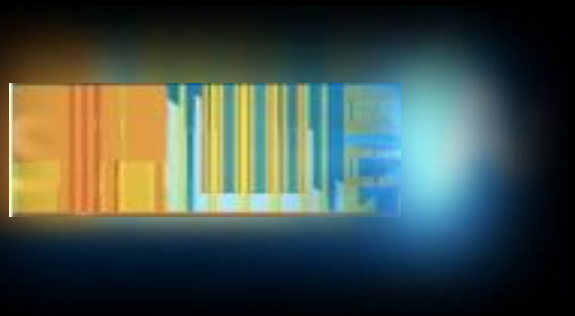


Performance and Power Analysis Utilizing Intel® Performance Bottleneck Analyzer (PBA)

By: Michael Chynoweth
Principal Engineer Intel Corporation

Contributors: Rajshree Chabukswar, Charlie Hewett, Seung-Woo Kim, Vardhan Dugar, Erik Niemeyer, Joe Olivas and Manuj Sabharwal

Legal Disclaimer

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, reference www.intel.com/software/products.

http://intel.com/software/products

Optimization Notice

Optimization Notice

Intel® compilers, associated libraries and associated development tools may include or utilize options that optimize for instruction sets that are available in both Intel® and non-Intel microprocessors (for example SIMD instruction sets), but do not optimize equally for non-Intel microprocessors. In addition, certain compiler options for Intel compilers, including some that are not specific to Intel micro-architecture, are reserved for Intel microprocessors. For a detailed description of Intel compiler options, including the instruction sets and specific microprocessors they implicate, please refer to the "Intel® Compiler User and Reference Guides" under "Compiler Options." Many library routines that are part of Intel® compiler products are more highly optimized for Intel microprocessors than for other microprocessors. While the compilers and libraries in Intel® compiler products offer optimizations for both Intel and Intel-compatible microprocessors, depending on the options you select, your code and other factors, you likely will get extra performance on Intel microprocessors.

Intel® compilers, associated libraries and associated development tools may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include Intel® Streaming SIMD Extensions 2 (Intel® SSE2), Intel® Streaming SIMD Extensions 3 (Intel® SSE3), and Supplemental Streaming SIMD Extensions 3 (Intel® SSSE3) instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors.

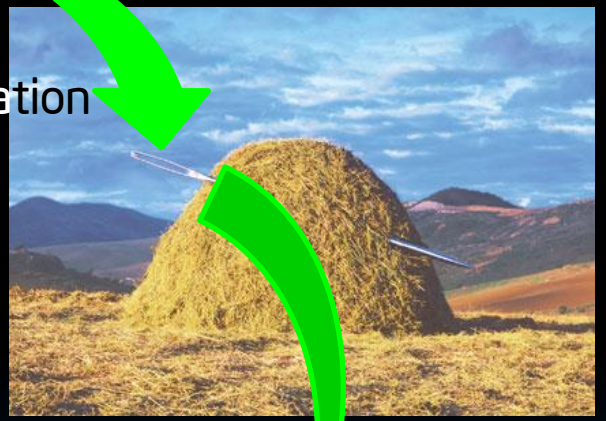
While Intel believes our compilers and libraries are excellent choices to assist in obtaining the best performance on Intel® and non-Intel microprocessors, Intel recommends that you evaluate other compilers and libraries to determine which best meet your requirements. We hope to win your business by striving to offer the best performance of any compiler or library; please let us know if you find we do not.

