

ΕΛΠ 605: Προχωρημένη Αρχιτεκτονική Υπολογιστών

Εργαστήριο Αρ. 4

perf Linux Performance Tool



Linux Performance Monitoring Tools

ps

top

Press (Shift+P)

to sort processes

as per CPU

utilization

```
petrosp@103ws30:~  
top - 08:58:32 up 10 days, 14:09, 4 users, load average: 0.01, 0.06, 0.05  
Tasks: 253 total, 1 running, 249 sleeping, 3 stopped, 0 zombie  
%Cpu(s): 2.2 us, 0.7 sy, 0.0 ni, 97.2 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st  
KiB Mem : 7933792 total, 5409420 free, 706624 used, 1817748 buff/cache  
KiB Swap: 10485756 total, 10485756 free, 0 used. 6825720 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
7879	gdm	20	0	1616924	143812	50008	S	4.3	1.8	225:25.00	gnome-shell
9	root	20	0	0	0	0	S	0.3	0.0	6:42.24	rcu_sched
992	root	20	0	4368	580	492	S	0.3	0.0	48:47.06	rngd
7804	root	20	0	168076	39928	16544	S	0.3	0.5	14:47.65	Xorg
7852	gdm	20	0	44848	1752	1412	S	0.3	0.0	13:25.21	dbus-daemon
1	root	20	0	193764	6920	3968	S	0.0	0.1	0:31.39	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.65	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:03.59	ksoftirqd/0
5	root	0	-20	0	0	0	S	0.0	0.0	0:00.00	kworker/0:0H
7	root	rt	0	0	0	0	S	0.0	0.0	0:05.24	migration/0
8	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_bh
10	root	rt	0	0	0	0	S	0.0	0.0	0:02.44	watchdog/0
11	root	rt	0	0	0	0	S	0.0	0.0	0:02.32	watchdog/1
12	root	rt	0	0	0	0	S	0.0	0.0	0:04.08	migration/1
13	root	20	0	0	0	0	S	0.0	0.0	0:01.34	ksoftirqd/1
16	root	rt	0	0	0	0	S	0.0	0.0	0:02.35	watchdog/2
17	root	rt	0	0	0	0	S	0.0	0.0	0:04.61	migration/2
18	root	20	0	0	0	0	S	0.0	0.0	0:01.09	ksoftirqd/2
21	root	rt	0	0	0	0	S	0.0	0.0	0:02.52	watchdog/3
22	root	rt	0	0	0	0	S	0.0	0.0	0:04.05	migration/3



Linux Performance Monitoring Tools

Htop (<https://linux.die.net/man/1/htop>)

an interactive system-monitor process-viewer

