
```

Why 600k?



Why 600k?

```
Doubles per array: 100 * 1024 * 1024 (100 million)
```

Why 600k?

```
Doubles per array: 100 * 1024 * 1024 (100 million)
Array size: 100 MiB * 8 = 800 MiB
```

Why 600k?

```
Doubles per array: 100 * 1024 * 1024 (100 million)
Array size: 100 MiB * 8 = 800 MiB
Total arrays size: 800 MiB * 3 = 2.4 GiB
```

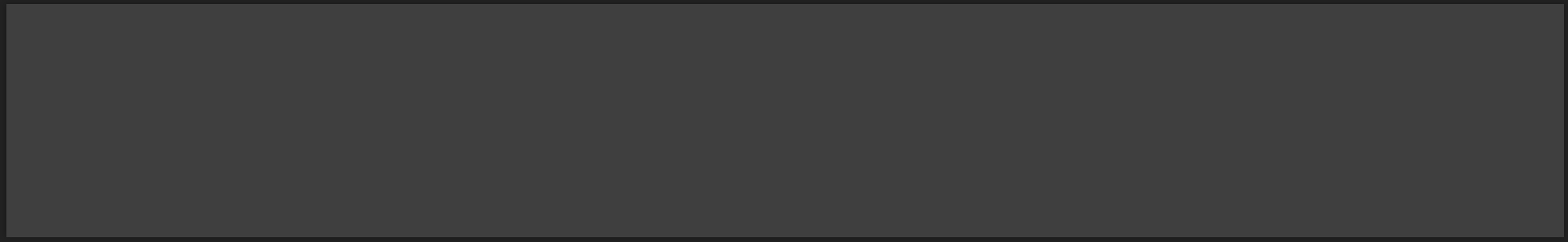
Why 600k?

```
Doubles per array: 100 * 1024 * 1024 (100 million)
Array size: 100 MiB * 8 = 800 MiB
Total arrays size: 800 MiB * 3 = 2.4 GiB
Page count: 2.4 GiB / 4KiB = 600k
```

Why 600k?

```
Doubles per array: 100 * 1024 * 1024 (100 million)
Array size: 100 MiB * 8 = 800 MiB
Total arrays size: 800 MiB * 3 = 2.4 GiB
Page count: 2.4 GiB / 4KiB = 600k
getpagesize() // get configured page size, usually 4k or 2MiB
```

When does allocation happen? (first-touch policy)



When does allocation happen? (first-touch policy)

```
// cat /proc/mem/info - 100 MiB used memory
```

When does allocation happen? (first-touch policy)

```
// cat /proc/mem/info - 100 MiB used memory
int* mem = (int*) malloc(1024 * 1024 * 1024); // allocate 1 GiB
```

When does allocation happen? (first-touch policy)

```
// cat /proc/mem/info - 100 MiB used memory
int* mem = (int*) malloc(1024 * 1024 * 1024); // allocate 1 GiB
// cat /proc/mem/info - 100 MiB used memory
```

When does allocation happen? (first-touch policy)

```
// cat /proc/mem/info - 100 MiB used memory
int* mem = (int*) malloc(1024 * 1024 * 1024); // allocate 1 GiB
// cat /proc/mem/info - 100 MiB used memory
mem[0] = 5;          // first page is allocated
```

When does allocation happen? (first-touch policy)

```
// cat /proc/mem/info - 100 MiB used memory
int* mem = (int*) malloc(1024 * 1024 * 1024); // allocate 1 GiB
// cat /proc/mem/info - 100 MiB used memory
mem[0] = 5;         // first page is allocated
*(mem + 1024) = 5; // second page is allocated
```

```
// ADD
for (int j=0; j<STREAM_ARRAY_SIZE; j++)
  c[j] = a[j] + b[j];
```

```
// ADD
for (int j=0; j<STREAM_ARRAY_SIZE; j++)
  c[j] = a[j] + b[j];