Notice revision #20101101

Taking Advantage of Engineering Innovation Through Tools

Issue Identified by Hand

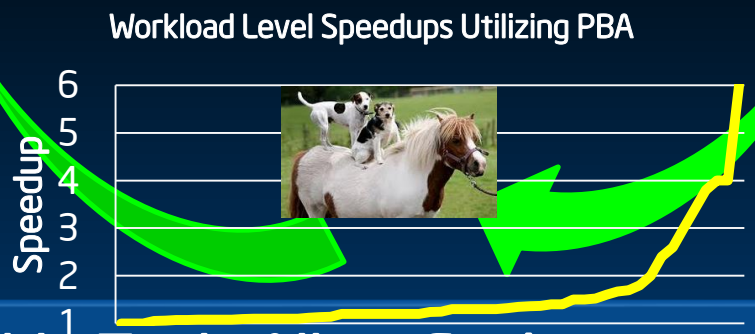
Automatic Identification



Innovation to Drive Automatic Identification

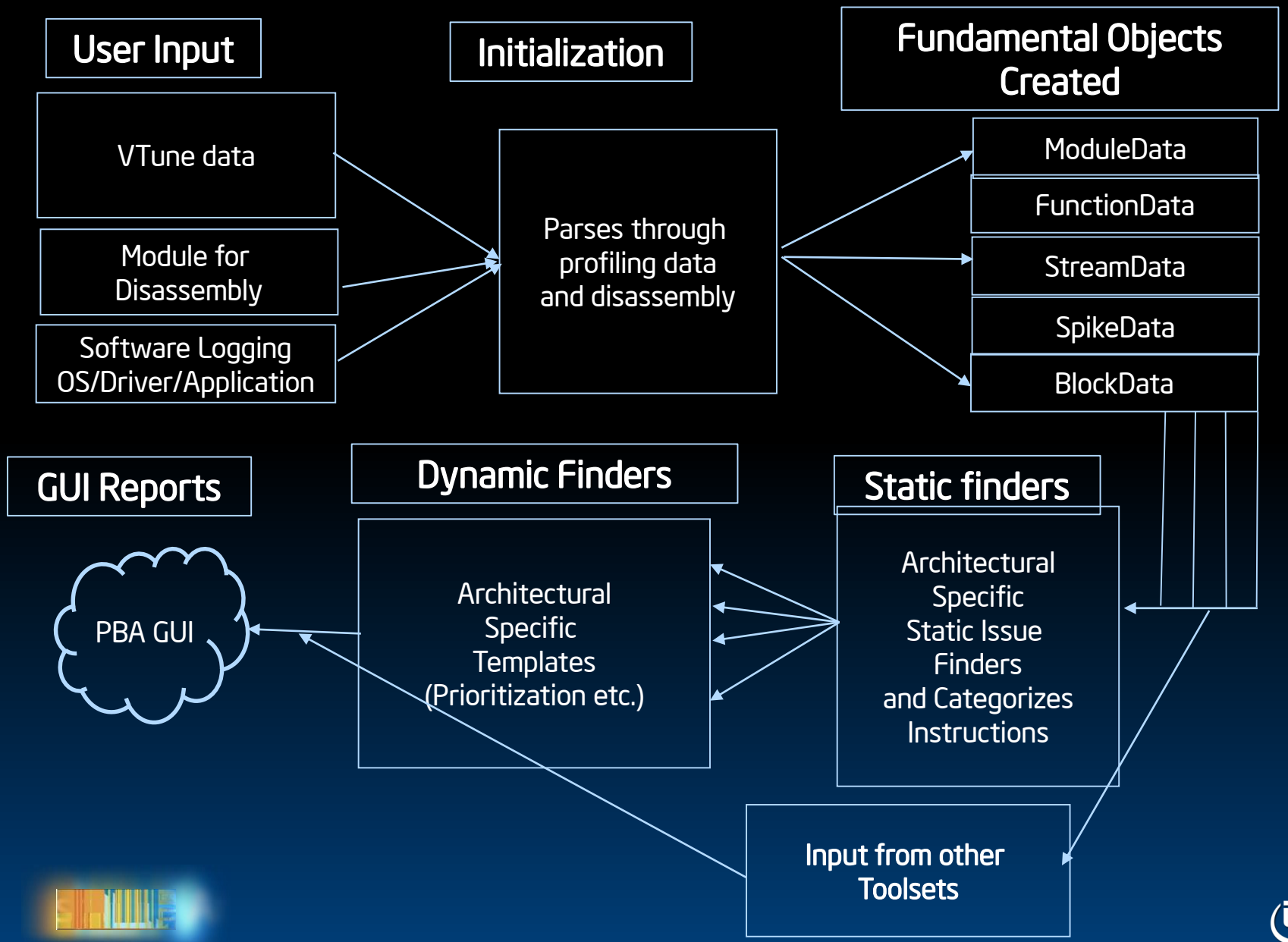
Capability is improved by other engineers