```
petros@103ws30:~  
1 [|||||] 5.3%] Tasks: 118, 185 thr; 1 running  
2 [||] 0.7%] Load average: 0.07 0.09 0.06  
3 [|||] 2.7%] Uptime: 10 days, 14:15:52  
4 [|||] 2.7%]  
Mem[|||||||||||||||||||||||||||||] 799M/7.57G  
Swp[ ] 0K/10.00G  
Send signal: PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command  
0 Cancel 7879 gdm 20 0 1579M 139M 50008 S 2.7 1.8 3h45:37 gnome-shell --mode=gdm  
1 SIGHUP 28889 petrosp 20 0 32064 3008 1452 R 1.3 0.0 0:00.15 htop  
2 SIGINT 1087 root 20 0 159M 10084 1020 S 0.7 0.1 3:44.97 /usr/bin/perl /usr/sbin/x2gocleansessions  
3 SIGQUIT 8092 gdm 20 0 1579M 139M 50008 S 0.0 1.8 5:03.64 gnome-shell --mode=gdm  
4 SIGILL 992 root 20 0 4368 580 492 S 0.0 0.0 48:48.28 /sbin/rngd -f  
5 SIGTRAP 7852 gdm 20 0 44848 1752 1412 S 0.0 0.0 13:25.96 /bin/dbus-daemon --config-file=/etc/at-spi2/accessibi  
6 SIGABRT 7804 root 20 0 164M 39928 16544 S 0.0 0.5 14:48.29 /usr/bin/Xorg :1 -background none -noreset -audit 4 -  
6 SIGIOT 7963 gdm 20 0 545M 7540 5600 S 0.0 0.1 12:14.83 /usr/libexec/caribou  
7 SIGBUS 1250 root 20 0 319M 26736 6416 S 0.0 0.3 0:09.60 /usr/bin/python -Es /usr/sbin/firewalld --nofork --no  
8 SIGFPE 1 root 20 0 189M 6920 3968 S 0.0 0.1 0:31.41 /usr/lib/systemd/systemd --switched-root --system --no  
9 SIGKILL 562 root 20 0 78208 38420 37968 S 0.0 0.5 0:11.30 /usr/lib/systemd/systemd-journald  
10 SIGUSR1 580 root 20 0 262M 5896 2616 S 0.0 0.1 0:00.00 /usr/sbin/lvmetad -f  
11 SIGSEGV 598 root 20 0 47504 5740 2804 S 0.0 0.1 0:00.87 /usr/lib/systemd/systemd-udev  
12 SIGUSR2 955 root 15 0 55416 1828 1384 S 0.0 0.0 0:00.33 /sbin/auditd -s
```



perf: Linux *profiling* with performance counters

Performance counters are CPU hardware registers that count hardware events such as instructions executed, cache-misses suffered, or branches mispredicted.

perf provides rich generalized abstractions over hardware specific capabilities. Among others, it provides per task, per CPU and per-workload counters, sampling on top of these and source code event annotation. Perf gives you visibility where the Hotspots of your program are.

https://perf.wiki.kernel.org/index.php/Main_Page



Intel Core Performance Monitor Unit (PMU)

The core PMU's capability is similar to those described in Section 18.7.1 and Section 18.8, with some differences and enhancements relative to Intel microarchitecture code name Westmere summarized in Table 18-25.

Table 18-25. Core PMU Comparison

Box	Intel® microarchitecture code name Sandy Bridge	Intel® microarchitecture code name Westmere	Comment
# of Fixed counters per thread	3	3	Use CPUID to enumerate # of counters.
# of general-purpose counters per core	8	8	
Counter width (R,W)	R:48 , W: 32/48	R:48, W:32	See Section 18.2.2.3.
# of programmable counters per thread	4 or (8 if a core not shared by two threads)	4	Use CPUID to enumerate # of counters.
Precise Event Based Sampling (PEBS) Events	See Table 18-27	See Table 18-10	IA32_PMC4-IA32_PMC7 do not support PEBS.
PEBS-Load Latency	See Section 18.9.4.2; Data source encoding, STLB miss encoding, Lock transaction encoding	Data source encoding	
PEBS-Precise Store	Section 18.9.4.3	No	
PEBS-PDIR	yes (using precise INST_RETIRED.ALL)	No	
Off-core Response Event	MSR 1A6H and 1A7H; Extended request and response types	MSR 1A6H and 1A7H, limited response types	Nehalem supports 1A6H only.



Basic perf Options

Dynamic Optimization using the Performance Counters for CPU and Memory Profiling.

>perf list

List of pre-defined events (to be used in -e):

cpu-cycles OR cycles	[Hardware event]
instructions	[Hardware event]
cache-references	[Hardware event]
cache-misses	[Hardware event]
branch-instructions OR branches	[Hardware event]
branch-misses	[Hardware event]
bus-cycles	[Hardware event]
stalled-cycles-frontend OR idle-cycles-frontend	[Hardware event]
stalled-cycles-backend OR idle-cycles-backend	[Hardware event]
ref-cycles	[Hardware event]
cpu-clock	[Software event]
task-clock	[Software event]
page-faults OR faults	[Software event]
context-switches OR cs	[Software event]
cpu-migrations OR migrations	[Software event]
minor-faults	[Software event]
major-faults	[Software event]
alignment-faults	[Software event]
emulation-faults	[Software event]
L1-dcache-loads	[Hardware cache event]
L1-dcache-load-misses	[Hardware cache event]
L1-dcache-stores	[Hardware cache event]
L1-dcache-store-misses	[Hardware cache event]
L1-dcache-prefetches	[Hardware cache event]

...

<https://perf.wiki.kernel.org/index.php/Tutorial>

<http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html>



Perf Linux Performance Tool

```
>gcc -O3 main.c
>time ./a.out
sum = 400000034998780787062429554585290932224.00
0.376u 0.000s 0:00.37 100.0%    0+0k 16+0io 0pf+0w
>perf stat ./a.out
sum = 400000034998780787062429554585290932224.00
```

Performance counter stats for './a.out':

373.038548 task-clock	#	0.998 CPUs utilized	
39 context-switches	#	0.105 K/sec	
0 cpu-migrations	#	0.000 K/sec	
129 page-faults	#	0.346 K/sec	
1,108,828,204 cycles	#	2.972 GHz	[50.12%]
<not supported> stalled-cycles-frontend			
<not supported> stalled-cycles-backend			
2,600,740,775 instructions	#	2.35 insns per cycle	[75.10%]
400,189,661 branches	#	1072.784 M/sec	[75.07%]
11,156 branch-misses	#	0.00% of all branches	[74.87%]
0.373782171 seconds time elapsed			



Perf multiple events

To measure more than one event, simply provide a comma-separated list with no space:

```
perf stat -e cycles,instructions,cache-misses [...]
```

the kernel will automatically multiplex the number of events that need to be provided.

multiplexing and scaling events

If there are more events than counters, the kernel uses time multiplexing (switch frequency = HZ, generally 100 or 1000) to give each event a chance to access the monitoring hardware. Multiplexing only applies to PMU events. With multiplexing, an event is not measured all the time. At the end of the run, the tool scales the count based on total time enabled vs time running.

The actual formula is:

```
final_count = raw_count * time_enabled/time_running
```

Events are currently managed in round-robin fashion. Therefore each event will eventually get a chance to run. To avoid scaling (in the presence of only one active perf_event user), one can try and reduce the number of events. The following table provides the number of counters for a few common processors:

Processor	Generic counters	Fixed counters
Intel Core	2	3
Intel Nehalem	4	3

<https://perf.wiki.kernel.org/index.php/Tutorial>



Perf Scaling

```
>perf stat -e cycles,cycles ./a.out  
sum = 400000034998780787062429554585290932224.00
```

Performance counter stats for './a.out':

1,232,324,077 cycles	#	0.000 GHz
1,232,336,337 cycles	#	0.000 GHz

0.590856656 seconds time elapsed

```
>perf stat -e cycles,cycles,cycles,cycles ./a.out  
sum = 400000034998780787062429554585290932224.00
```

Performance counter stats for './a.out':

1,204,204,939 cycles	#	0.000 GHz	[74.97%]
1,205,541,066 cycles	#	0.000 GHz	[74.84%]
1,205,566,744 cycles	#	0.000 GHz	[74.84%]
1,196,072,400 cycles	#	0.000 GHz	[75.51%]

0.557318767 seconds time elapsed



Perf Linux Performance Tool

```
>perf stat -r 10 ./a.out # Run 10 Times
sum = 400000034998780787062429554585290932224.00
sum = 400000034998780787062429554585290932224.00
...
```

Performance counter stats for './a.out' (10 runs):

372.392419 task-clock	#	0.998 CPUs utilized	(+- 0.09%)
38 context-switches	#	0.102 K/sec	(+- 0.55%)
1 cpu-migrations	#	0.001 K/sec	(+- 44.72%)
130 page-faults	#	0.348 K/sec	(+- 0.12%)
1,108,840,183 cycles	#	2.978 GHz	(+- 0.02%) [50.04%]
<not supported> stalled-cycles-frontend			
<not supported> stalled-cycles-backend			
2,601,117,799 instructions	#	2.35 insns per cycle	(+- 0.01%) [75.02%]
400,349,107 branches	#	1075.073 M/sec	(+- 0.01%) [75.01%]
10,595 branch-misses	#	0.00% of all branches	(+- 2.17%) [75.07%]
0.373017983 seconds time elapsed			



Perf Linux Performance Tool

USER LEVEL INSTRUCTIONS

```
>perf stat -e instructions:u ./a.out  
sum = 400000034998780787062429554585290932224.00
```

Performance counter stats for './a.out':

```
2,600,096,449 instructions:u          #    0.00  insns per cycle
```

```
0.375571376 seconds time elapsed
```

KERNEL LEVEL INSTRUCTIONS

```
>perf stat -e instructions:k ./a.out  
sum = 400000034998780787062429554585290932224.00
```

Performance counter stats for './a.out':

```
1,867,178 instructions:k            #    0.00  insns per cycle
```

```
0.377330791 seconds time elapsed
```



Matrix Multiplication Examples

```
gcc -Werror -Wall matrix_serial_ver1.c -o matrix_serial_ver1.out  
>./matrix_serial_ver1.out  
Elapsed Time: 6.32 Sec.
```

```
>gcc -Werror -Wall matrix_serial_ver2.c -o matrix_serial_ver2.out  
>./matrix_serial_ver2.out  
Elapsed Time: 5.38 Sec.
```

```
>gcc -Werror -Wall matrix_serial_ver3.c -o matrix_serial_ver3.out  
>./matrix_serial_ver3.out  
Elapsed Time: 4.77 Sec.
```

```
>gcc -Werror -Wall matrix_serial_ver4.c -o matrix_serial_ver4.out  
./matrix_serial_ver4.out  
Elapsed Time: 4.60 Sec.
```



Matrix Multiplication Examples

```
>perf stat -e cycles -e instructions -e cache-references -e cache-misses ./matrix_serial_ver1.out
```

Elapsed Time: 6.35 Sec.

Performance counter stats for './matrix_serial_ver1.out':

18,929,166,306 cycles	#	0.000 GHz	[50.00%]
34,062,608,328 instructions	#	1.80 insns per cycle	[75.01%]
1,066,881,565 cache-references			[74.99%]
83,905 cache-misses	#	0.008 % of all cache refs	[75.02%]
6.362608904 seconds time elapsed			

```
>perf stat -e cycles -e instructions -e cache-references -e cache-misses ./matrix_serial_ver2.out
```

Elapsed Time: 5.40 Sec.

Performance counter stats for './matrix_serial_ver2.out':

16,114,935,514 cycles	#	0.000 GHz	[50.00%]
34,051,559,029 instructions	#	2.11 insns per cycle	[75.01%]
64,741,737 cache-references			[74.99%]
24,675 cache-misses	#	0.038 % of all cache refs	[75.02%]

5.418502175 seconds time elapsed



Matrix Multiplication Examples

