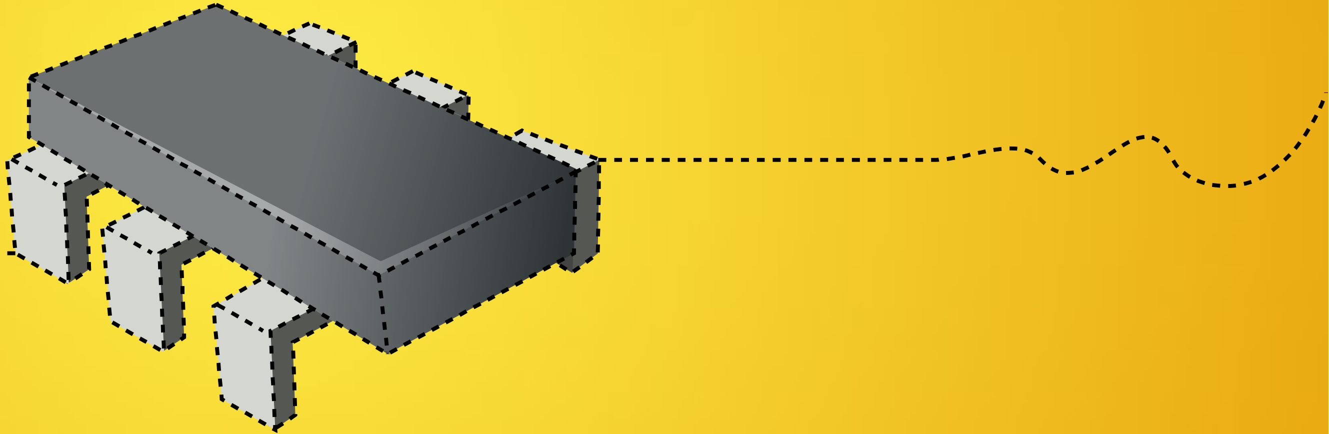
Hardware-assisted software tracing



Adrien Vergé
adrienverge@gmail.com

POLYTECHNIQUE MONTRÉAL

Embedded Linux Conference 2014

talk

about

tracing

improve tracing

using

hardware

1 Tracing
2 Hardware
3 Improvements

1

Tracing

“a technique used to understand what is going on in a system in order to debug or monitor it”

recording events

from the kernel:	IRQ handlers, system calls, scheduling activity, network activity, etc.
in user-space:	tracepoints inside your application

Why is my software crashing?

Where are the bottlenecks?

How to improve performance?

use less resources

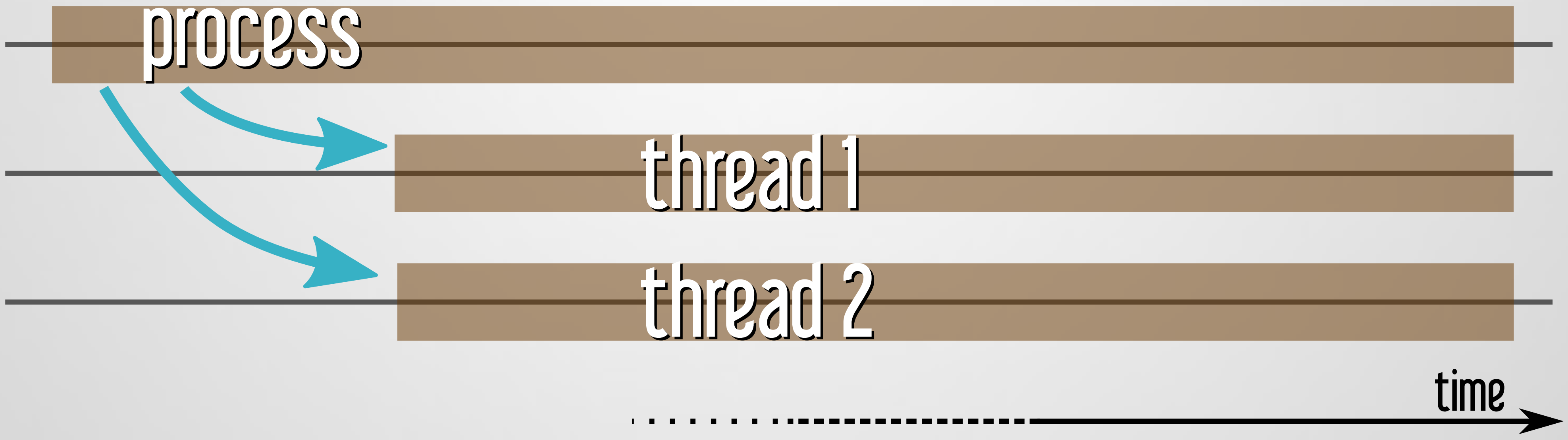
run faster

save battery

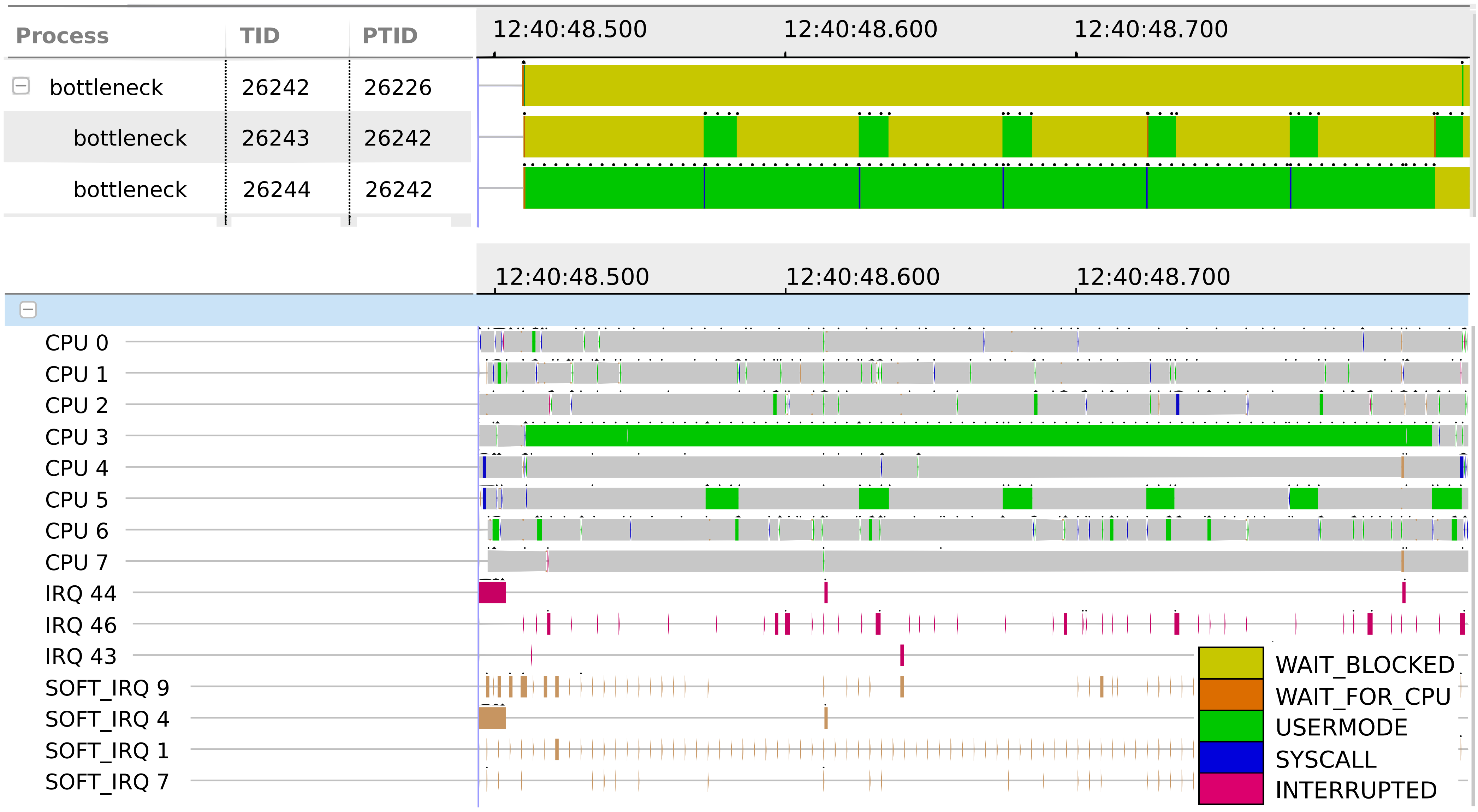
example

a process spawns 2 threads:

#1 produces chunks of data that #2 consumes



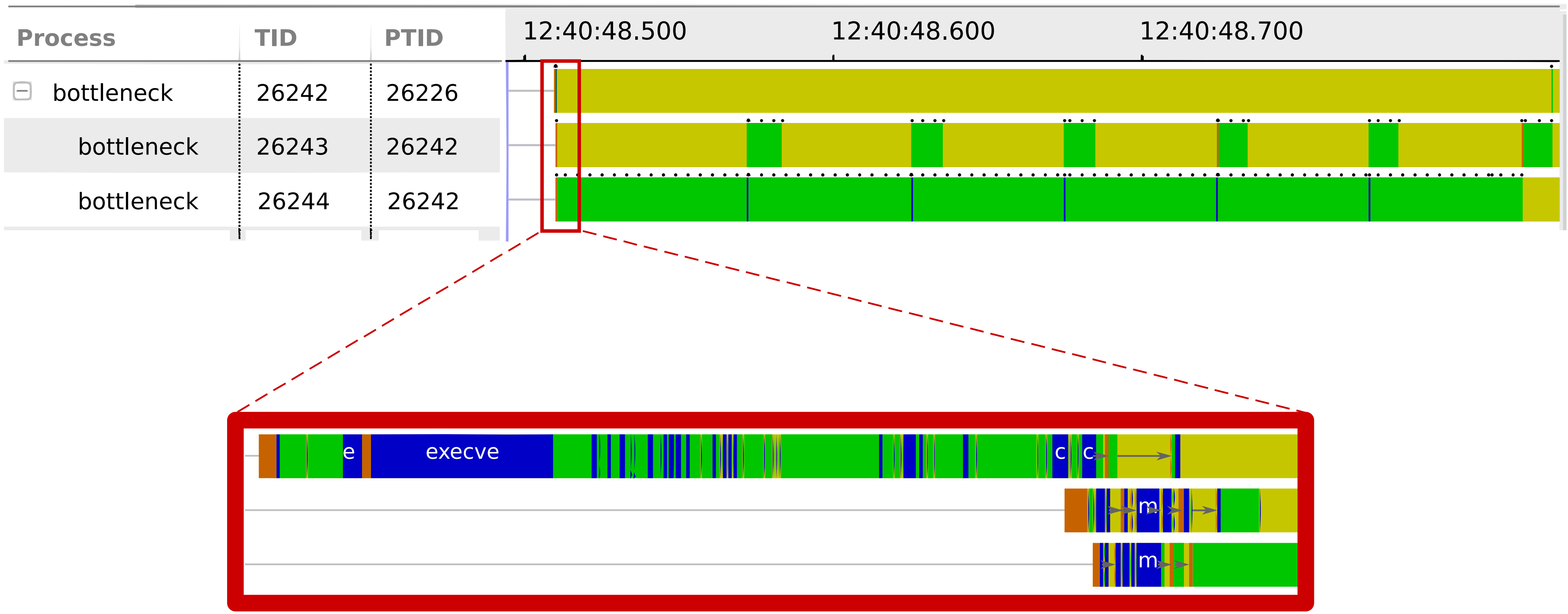
example: LTTng+TMF



Process	TID	PTID
⊟ bottleneck	26242	26226
bottleneck	26243	26242
bottleneck	26244	26242

🟨	WAIT_BLOCKED
🟧	WAIT_FOR_CPU
🟩	USERMODE
🟦	SYSCALL
🟥	INTERRUPTED

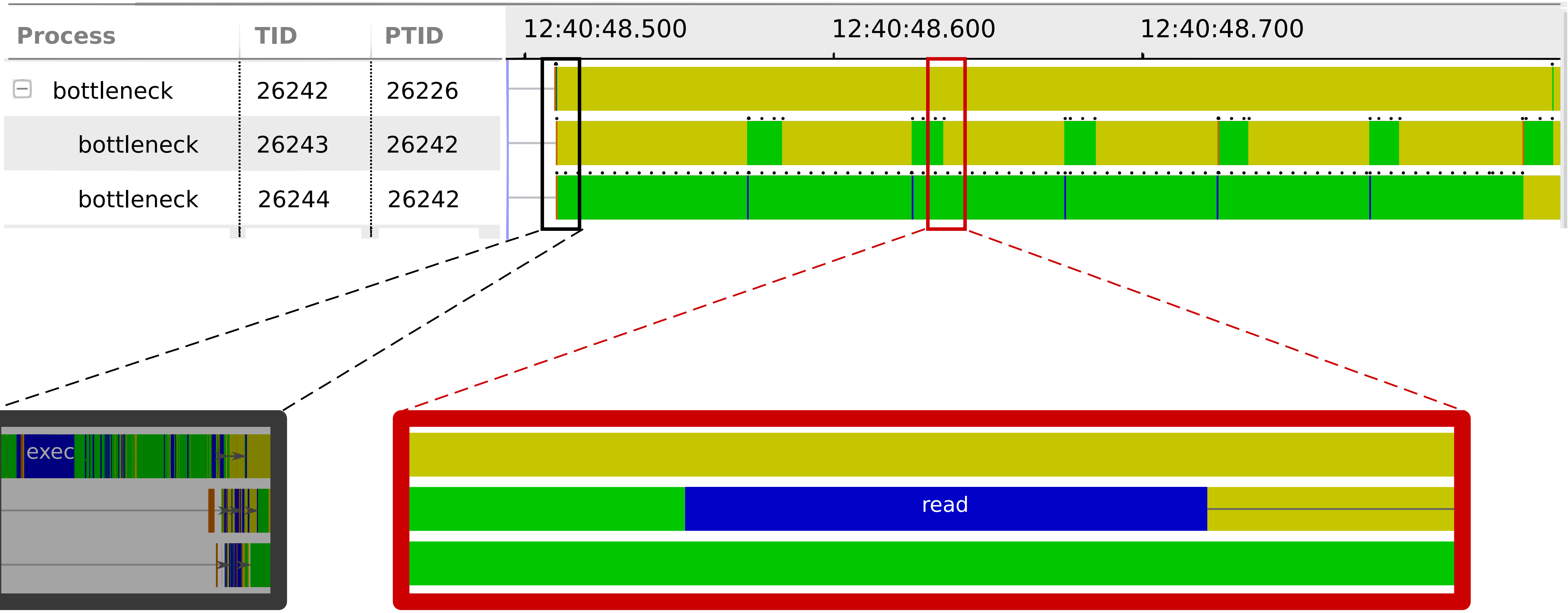
example: LTTng+TMF



Process	TID	PTID
bottleneck	26242	26226
bottleneck	26243	26242
bottleneck	26244	26242

🟨	WAIT_BLOCKED
🟧	WAIT_FOR_CPU
🟩	USERMODE
🟦	SYSCALL
🟥	INTERRUPTED

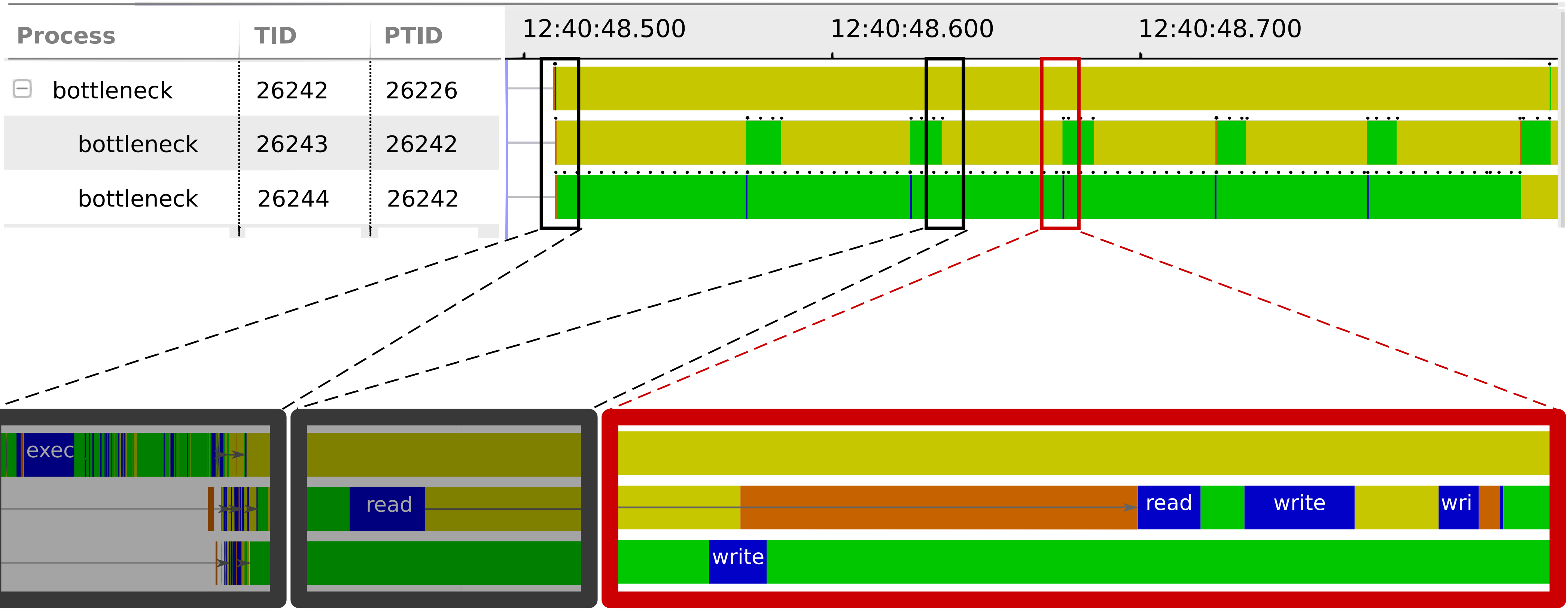
example: LTTng+TMF



Process	TID	PTID
⊟ bottleneck	26242	26226
bottleneck	26243	26242
bottleneck	26244	26242