// this has consistently ~same speed as ADD
for (int j=0; j<STREAM_ARRAY_SIZE; j++)
  a[j] = b[j] + scalar * c[j];
```

```
// ADD
for (int j=0; j<STREAM_ARRAY_SIZE; j++)
  c[j] = a[j] + b[j];


// this has consistently ~same speed as ADD
for (int j=0; j<STREAM_ARRAY_SIZE; j++)
  a[j] = b[j] + scalar * c[j];


// but this is slower
for (int j=0; j<STREAM_ARRAY_SIZE; j++)
  a[j] = b[j] + scalar + c[j];
```

Dissasembly binary (FMA instruction)

```
$ gcc ... -g
$ objdump -S -l
```

Dissasembly binary (FMA instruction)

```
$ gcc ... -g
$ objdump -S -l

a[j] = b[j]+scalar*c[j];
401ef6: c4 c2 f1 a9 04 c3       vfmadd213sd (%r11,%rax,8),%xmm1,%xmm0
```

Multithreaded (12 threads, Gb/s) (* No Pin/Pin)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC

Multithreaded (12 threads, Gb/s) (* No Pin/Pin)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	42.3/38.9*	50/107.6*	44.5	44.1/86*

Multithreaded (12 threads, Gb/s) (* No Pin/Pin)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	42.3/38.9*	50/107.6*	44.5	44.1/86*
numactl -l	42.4	60/107.2*	48.5	57.5/86*

Multithreaded (12 threads, Gb/s) (* No Pin/Pin)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	42.3/38.9*	50/107.6*	44.5	44.1/86*
numactl -l	42.4	60/107.2*	48.5	57.5/86*
cpubind=0 membind=0	38.9	52.6	44.4	44.5

Multithreaded (12 threads, Gb/s) (* No Pin/Pin)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	42.3/38.9*	50/107.6*	44.5	44.1/86*
numactl -l	42.4	60/107.2*	48.5	57.5/86*
cpubind=0 membind=0	38.9	52.6	44.4	44.5
cpubind=0 membind=1	16.2	18.1	17.6	17.6

Multithreaded (12 threads, Gb/s) (* No Pin/Pin)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	42.3/38.9*	50/107.6*	44.5	44.1/86*
numactl -l	42.4	60/107.2*	48.5	57.5/86*
cpubind=0 membind=0	38.9	52.6	44.4	44.5
cpubind=0 membind=1	16.2	18.1	17.6	17.6
interleave=all	33.2	40/60.7*	35.9	55.5

Multithreaded (12 threads, Gb/s) (* No Pin/Pin)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC
./stream	42.3/38.9*	50/107.6*	44.5	44.1/86*
numactl -l	42.4	60/107.2*	48.5	57.5/86*
cpubind=0 membind=0	38.9	52.6	44.4	44.5
cpubind=0 membind=1	16.2	18.1	17.6	17.6
interleave=all	33.2	40/60.7*	35.9	55.5
cpubind=0 interleave=all	31.2	36.8	33.6	33.6

Multithreaded (24 threads, Gb/s)

Command	Copy GCC	Copy ICC	Triad GCC	Triad ICC

Multithreaded (24 threads, Gb/s)				
Command	**Copy GCC**	**Copy ICC**	**Triad GCC**	**Triad ICC**
./stream	77.8	104.2	88.8	87.8

Multithreaded (24 threads, Gb/s)				
Command	**Copy GCC**	**Copy ICC**	**Triad GCC**	**Triad ICC**
./stream	77.8	104.2	88.8	87.8
numactl -l	77.8	104.1	88.8	88.1

Multithreaded (24 threads, Gb/s)				
Command	**Copy GCC**	**Copy ICC**	**Triad GCC**	**Triad ICC**
./stream	77.8	104.2	88.8	87.8
numactl -l	77.8	104.1	88.8	88.1
interleave=all	53.4	64.9	59.0	59.0