```
>gcc -Werror -Wall matrix_serial_ver3.c -o matrix_serial_ver3.out  
> stat -e cycles -e instructions -e cache-references -e cache-misses ./matrix_serial_ver3.out
```

Elapsed Time: 4.79 Sec.

Performance counter stats for './matrix_serial_ver3.out':

14,296,893,562 cycles	#	0.000 GHz	[50.00%]
20,059,297,912 instructions	#	1.40 insns per cycle	[74.99%]
1,067,327,910 cache-references			[75.02%]
107,539 cache-misses	#	0.010 % of all cache refs	[74.99%]
4.804650637 seconds time elapsed			

```
>gcc -Werror -Wall matrix_serial_ver4.c -o matrix_serial_ver4.out  
> stat -e cycles -e instructions -e cache-references -e cache-misses ./matrix_serial_ver4.out
```

Elapsed Time: 4.55 Sec.

Performance counter stats for './matrix_serial_ver4.out':

13,582,459,137 cycles	#	0.000 GHz	[50.01%]
26,075,199,666 instructions	#	1.92 insns per cycle	[75.02%]
64,323,587 cache-references			[74.99%]
30,268 cache-misses	#	0.047 % of all cache refs	[75.00%]
4.566277059 seconds time elapsed			



gcc Optimizations and Branch Prediction

```
gcc main.c -O0 -o main.out
```

```
>perf stat -e cycles -e instructions -e branches -e branch-misses ./main.out
```

```
sum = 400000034998780787062429554585290932224.00
```

```
Performance counter stats for './main.out':
```

4,120,356,326 cycles	#	0.000 GHz	[49.99%]
5,306,382,355 instructions	#	1.29 insns per cycle	[74.96%]
899,775,764 branches			[75.03%]
8,742,590 branch-misses	#	0.97% of all branches	[75.03%]
1.387120365 seconds time elapsed			

```
>gcc main.c -O1 -o main.out
```

```
>perf stat -e cycles -e instructions -e branches -e branch-misses ./main.out
```

```
sum = 400000034998780787062429554585290932224.00
```

```
Performance counter stats for './main.out':
```

1,950,230,142 cycles	#	0.000 GHz	[50.07%]
2,782,472,831 instructions	#	1.43 insns per cycle	[75.05%]
600,311,976 branches			[75.02%]
1,368,519 branch-misses	#	0.23% of all branches	[74.95%]
0.662207246 seconds time elapsed			



gcc Optimizations and Branch Prediction

```
>gcc main.c -O2 -o main.out
>perf stat -e cycles -e instructions -e branches -e branch-misses ./main.out
sum = 400000034998780787062429554585290932224.00
Performance counter stats for './main.out':
  1,109,248,454 cycles                #    0.000 GHz                [49.96%]
  2,599,913,162 instructions          #    2.34  insns per cycle    [75.12%]
  400,489,432 branches                #                               [75.12%]
    11,973 branch-misses              #    0.00% of all branches    [74.98%]
  0.379008882 seconds time elapsed
```

```
main.c -O3 -o main.out
stat -e cycles -e instructions -e branches -e branch-misses ./main.out
sum = 400000034998780787062429554585290932224.00
Performance counter stats for './main.out':
  1,108,212,355 cycles                #    0.000 GHz                [50.09%]
  2,603,070,814 instructions          #    2.35  insns per cycle    [75.09%]
  399,947,490 branches                #                               [75.11%]
    12,176 branch-misses              #    0.00% of all branches    [74.93%]
  0.378495972 seconds time elapsed
```

<https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html>



Perf annotate

Not Working in the Labs.

perf annotate can generate sourcecode level information if the application is compiled with `-ggdb`.

```
>ssh cs6472
```

```
>gcc -Werror -Wall -ggdb matrix_serial_ver1.c -o matrix_serial_ver1.out
```

```
> perf_3.2.0-38 record ./matrix_serial_ver1.out
```

WARNING: Kernel address maps (/proc/{kallsyms,modules}) are restricted,
check /proc/sys/kernel/kptr_restrict.

Samples in kernel functions may not be resolved if a suitable vmlinux
file is not found in the buildid cache or in the vmlinux path.

Samples in kernel modules won't be resolved at all.

If some relocation was applied (e.g. kexec) symbols may be misresolved
even with a suitable vmlinux or kallsyms file.

Elapsed Time: 6.40 Sec.

[perf record: Woken up 4 times to write data]

[perf record: Captured and wrote 0.982 MB perf.data (~42916 samples)]