🟨	WAIT_BLOCKED
🟧	WAIT_FOR_CPU
🟩	USERMODE
🟦	SYSCALL
🟥	INTERRUPTED

example: LTTng+TMF



Process	TID	PTID
⊟ bottleneck	26242	26226
bottleneck	26243	26242
bottleneck	26244	26242

🟨 WAIT_BLOCKED
🟧 WAIT_FOR_CPU
🟩 USERMODE
🟦 SYSCALL
🟥 INTERRUPTED

tracing:

recording events for use in further analysis

```
[   16.246595] Bluetooth: HCI device and connection manager initialized
[   16.246602] Bluetooth: HCI socket layer initialized
[   16.246605] Bluetooth: L2CAP socket layer initialized
[   16.246609] Bluetooth: SCO socket layer initialized
[   16.317299] Bluetooth: RFCOMM TTY layer initialized
[   16.317303] Bluetooth: RFCOMM socket layer initialized
[   16.317306] Bluetooth: RFCOMM ver 1.11
[   16.496886] Bluetooth: BNEP (Ethernet Emulation) ver 1.3
[   16.496889] Bluetooth: BNEP filters: protocol multicast
[   16.496897] Bluetooth: BNEP socket layer initialized
[   17.045998] NFSD: Using /var/lib/nfs/v4recovery as the NFSv4 state recovery directory
[   17.046487] NFSD: starting 90-second grace period (net ffffffff81886100)
[   17.327576] e1000e 0000:00:19.0: irq 42 for MSI/MSI-X
[   17.430960] e1000e 0000:00:19.0: irq 42 for MSI/MSI-X
[   17.431056] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[   19.452905] e1000e: eth0 NIC Link is Up 100 Mbps Half Duplex, Flow Control: Rx/Tx
[   19.452910] e1000e 0000:00:19.0 eth0: 10/100 speed: disabling TSO
[   19.456253] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[   21.325535] systemd-logind[3544]: New seat seat0.
[   21.344148] systemd-logind[3544]: Watching system buttons on /dev/input/event2 (Power Button)
[   21.344198] systemd-logind[3544]: Watching system buttons on /dev/input/event3 (Video Bus)
[   21.344242] systemd-logind[3544]: Watching system buttons on /dev/input/event1 (Power Button)
[   21.344788] systemd-logind[3544]: New session 1 of user Debian-gdm.
[   21.344859] systemd-logind[3544]: Linked /tmp/.X11-unix/X0 to /run/user/104/X11-display.
[   31.982144] systemd-logind[3544]: New session 2 of user adrien.
[   31.982187] systemd-logind[3544]: Linked /tmp/.X11-unix/X0 to /run/user/1000/X11-display.
[   37.509891] FAT-fs (sdb1): utf8 is not a recommended IO charset for FAT filesystems, filesystem will
```

tracing:

recording events for use in further analysis

So it's just logging?

```
[   16.246595] Bluetooth: HCI device and connection manager initialized
[   16.246602] Bluetooth: HCI socket layer initialized
[   16.246605] Bluetooth: L2CAP socket layer initialized
[   16.246609] Bluetooth: SCO socket layer initialized
[   16.317299] Bluetooth: RFCOMM TTY layer initialized
[   16.317303] Bluetooth: RFCOMM socket layer initialized
[   16.317306] Bluetooth: RFCOMM ver 1.11
[   16.496886] Bluetooth: BNEP (Ethernet Emulation) ver 1.3
[   16.496889] Bluetooth: BNEP filters: protocol multicast
[   16.496897] Bluetooth: BNEP socket layer initialized
[   17.045998] NFSD: Using /var/lib/nfs/v4recovery as the NFSv4 state recovery directory
[   17.046487] NFSD: starting 90-second grace period (net ffffffff81886100)
[   17.327576] e1000e 0000:00:19.0: irq 42 for MSI/MSI-X
[   17.430960] e1000e 0000:00:19.0: irq 42 for MSI/MSI-X
[   17.431056] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[   19.452905] e1000e: eth0 NIC Link is Up 100 Mbps Half Duplex, Flow Control: Rx/Tx
[   19.452910] e1000e 0000:00:19.0 eth0: 10/100 speed: disabling TSO
[   19.456253] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[   21.325535] systemd-logind[3544]: New seat seat0.
[   21.344148] systemd-logind[3544]: Watching system buttons on /dev/input/event2 (Power Button)
[   21.344198] systemd-logind[3544]: Watching system buttons on /dev/input/event3 (Video Bus)
[   21.344242] systemd-logind[3544]: Watching system buttons on /dev/input/event1 (Power Button)
[   21.344788] systemd-logind[3544]: New session 1 of user Debian-gdm.
[   21.344859] systemd-logind[3544]: Linked /tmp/.X11-unix/X0 to /run/user/104/X11-display.
[   31.982144] systemd-logind[3544]: New session 2 of user adrien.
[   31.982187] systemd-logind[3544]: Linked /tmp/.X11-unix/X0 to /run/user/1000/X11-display.
[   37.509891] FAT-fs (sdb1): utf8 is not a recommended IO charset for FAT filesystems, filesystem will
```

tracing vs. logging

compact binary trace format
buffering — avoid disk IO
lockless algorithms
low-level optimizations

result : ~200 µs vs. ~200 ns / event

tracing users

heavy workload servers

intrusion detection

IBM

Google

Autodesk

OPAL-RT

CAE

real-time

tracing users

heavy workload servers

intrusion detection

Google

IBM

Siemens

STMicroelectronics

Wind River

Freescale

Autodesk

Montavista

OPAL-RT

CAE

Ericsson

Nokia

real-time

embedded systems

tracing users

heavy workload servers

intrusion detection

IBM

Google

Siemens

STMicroelectronics

Wind River

Freescale

Autodesk

Montavista

YOU!

OPAL-RT

CAE

Ericsson

Nokia

real-time

embedded systems

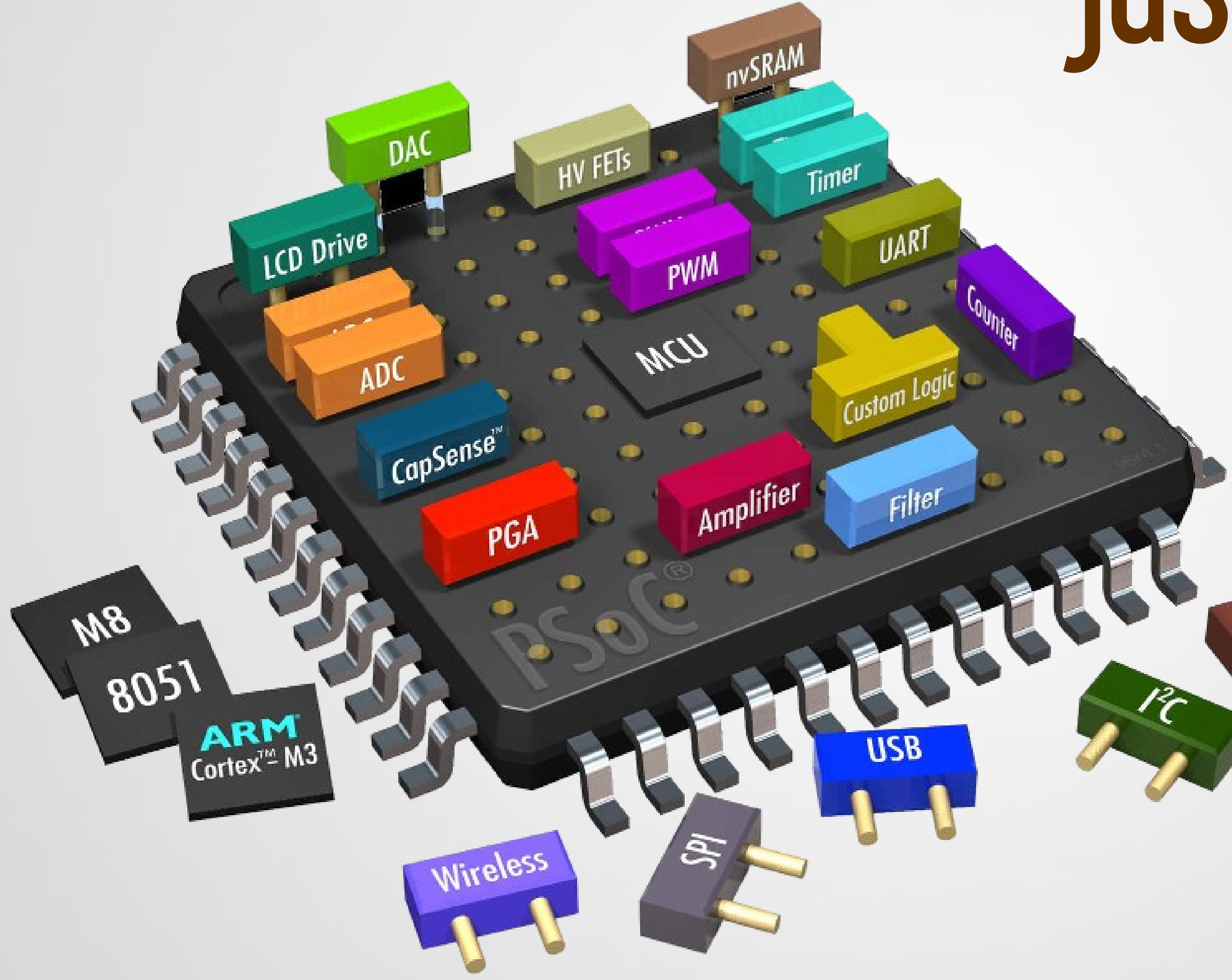
Beyond Heisenberg: *observe without altering*

needs

- perform light (size) and fast (time)
- don't pollute memory space
- thousands of events / s