Automated capability used by other engineers

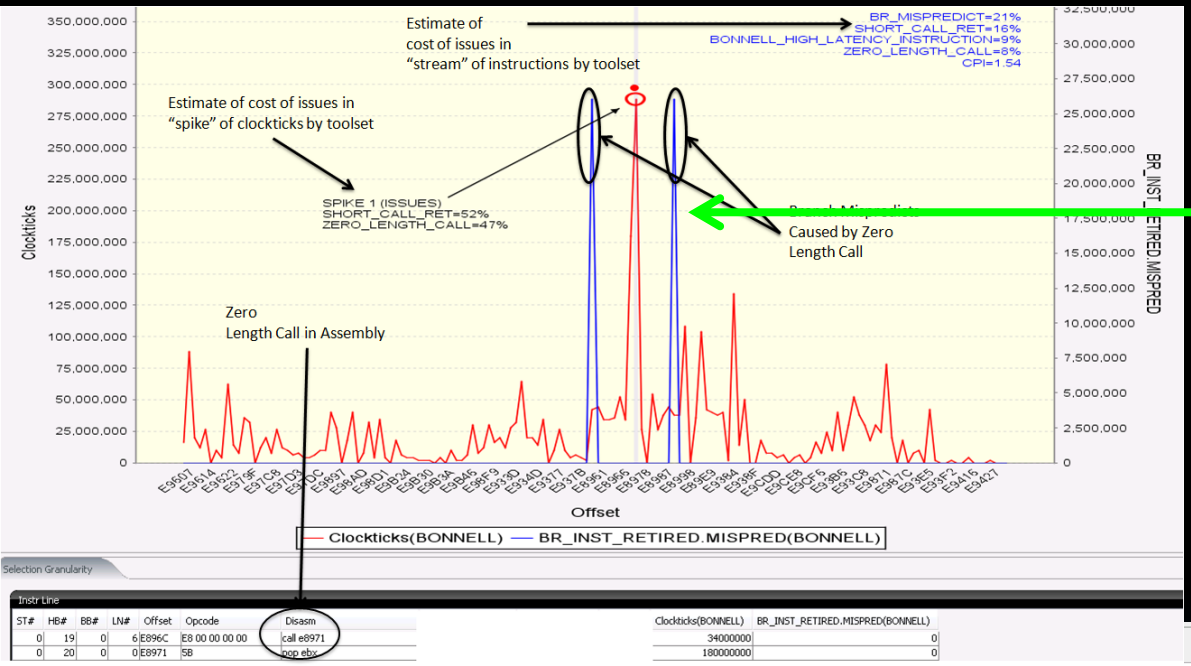


Extensible Tools Allow Engineers to Attack Difficult Problems as a Community

Intel® PBA Flow of Analysis

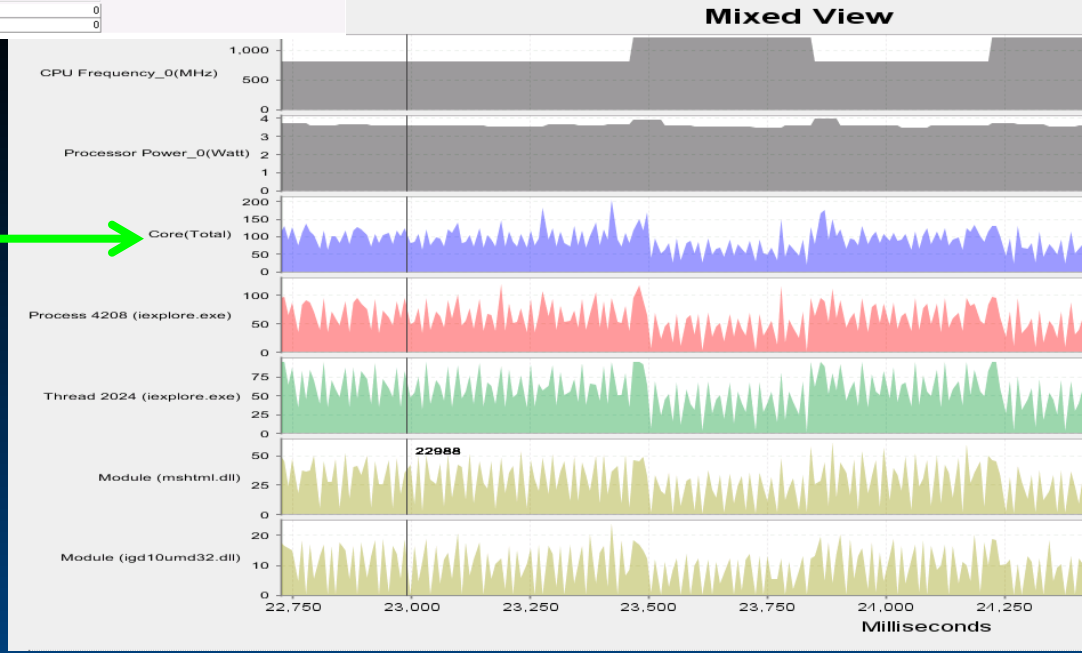