```
> perf_3.2.0-38 annotate
```



>perf annotate

```
103ws30.in.cs.ucy.ac.cy - PuTTY
main
β
β 0000000000400511 <main>:
β   return x;
β   }
β
β
β
β int main (void)
β {
β   push   %rbp
β   mov    %rsp,%rbp
β   sub   $0x10,%rsp
β   double sum = 0.0;
β   mov   $0x0,%eax
β   mov   %rax,-0x10(%rbp)
β   unsigned i;
β
β   for (i = 0; i < 100000000; i++)
β   movl  $0x0,-0x4(%rbp)
β   β jmp   71
β   {
β       sum += powern (i, i % 5);
7.58 β1a:  mov   -0x4(%rbp),%ecx
β   mov   $0xc000000d,%edx
β   mov   %ecx,%eax
β   mul   %edx
13.68 β   shr   $0x2,%edx
2.43 β   mov   %edx,%eax
8.53 β   shl   $0x2,%eax
2.07 β   add   %edx,%eax
2.78 β   mov   %ecx,%edx
β   sub   %eax,%edx
9.83 β   mov   -0x4(%rbp),%eax
β   test  %rax,%rax
β   β js   43
β   cvtsi2 %rax,%xmm0
9.77 β βββ jmp   58
β43:β   mov   %rax,%rcx
β   β shr   %rcx
β   β and  $0x1,%eax
β   β or   %rax,%rcx
β   β cvtsi2 %rcx,%xmm0
β   β addsd %xmm0,%xmm0
7.05 β58:ββββ mov   %edx,%edi
β   β callq powern
8.23 β   movsd -0x10(%rbp),%xmm1
β   addsd %xmm1,%xmm0
19.91 β   movsd %xmm0,-0x10(%rbp)
press 'h' for help on key bindings
```



Assembly

```
>gcc -S main.c
```

```
>vi main.s
```

```
>gcc main.s -o main.s.out
```

```
103ws30.in.cs.ucy.ac.cy - PuTTY
.LC2:
    .string "sum = %.2f\n"
    .text
.globl main
.type   main, @function
main:
.LFB1:
    .cfi_startproc
    pushq   %rbp
    .cfi_def_cfa_offset 16
    .cfi_offset 6, -16
    movq   %rsp, %rbp
    .cfi_def_cfa_register 6
    subq   $16, %rsp
    movl   $0, %eax
    movq   %rax, -16(%rbp)
    movl   $0, -4(%rbp)
    jmp   .L6

.L9:
    movl   -4(%rbp), %ecx
    movl   $-858993459, %edx
    movl   %ecx, %eax
    mull   %edx
    shr   $2, %edx
    movl   %edx, %eax
    sall  $2, %eax
    addl   %edx, %eax
    movl   %ecx, %edx
    subl   %eax, %edx
    mov   -4(%rbp), %eax
    testq  %rax, %rax
    js    .L7
    cvtsi2sdq   %rax, %xmm0
    jmp   .L8

.L7:
    movq   %rax, %rcx
    shrq   %rcx
    andl   $1, %eax
38,5 57%
```



>objdump -d
main.out

>hexdump -C
main.out

```
103ws30.in.cs.ucy.ac.cy - PuTTY
40050a: f2 0f 10 45 d8      movsd  -0x28(%rbp),%xmm0
40050f: c9                  leaveq
400510: c3                  retq

000000000400511 <main>:
400511: 55                  push  %rbp
400512: 48 89 e5            mov   %rsp,%rbp
400515: 48 83 ec 10         sub  $0x10,%rsp
400519: b8 00 00 00 00     mov  $0x0,%eax
40051e: 48 89 45 f0         mov  %rax,-0x10(%rbp)
400522: c7 45 fc 00 00 00 00 movl  $0x0,-0x4(%rbp)
400529: eb 57              jmp   400582 <main+0x71>
40052b: 8b 4d fc            mov  -0x4(%rbp),%ecx
40052e: ba cd cc cc cc     mov  $0xcccccccd,%edx
400533: 89 c8              mov  %ecx,%eax
400535: f7 e2              mul  %edx
400537: c1 ea 02           shr  $0x2,%edx
40053a: 89 d0              mov  %edx,%eax
40053c: c1 e0 02           shl  $0x2,%eax
40053f: 01 d0              add  %edx,%eax
400541: 89 ca              mov  %ecx,%edx
400543: 29 c2              sub  %eax,%edx
400545: 8b 45 fc            mov  -0x4(%rbp),%eax
400548: 48 85 c0           test %rax,%rax
40054b: 78 07              js   400554 <main+0x43>
40054d: f2 48 0f 2a c0     cvtsi2sd %rax,%xmm0
400552: eb 15              jmp  400569 <main+0x58>
400554: 48 89 c1           mov  %rax,%rcx
400557: 48 d1 e9           shr  %rcx
40055a: 83 e0 01           and  $0x1,%eax
40055d: 48 09 c1           or   %rax,%rcx
400560: f2 48 0f 2a c1     cvtsi2sd %rcx,%xmm0
400565: f2 0f 58 c0         addsd %xmm0,%xmm0
400569: 89 d7              mov  %edx,%edi
40056b: e8 54 ff ff ff     callq 4004c4 <powern>
400570: f2 0f 10 4d f0     movsd -0x10(%rbp),%xmm1
400575: f2 0f 58 c1         addsd %xmm1,%xmm0
400579: f2 0f 11 45 f0     movsd %xmm0,-0x10(%rbp)
40057e: 83 45 fc 01         addl $0x1,-0x4(%rbp)
400582: 81 7d fc ff e0 f5 05 cmpl $0x5f5e0ff,-0x4(%rbp)
400589: 76 a0              jbe  40052b <main+0x1a>
40058b: b8 a8 06 40 00     mov  $0x4006a8,%eax
400590: f2 0f 10 45 f0     movsd -0x10(%rbp),%xmm0
400595: 48 89 c7           mov  %rax,%rdi
400598: b8 01 00 00 00     mov  $0x1,%eax
40059d: e8 16 fe ff ff     callq 4003b8 <printf@plt>
4005a2: b8 00 00 00 00     mov  $0x0,%eax
4005a7: c9                  leaveq
```



gdb