2 Hardware

Microchips are no longer just CPUs



credit: ARM

lots of tracing units

Freescale (PowerPC)
Nexus Program Trace,
Data Acquisition…

ARM
CoreSight
ETM, ETB, STM…

Intel (x86)
BTS, LBR, PT…

lots of tracing units

STM	(event tracing)
ETM	(execution tracing)
BTS	(execution tracing)

lots of tracing units

supported by (probably good)

proprietary software

widely spread

credit: Samsung, tabletolic.com, player.de, digitaltrends.com

Do you have one of these?

widely spread

credit: Intel

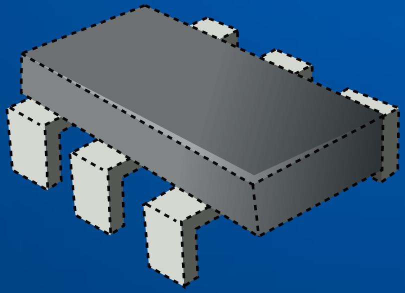
Is your Intel CPU newer than this one?

3

Improvements

3

Improvements

1/3  STM on ARM

System Trace Module (STM)

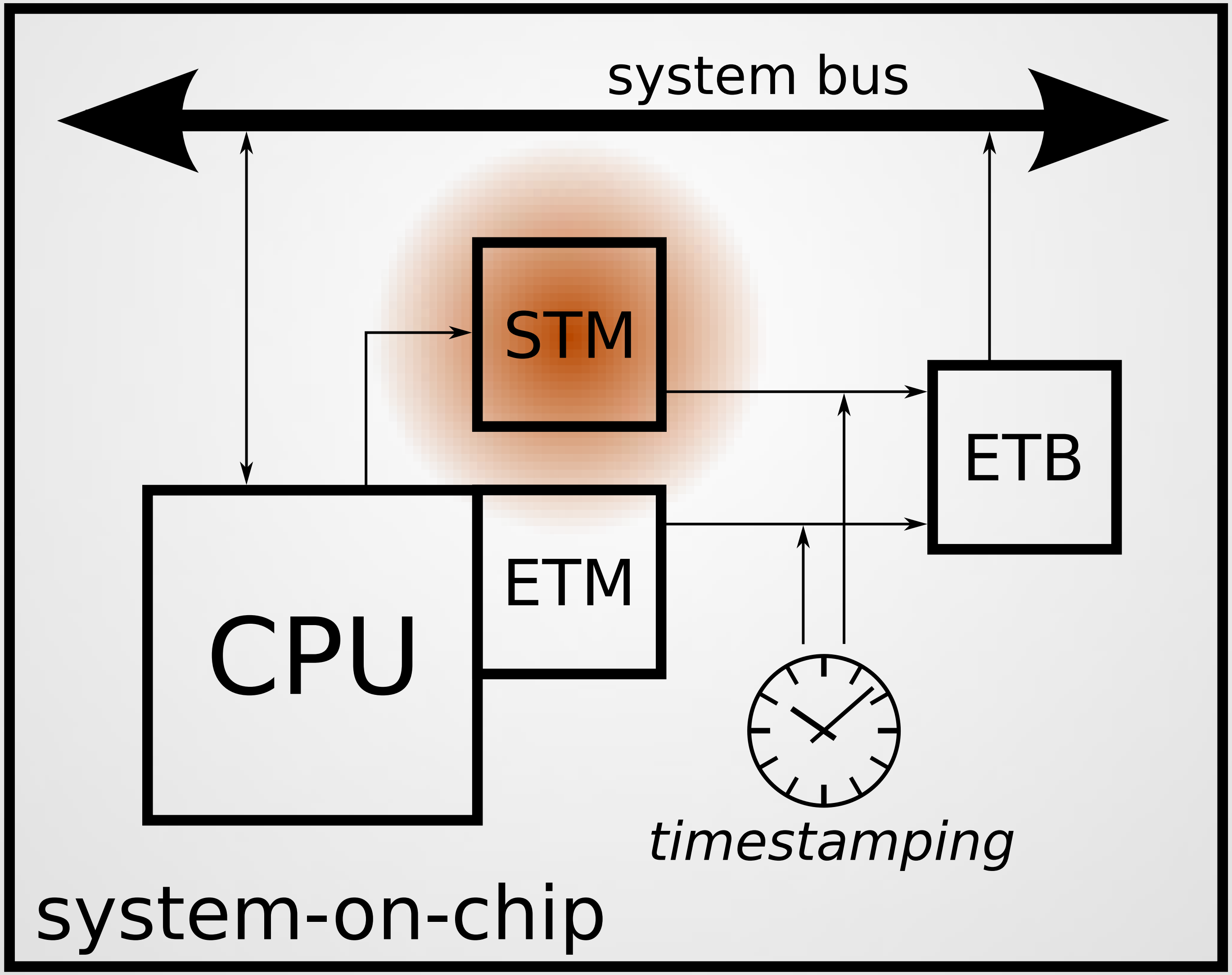
Goal: help software recording events

System Trace Module (STM)

Provides
dedicated resources
bus, buffer, timestamping

Need to
instrument
software

System Trace Module (STM)



implementation

“LTTng-equivalent”

The traced process is instrumented: calling tracepoint() writes to the STM.

Embedding payload is possible.

A consumer process retrieves generated traces and stores them.

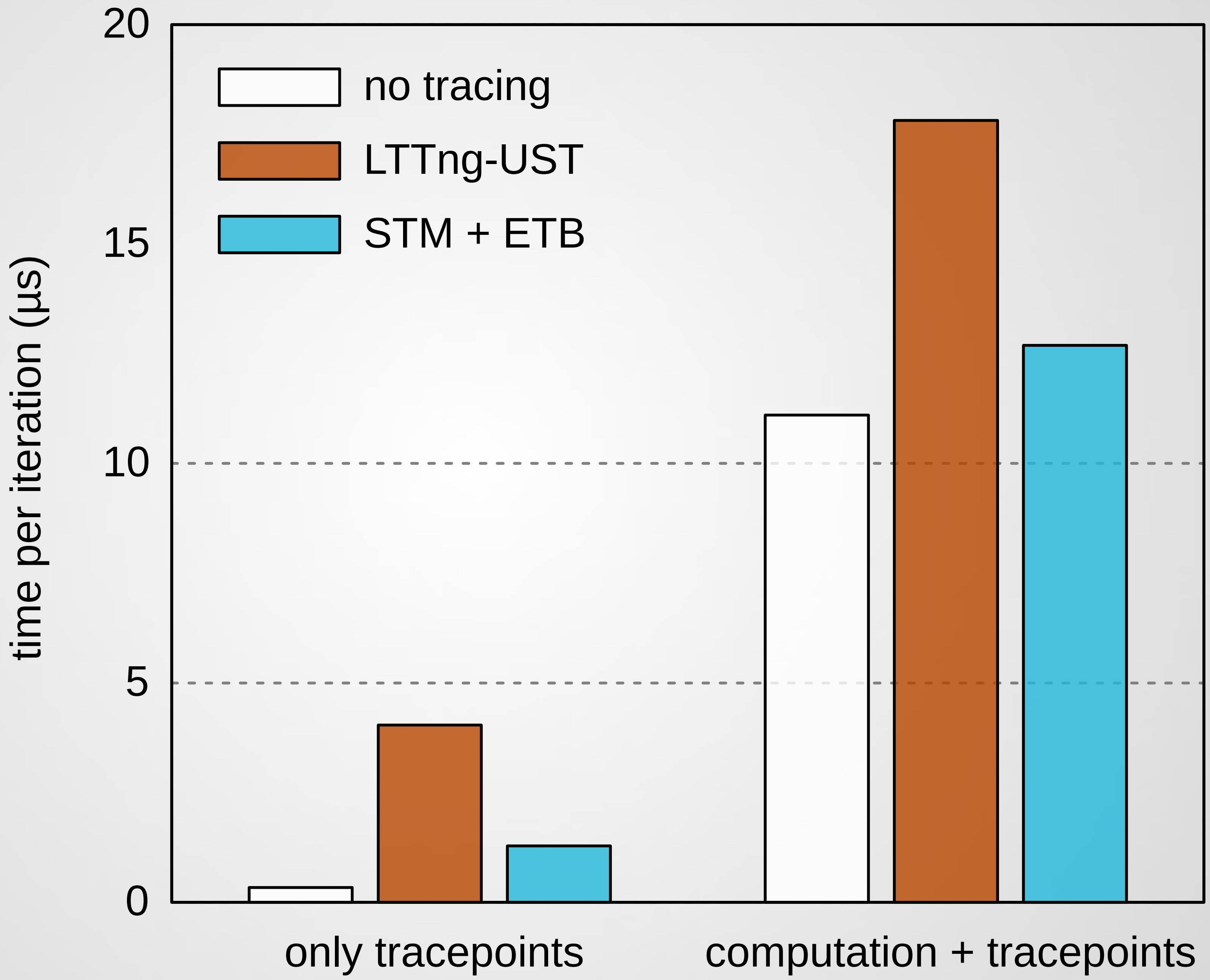
implementation

Traces are encoded in STP.