2 Primary Views of PBA: Replacing This



#1 Streams: Recreate paths of execution

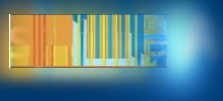
#2 Overtime: CPU, Process, Module, Function Frequency, Power, SW Event, etc.



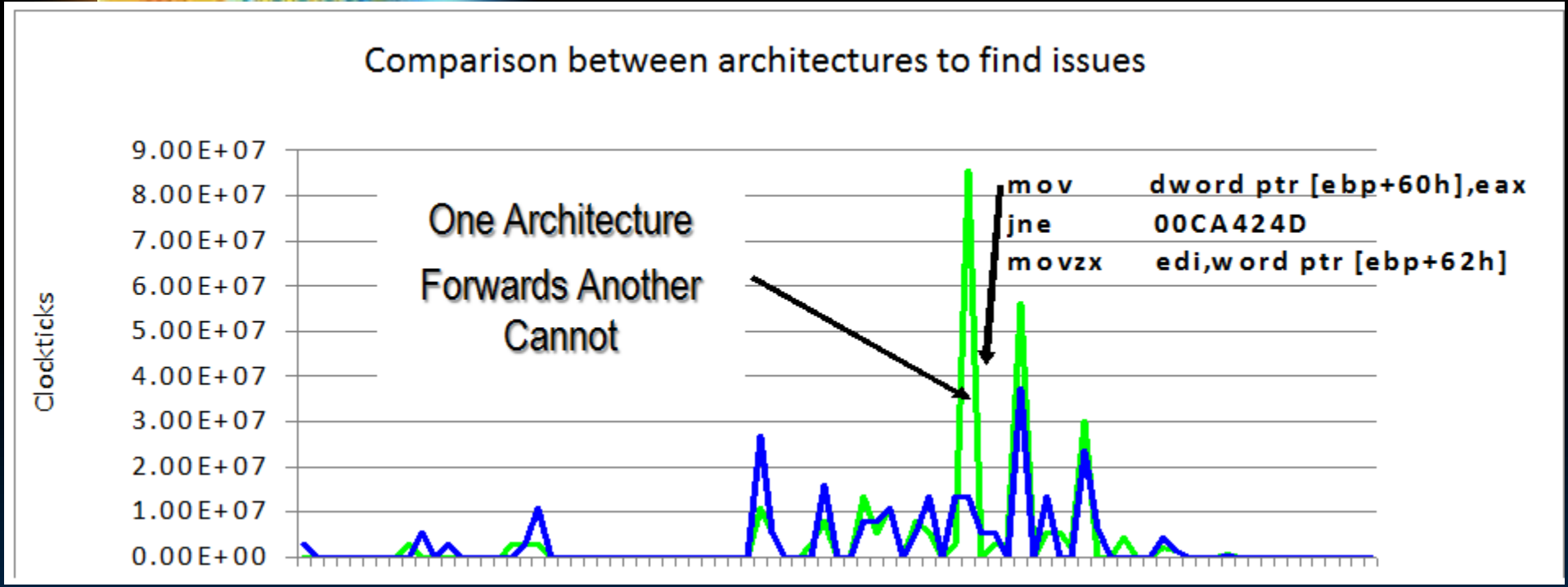
What is A Stream?