```
gcc -ggdb main.c -o main.out
```

```
bash
```

```
gdb main.out
```

```
    b main (breakpoint at main)
```

```
    r      (run)
```

```
disassemble
```

```
s
```

```
s      (step)
```

```
bt      (Backtrace)
```

```
set disassemble-next-line on
```

```
103ws30.in.cs.ucy.ac.cy - PuTTY
Undefined info command: "main". Try "help info".
(gdb) disassemble
Dump of assembler code for function main:
0x000000000400511 <+0>:   push   %rbp
0x000000000400512 <+1>:   mov    %rsp,%rbp
0x000000000400515 <+4>:   sub   $0x10,%rsp
=>0x000000000400519 <+8>:   mov   $0x0,%eax
0x00000000040051e <+13>:  mov   %rax,-0x10(%rbp)
0x000000000400522 <+17>:  movl  $0x0,-0x4(%rbp)
0x000000000400529 <+24>:  jmp   0x400582 <main+113>
0x00000000040052b <+26>:  mov   -0x4(%rbp),%ecx
0x00000000040052e <+29>:  mov   $0xcccccccd,%edx
0x000000000400533 <+34>:  mov   %ecx,%eax
0x000000000400535 <+36>:  mul  %edx
0x000000000400537 <+38>:  shr  $0x2,%edx
0x00000000040053a <+41>:  mov   %edx,%eax
0x00000000040053c <+43>:  shl  $0x2,%eax
0x00000000040053f <+46>:  add  %edx,%eax
0x000000000400541 <+48>:  mov   %ecx,%edx
0x000000000400543 <+50>:  sub  %eax,%edx
0x000000000400545 <+52>:  mov  -0x4(%rbp),%eax
0x000000000400548 <+55>:  test  %rax,%rax
0x00000000040054b <+58>:  js   0x400554 <main+67>
0x00000000040054d <+60>:  cvtsi2sd %rax,%xmm0
0x000000000400552 <+65>:  jmp  0x400569 <main+88>
0x000000000400554 <+67>:  mov  %rax,%rcx
0x000000000400557 <+70>:  shr  %rcx
0x00000000040055a <+73>:  and  $0x1,%eax
0x00000000040055d <+76>:  or   %rax,%rcx
0x000000000400560 <+79>:  cvtsi2sd %rcx,%xmm0
0x000000000400565 <+84>:  addsd %xmm0,%xmm0
0x000000000400569 <+88>:  mov  %edx,%edi
0x00000000040056b <+90>:  callq 0x4004c4 <powern>
0x000000000400570 <+95>:  movsd -0x10(%rbp),%xmm1
0x000000000400575 <+100>: addsd %xmm1,%xmm0
0x000000000400579 <+104>: movsd %xmm0,-0x10(%rbp)
0x00000000040057e <+109>: addl $0x1,-0x4(%rbp)
0x000000000400582 <+113>: cmpl $0x5f5e0ff,-0x4(%rbp)
0x000000000400589 <+120>: jbe  0x40052b <main+26>
0x00000000040058b <+122>: mov  $0x4006a8,%eax
0x000000000400590 <+127>: movsd -0x10(%rbp),%xmm0
0x000000000400595 <+132>: mov  %rax,%rdi
0x000000000400598 <+135>: mov  $0x1,%eax
0x00000000040059d <+140>: callq 0x4003b8 <printf@plt>
0x0000000004005a2 <+145>: mov  $0x0,%eax
0x0000000004005a7 <+150>: leaveq
0x0000000004005a8 <+151>: retq
End of assembler dump.
(gdb) █
```

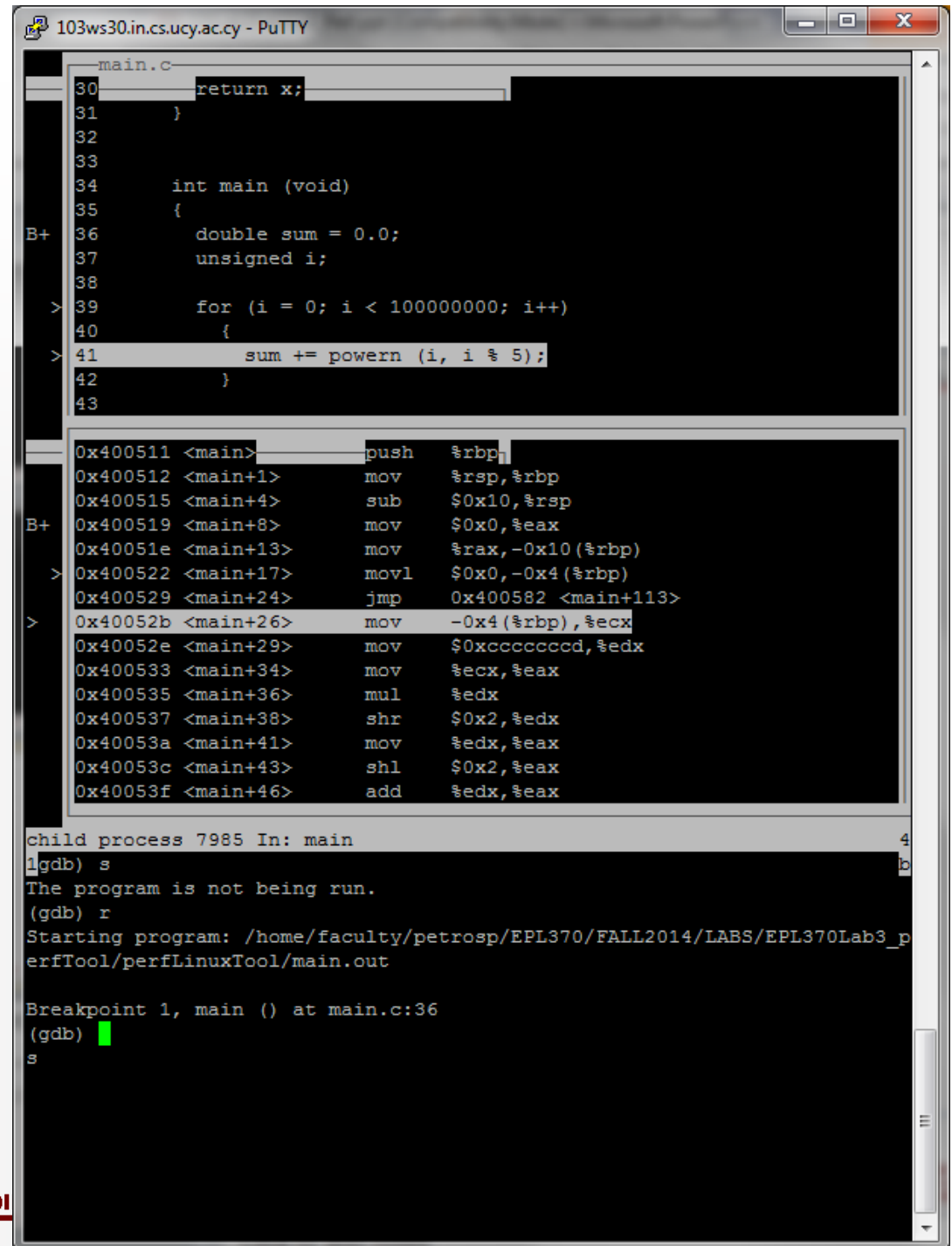


gdb Disassemble next line