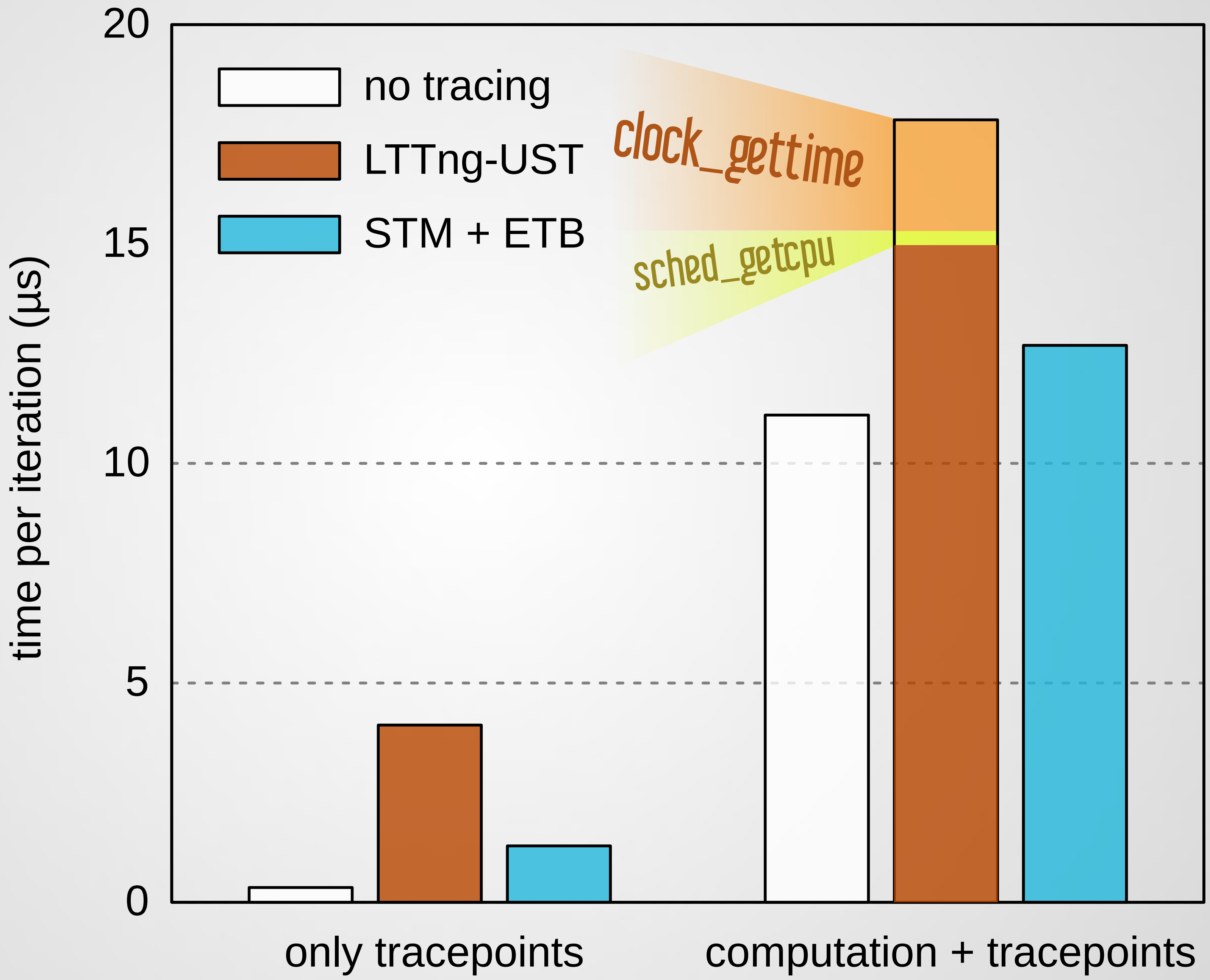
optimized, compact but proprietary format

results

indicative benchmark: overhead mostly depends on the traced application!

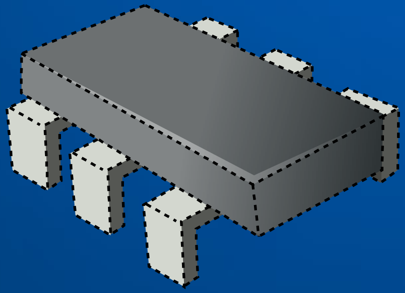


results


time per iteration (µs)
20
15
10
5
0
no tracing
LTTng-UST
STM + ETB
clock_gettime
sched_getcpu
only tracepoints
computation + tracepoints

3

Improvements

2/3  ETM on ARM

Embedded Trace Macrocell (ETM)

Goal: trace execution

Embedded Trace Macrocell (ETM)

Goal: trace execution

i.e. save every executed instruction address

Embedded Trace Macrocell (ETM)

Provides

dedicated resources

address comparators,
buffer, timestamping

Embedded Trace Macrocell (ETM)

Can focus on a
specific process or function
triggers upon custom conditions

Provides
dedicated resources
address comparators,
buffer, timestamping

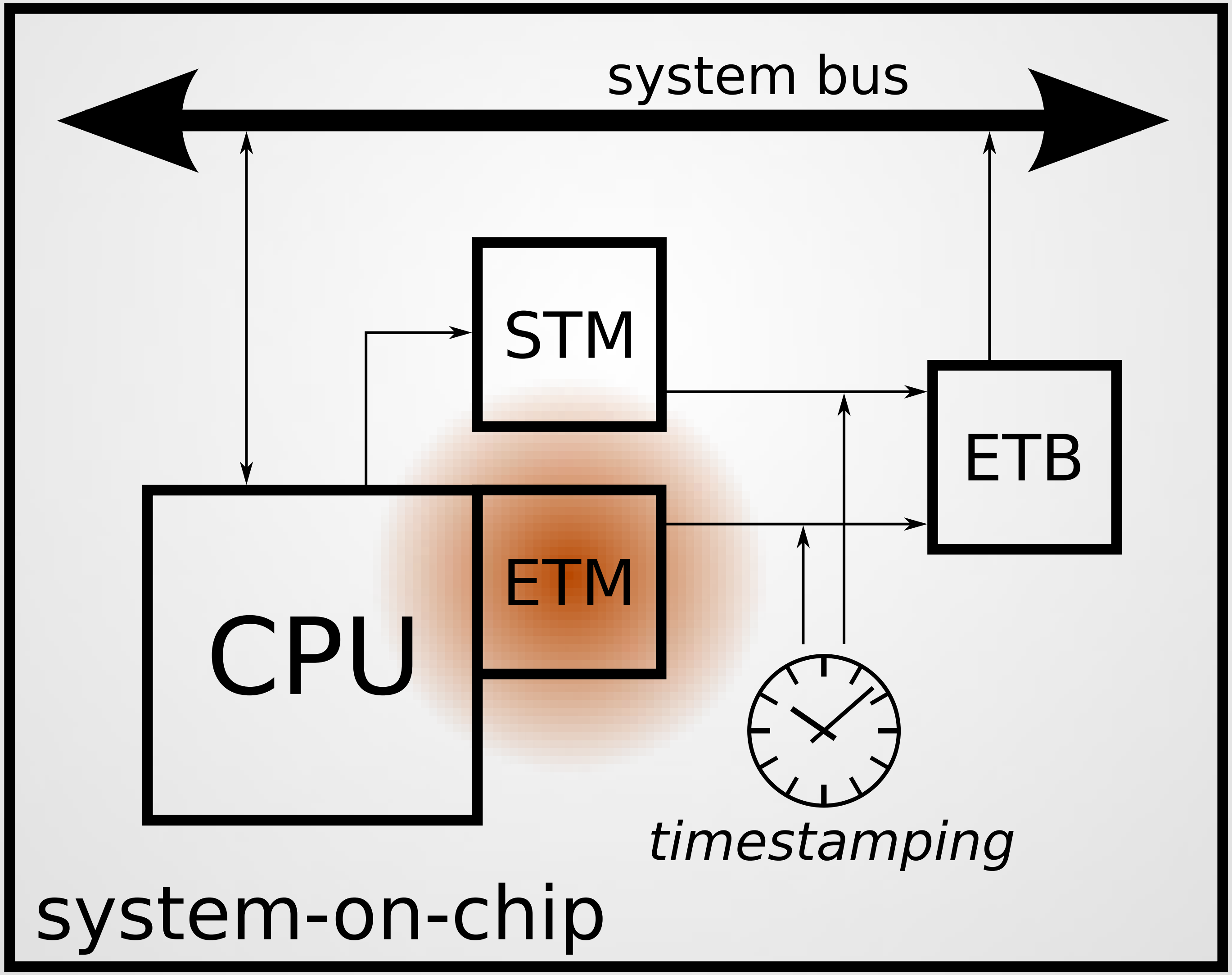
Embedded Trace Macrocell (ETM)