- Reconstructing the flow of basic blocks
- Shows flow of instructions as it ran on the core
- We pull all events along the stream for a "Poor-mans" pipe trace
- Find and reconstruct loops as a granularity
- Catches issues across branches

Address	Instruction
30AA668A	call 30AA63F7
30AA63F7	mov ecx, dword ptr [edi]
30AA63FF	sar ebx, 18h
30AA6403	mov esi, eax
30AA6405	js 30AA64D8
30AA640B	movzx eax, word ptr [edi+4]
30AA640F	cmp ax, 0FFFEh
30AA6413	je 30AA647B
30AA647B	movzx eax, word ptr [edi+4]



Comparing Architectures with Streams

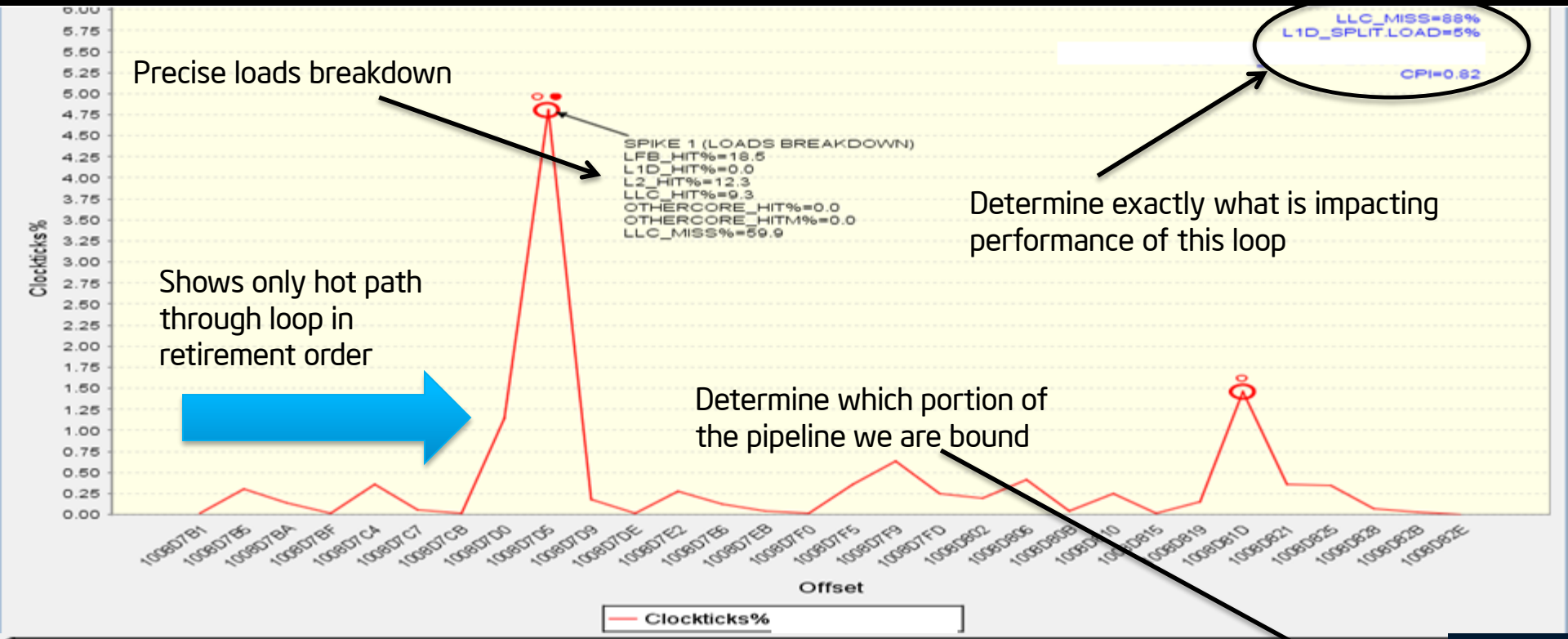


Streams Allow Us To Find Issues Across Branch Boundaries

DEMO #1 Streams:

"If you want micro-architectural issues fixed…make it easy"