```
setenv SHELL /bin/bash
gdb a.out
GNU gdb (GDB) Red Hat Enterprise Linux (7.2-56.el6)
..
(gdb) b main
(gdb) set disassemble-next-line on (gdb) s (gdb) r
Starting program: /home/faculty/petrosp/EPL370/FALL2012/LABS/EPL370Lab5/a.out
Breakpoint 1, main () at gprof-prog1.c:49
49      init( x );
=> 0x0000000004005bc <main+11>:      48 8d 85 60 fe ff ff      lea    -0x1a0(%rbp),%rax
    0x0000000004005c3 <main+18>:      48 89 c7                  mov    %rax,%rdi
    0x0000000004005c6 <main+21>:      e8 f9 fe ff ff          callq 0x4004c4 <init>
Missing separate debuginfos, use: debuginfo-install glibc-2.12-1.80.el6_3.4.x86_64
(gdb) s
init (x=0x7fffffffdf90) at gprof-prog1.c:8
8      for ( i = 0; i < 100; i++ ) {
=> 0x0000000004004cc <init+8>:  c7 45 fc 00 00 00 00      movl  $0x0,-0x4(%rbp)
    0x0000000004004d3 <init+15>:  eb 16                    jmp   0x4004eb <init+39>
(gdb)
```



```
gcc -ggdb main.c -o main.out
bash
gdb main.out
b main
r
layout asm
ctrl-x 2
s
```