Can focus on a
specific process or function
triggers upon custom conditions

Provides
dedicated resources
address comparators,
buffer, timestamping

No need to
instrument
software

Embedded Trace Macrocell (ETM)



implementation

ETM not meant to trace events

implementation

ETM not meant
to trace events

Idea:

do execution tracing
on event addresses

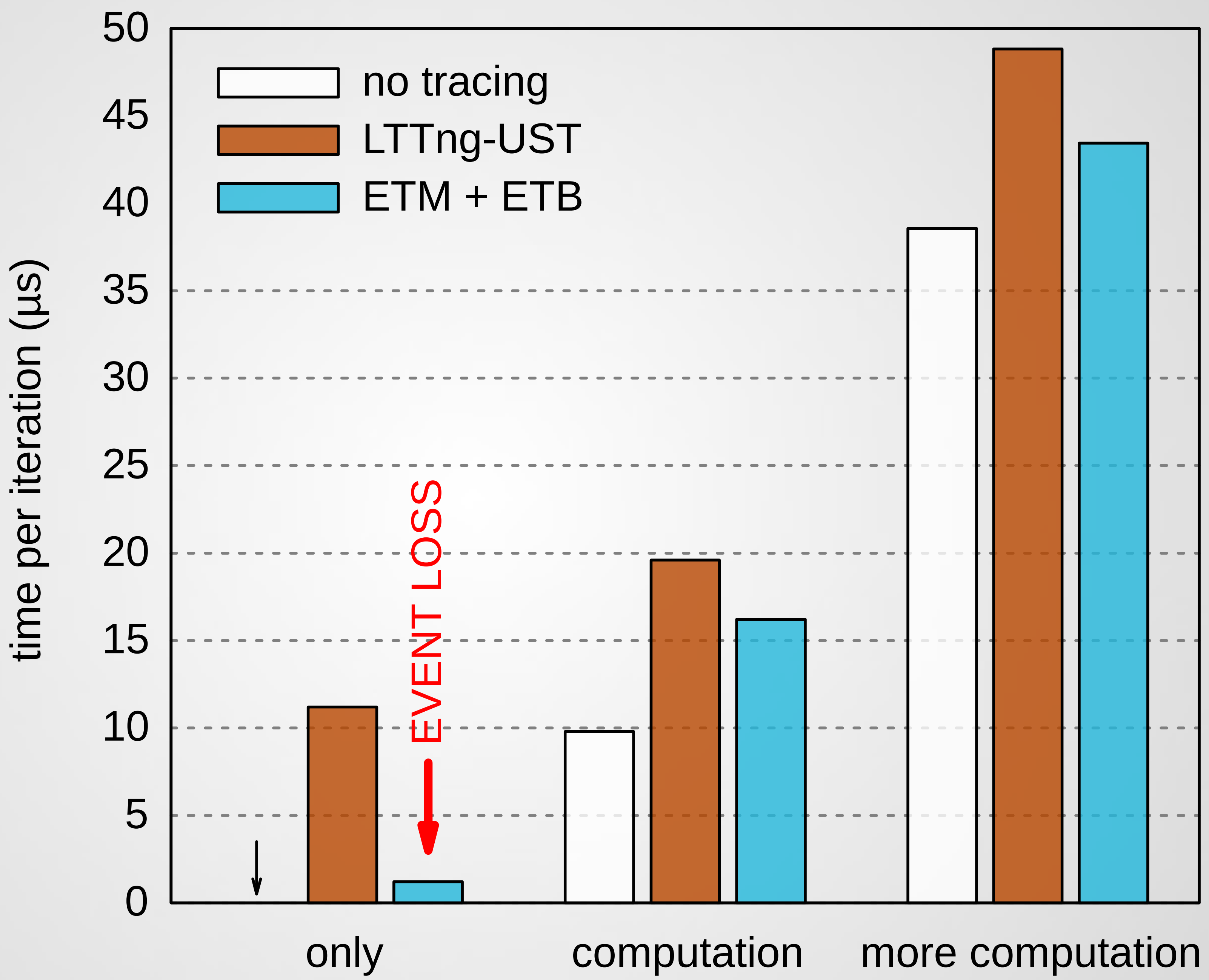
set address comparators
to trigger in [event, event+4]

implementation

needed to write
kernel support for

process and
function tracing

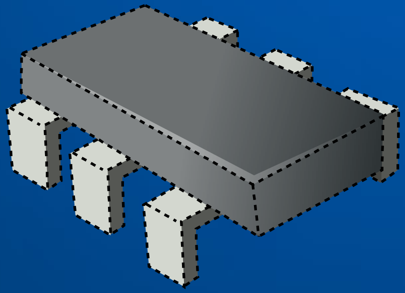
results



no tracing
LTTng-UST
ETM + ETB
time per iteration (µs)
0
5
10
15
20
25
30
35
40
45
50
EVENT LOSS
only
computation
more computation

3

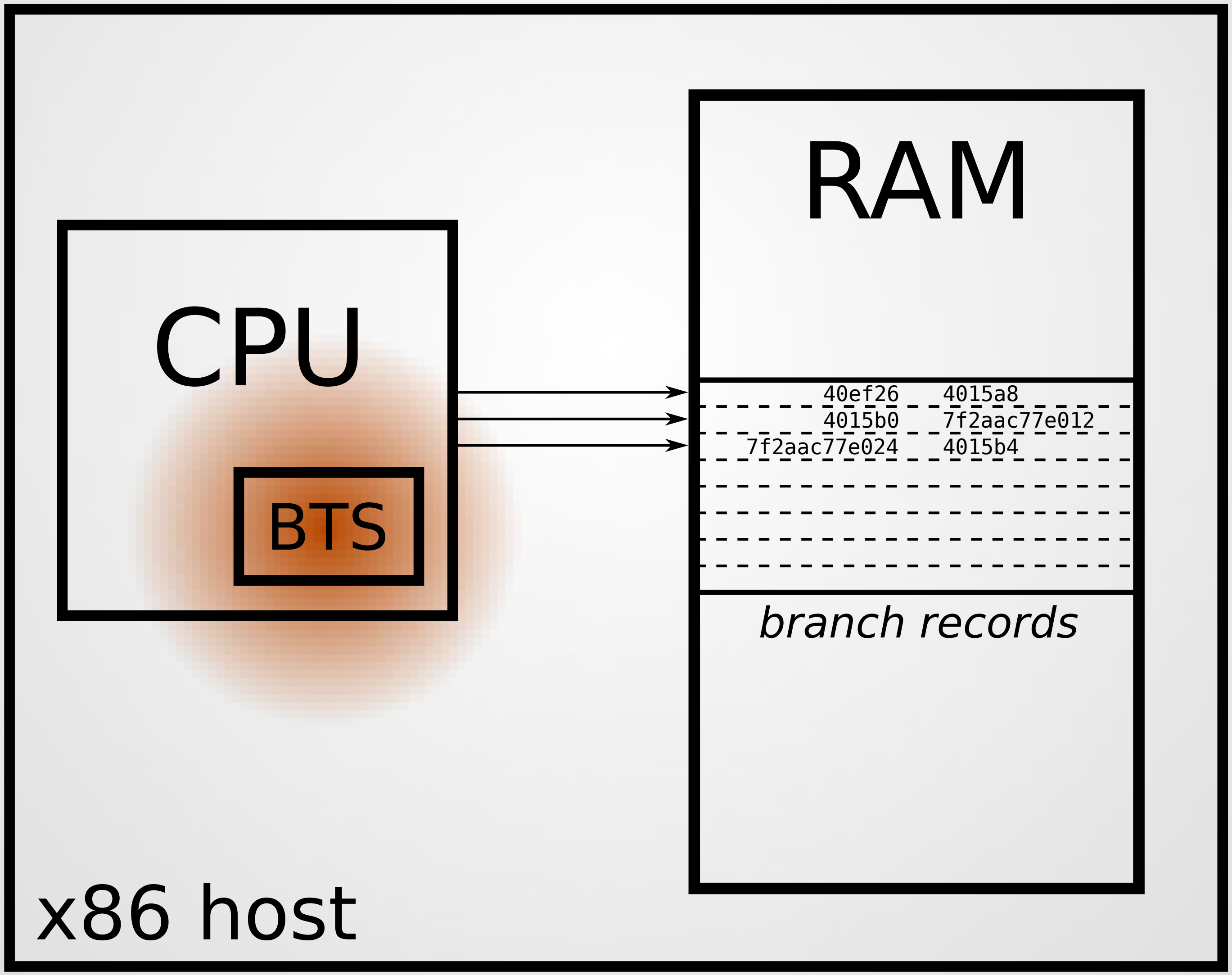
Improvements

3/3  BTS on x86

Branch Trace Store (BTS)