Streams: Summary of Determining Issues in a Loop

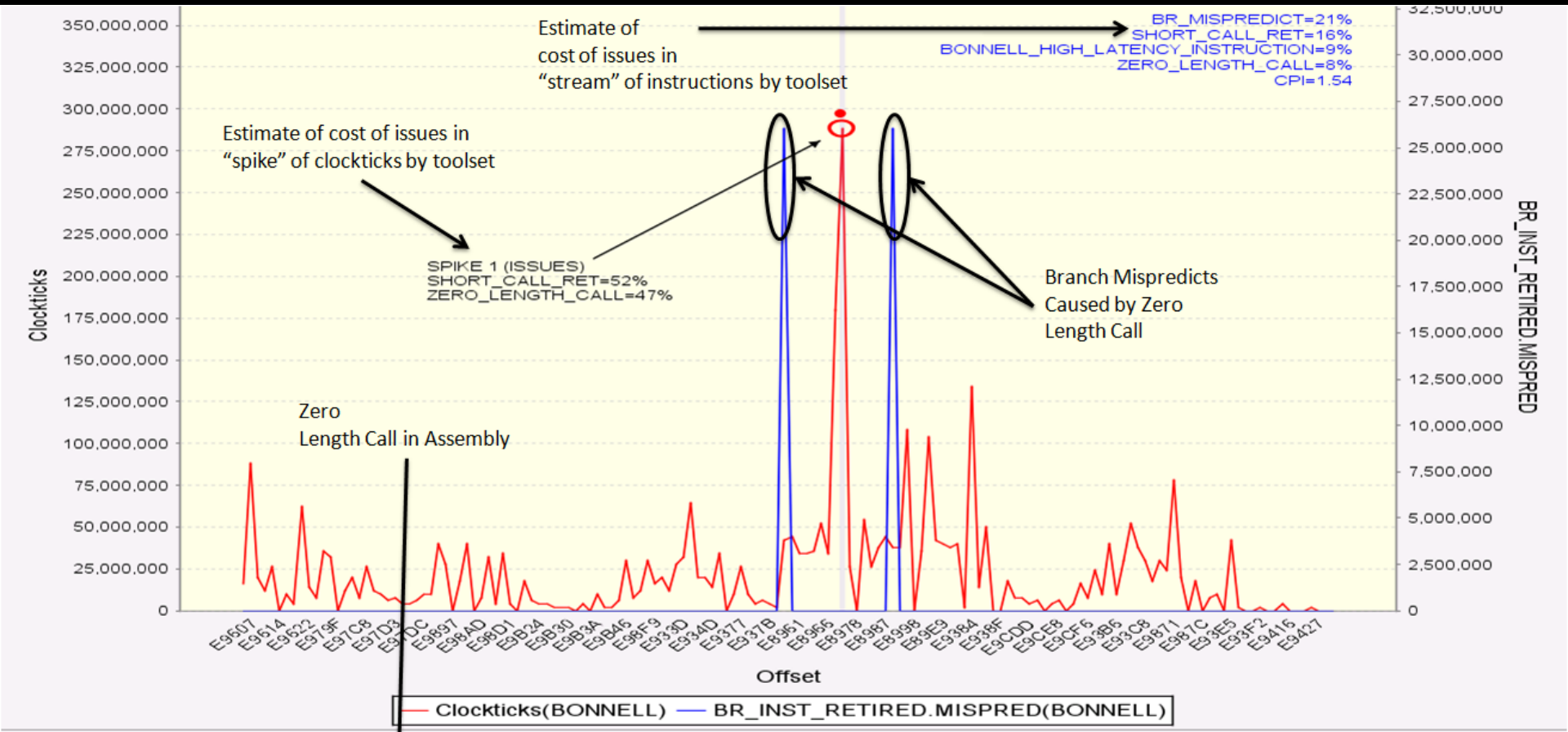


LoopStream

ST#	Offset	HitCount	Avg Trip Count	Clockticks...	FrontEnd%...	BadSpeculation%...	Retiring%...	BackEndBound%...
6	1008D7B1	167600000	12.0	12.11	0.95	0.00	31.74	67.31