```
103ws30.in.cs.ucy.ac.cy - PuTTY
main.c
30     return x;
31     }
32
33
34     int main (void)
35     {
B+ 36     double sum = 0.0;
37     unsigned i;
38
39     for (i = 0; i < 1000000000; i++)
40     {
41     sum += powern (i, i % 5);
42     }
43
0x400511 <main>    push    %rbp
0x400512 <main+1>    mov     %rsp,%rbp
0x400515 <main+4>    sub    $0x10,%rsp
B+ 0x400519 <main+8>    mov    $0x0,%eax
0x40051e <main+13>   mov    %rax,-0x10(%rbp)
> 0x400522 <main+17>   movl   $0x0,-0x4(%rbp)
0x400529 <main+24>   jmp    0x400582 <main+113>
> 0x40052b <main+26>   mov    -0x4(%rbp),%ecx
0x40052e <main+29>   mov    $0xcccccccd,%edx
0x400533 <main+34>   mov    %ecx,%eax
0x400535 <main+36>   mul   %edx
0x400537 <main+38>   shr   $0x2,%edx
0x40053a <main+41>   mov   %edx,%eax
0x40053c <main+43>   shl   $0x2,%eax
0x40053f <main+46>   add   %edx,%eax

child process 7985 In: main
(gdb) s
The program is not being run.
(gdb) r
Starting program: /home/faculty/petrosp/EPL370/FALL2014/LABS/EPL370Lab3_perfTool/perfLinuxTool/main.out

Breakpoint 1, main () at main.c:36
(gdb) █
s
```



top

```
103ws30.in.cs.ucy.ac.cy - PuTTY
top - 14:41:59 up 4 days, 5:50, 3 users, load average: 0.46, 1.13, 1.12
Tasks: 208 total, 2 running, 206 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.7%us, 0.3%sy, 0.0%ni, 98.8%id, 0.2%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 3852492k total, 1958740k used, 1893752k free, 96892k buffers
Swap: 4192956k total, 0k used, 4192956k free, 737664k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 1824 root        20   0 188m  69m  36m  S   1.3   1.8   1:24.31 Xorg
 3177 agoset01    20   0 545m  29m  14m  S   1.0   0.8   0:20.20 emacs
10092 agoset01    20   0 321m  25m  13m  S   0.3   0.7   0:01.15 emacs
10387 agoset01    20   0 319m  24m  12m  S   0.3   0.6   0:00.42 emacs
10691 petrosp     20   0 26040 1420 1028  R   0.3   0.0   0:00.03 top
   1 root        20   0 21436 1472 1148  S   0.0   0.0   0:00.42 init
   2 root        20   0   0     0     0  S   0.0   0.0   0:00.01 kthreadd
   3 root        RT   0   0     0     0  S   0.0   0.0   0:01.44 migration/0
   4 root        20   0   0     0     0  S   0.0   0.0   0:00.17 ksoftirqd/0
   5 root        RT   0   0     0     0  S   0.0   0.0   0:00.00 migration/0
   6 root        RT   0   0     0     0  S   0.0   0.0   0:00.28 watchdog/0
   7 root        RT   0   0     0     0  S   0.0   0.0   0:01.30 migration/1
   8 root        RT   0   0     0     0  S   0.0   0.0   0:00.00 migration/1
   9 root        20   0   0     0     0  S   0.0   0.0   0:00.13 ksoftirqd/1
  10 root        RT   0   0     0     0  S   0.0   0.0   0:00.23 watchdog/1
  11 root        20   0   0     0     0  S   0.0   0.0   0:11.89 events/0
  12 root        20   0   0     0     0  S   0.0   0.0   0:31.15 events/1
  13 root        20   0   0     0     0  S   0.0   0.0   0:00.00 cgroup
  14 root        20   0   0     0     0  S   0.0   0.0   0:00.00 khelper
  15 root        20   0   0     0     0  S   0.0   0.0   0:00.00 netns
  16 root        20   0   0     0     0  S   0.0   0.0   0:00.00 async/mgr
  17 root        20   0   0     0     0  S   0.0   0.0   0:00.00 pm
  18 root        20   0   0     0     0  S   0.0   0.0   0:00.99 sync_supers
  19 root        20   0   0     0     0  S   0.0   0.0   0:00.66 bdi-default
  20 root        20   0   0     0     0  S   0.0   0.0   0:00.00 kintegrityd/0
  21 root        20   0   0     0     0  S   0.0   0.0   0:00.00 kintegrityd/1
  22 root        20   0   0     0     0  S   0.0   0.0   0:00.68 kblockd/0
  23 root        20   0   0     0     0  S   0.0   0.0   0:00.66 kblockd/1
```



GNUPlot

<http://www.gnuplot.info/>

<http://www.gnuplot.info/documentation.html>

(log in to **cs6472** or any other machine that has gnuplot, and run **gnuplot**)

```
plot "data.txt" using 1:2 title 'Column 2', "data.txt" using 1:3 title 'Column 3'
```

```
plot "data.txt" using 1:2 title 'Column 2'
```

```
gnuplot> set term png (will produce .png output)
```

```
gnuplot> set output "printme.png" (output to any filename you use)
```

```
gnuplot> replot
```