Goal: trace execution

Branch Trace Store (BTS)



Branch Trace Store (BTS)

does not provide dedicated buffers

cannot focus on a specific process or function: traces every branch!

```
$ perf record -e branches:u -c 1 -d ./myprogram
$ perf script -f time,ip,addr
101918.272364:  ffffffff814a6f2c =>     7f8d7b9b3180
101918.272364:  ffffffff814a6f2c =>     7f8d7b9b3180
101918.272364:      7f8d7b9b3183 =>     7f8d7b9b6730
101918.272364:  ffffffff814a6f2c =>     7f8d7b9b6730
101918.272364:  ffffffff814a6f2c =>     7f8d7b9b674f
101918.272364:  ffffffff814a6f2c =>     7f8d7b9b6756
101918.272364:      7f8d7b9b67c2 =>     7f8d7b9b67df
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67ef =>     7f8d7b9b6a30
101918.272364:      7f8d7b9b6a38 =>     7f8d7b9b6a58
101918.272364:      7f8d7b9b6a62 =>     7f8d7b9b6bc0
101918.272364:      7f8d7b9b6bd7 =>     7f8d7b9b67d3
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
101918.272364:      7f8d7b9b67e3 =>     7f8d7b9b67c8
```

BTS not meant to trace events

if enabled, traces every branch

"Is hardware-assisted branch tracing faster than pure-software event tracing?"

implementation

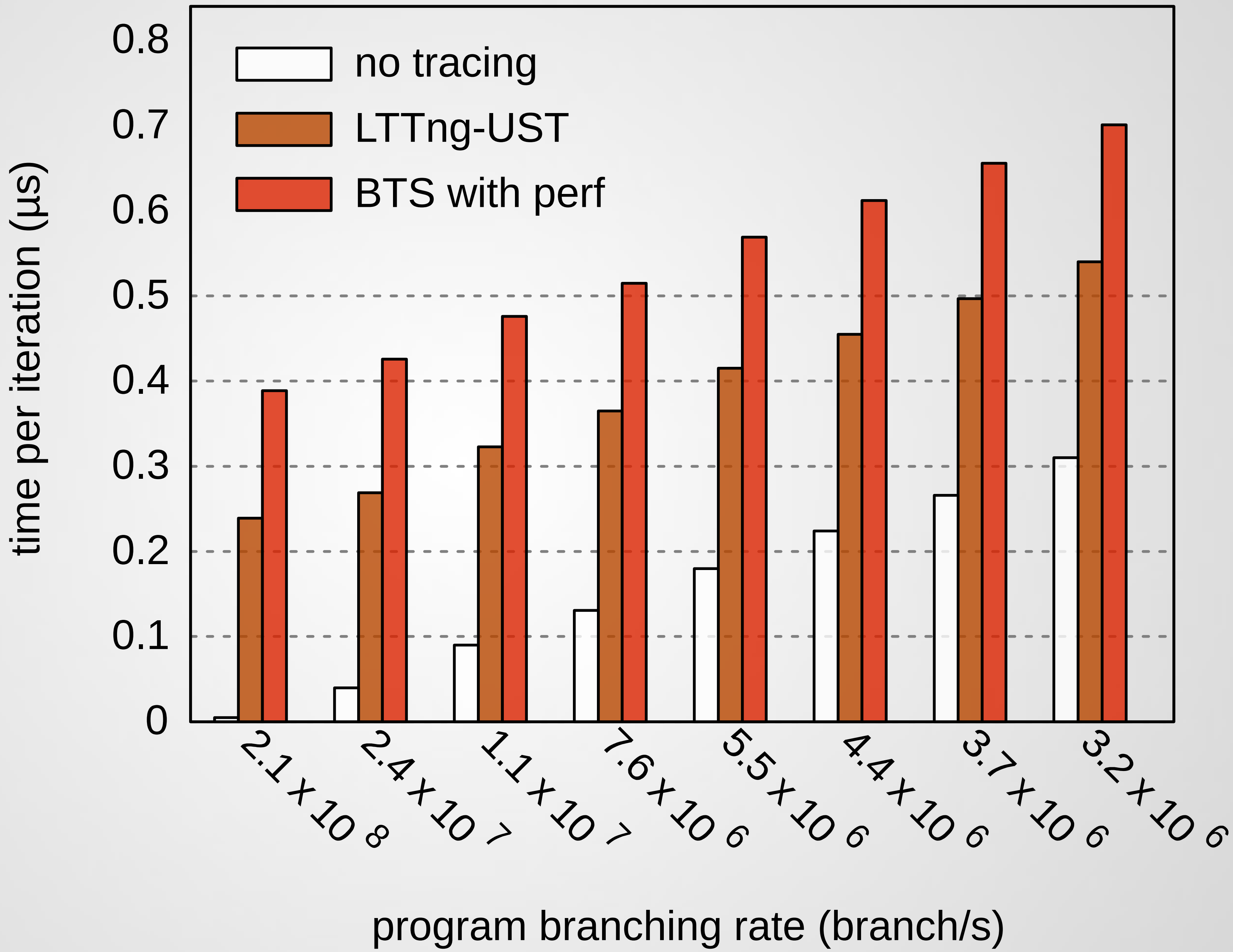
hardware-traced with BTS:

simple program, every branch recorded

software-traced with LTTng:

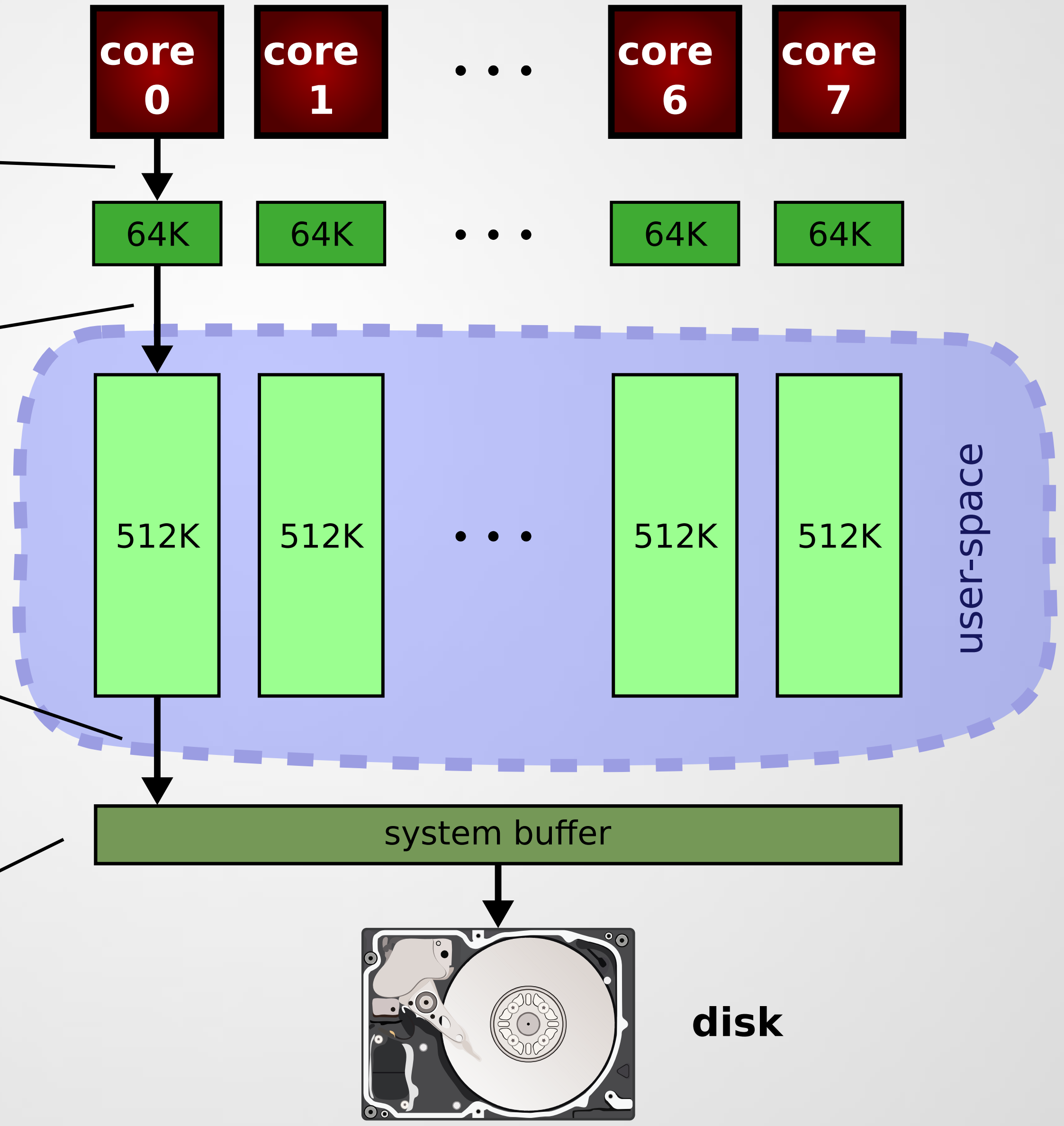
same program, add a tracepoint() at every branch

results



time per iteration (µs)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
no tracing
LTTng-UST
BTS with perf
2.1 x 10 8
2.4 x 10 7
1.1 x 10 7
7.6 x 10 6
5.5 x 10 6
4.4 x 10 6
3.7 x 10 6
3.2 x 10 6
program branching rate (branch/s)

original perf



BTS writes trace to a dedicated buffer

trace is copied to a bigger memory zone upon buffer full or context switch

user stores trace to disk using the write system call

possible copy in another buffer because no O_SYNC flag

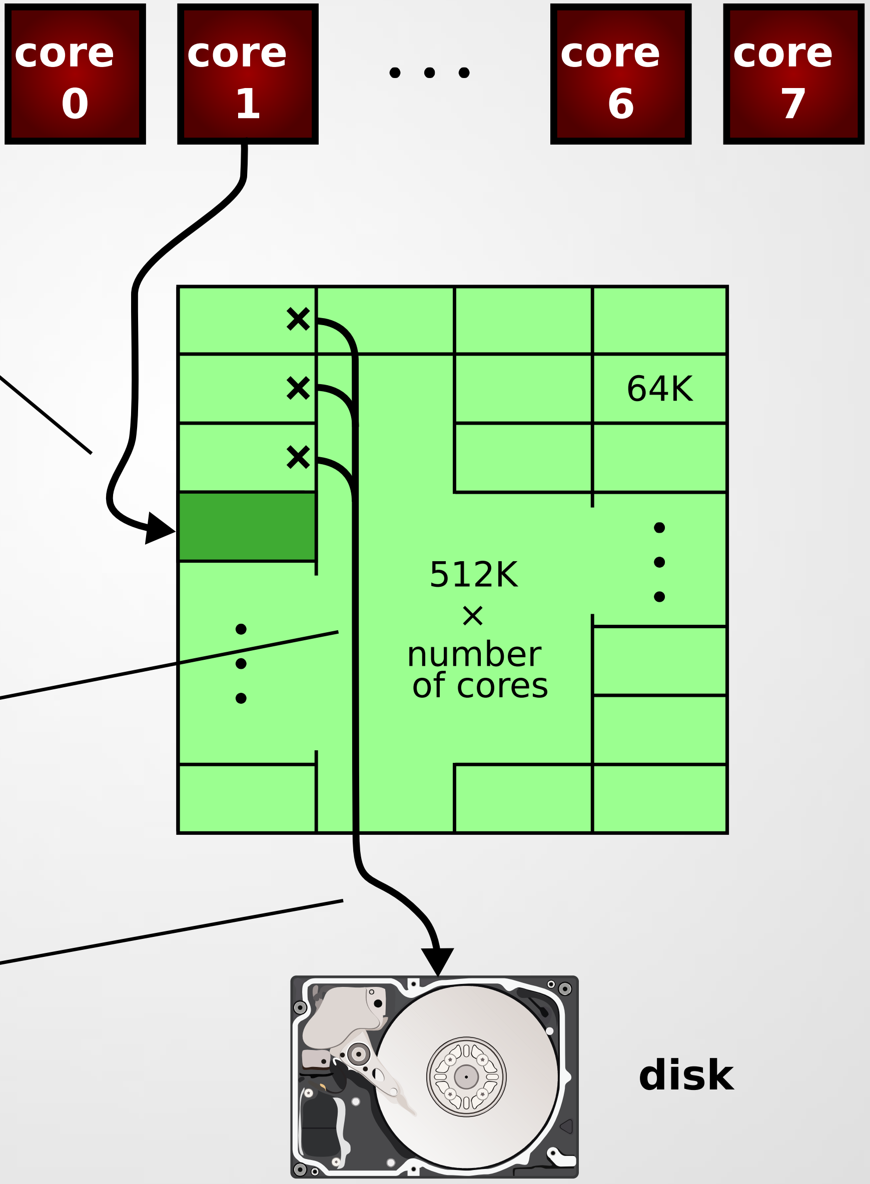
new "spliced" perf

BTS writes trace
to a dedicated buffer

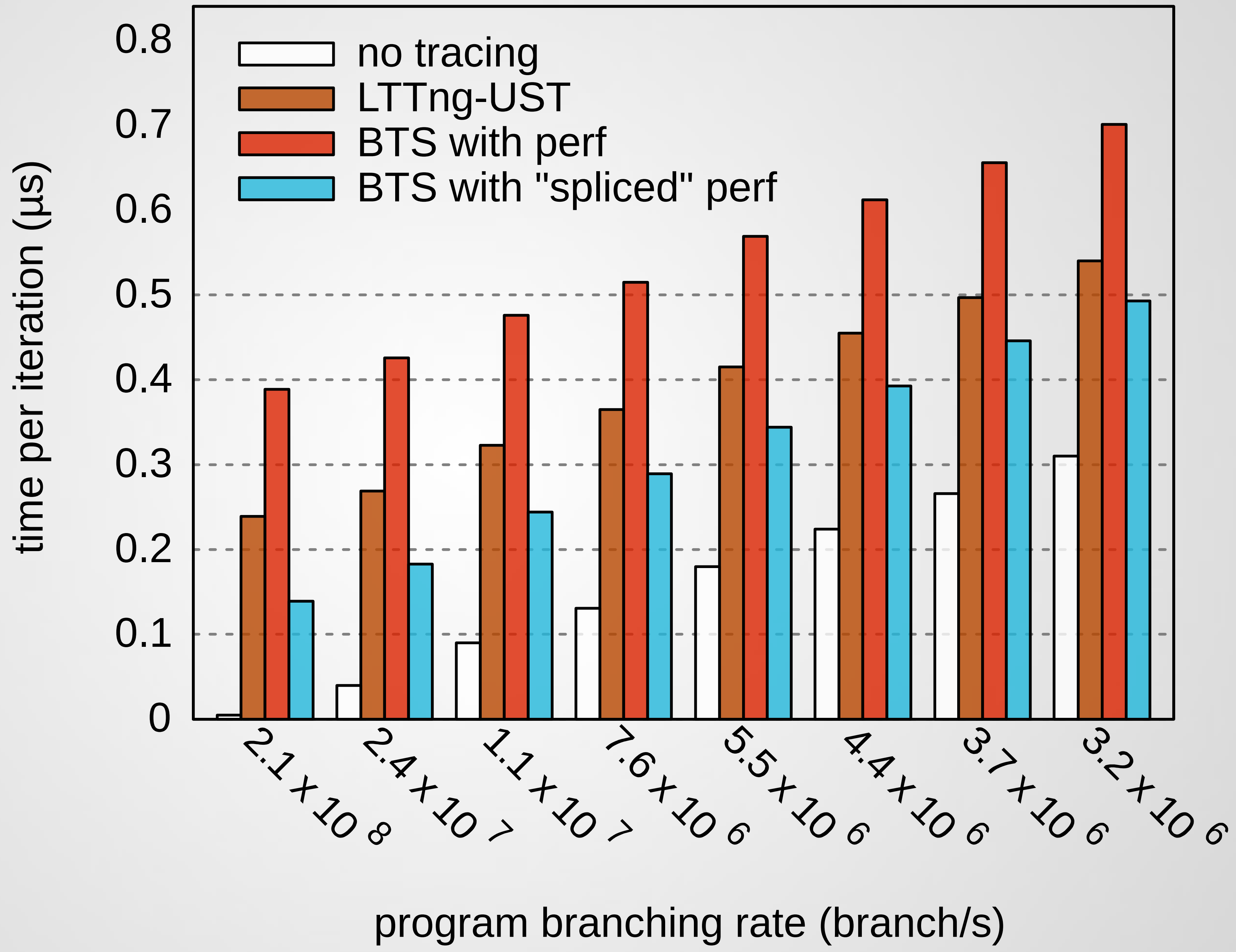
upon buffer full or
context switch, move to
the next sub-buffer

filled sub-buffers
are labeled to be
written to disk later

writing is done
by a kernel task
in user context



results



0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
time per iteration (µs)
no tracing
LTTng-UST
BTS with perf
BTS with "spliced" perf
2.1 x 10 8
2.4 x 10 7
1.1 x 10 7
7.6 x 10 6
5.5 x 10 6
4.4 x 10 6
3.7 x 10 6
3.2 x 10 6
program branching rate (branch/s)

Results

STM	-75 % overhead compared to LTTng-UST needs post-decoding
ETM	-30 % to -50 % overhead limited number of tracepoints no payload
BTS	not suited for event tracing (not flexible) compared to vanilla perf, 2× faster

other hardware

Freescale:	Data Acquisition Program Trace
Intel:	Processor Trace

last words

tracing

helps you build

efficient
software

using LTTng: very low footprint

Cortex-A9: ~ 5 µs / event

Core i7: ~ 200 ns / event

using hardware: almost zero footprint

trace in production!

Links

LTTng and TMF:

https://lttng.org/

STM libraries:

https://github.com/adrienverge/libcoresightomap4430

ETM patch:

https://lkml.org/lkml/2014/1/30/259

BTS patch:

https://github.com/adrienverge/linux/tree/patch_perf_bts_splice

Thank you

Questions?