Get accurate loop trip count
With no instrumentation using LBR

Streams: Correlating Static and Dynamic Identification



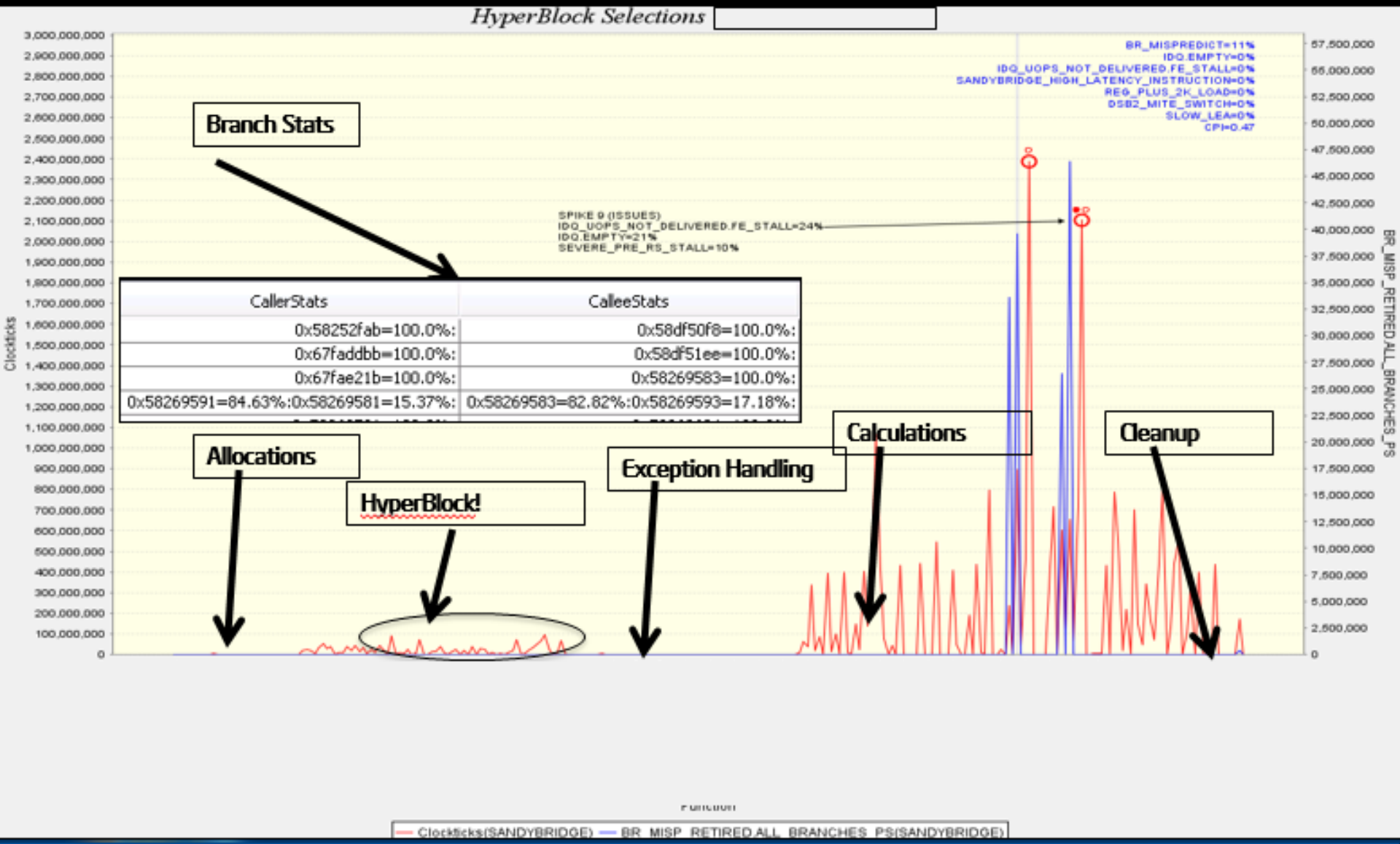
Selection Granularity

Instr Line

ST#	HB#	BB#	LN#	Offset	Opcode	Disasm	Clockticks(BONNELL)	BR_INST_RETIRED.MISPRED(BONNELL)	Static ASM Notes
0	19	0	6	E896C	E8 00 00 00 00	call e8971	34000000	0	AGEN_STALL:ZERO_LENGTH_CALL
0	20	0	0	E8971	5B	pop ebx	180000000	0	SHORT_CALL_RET

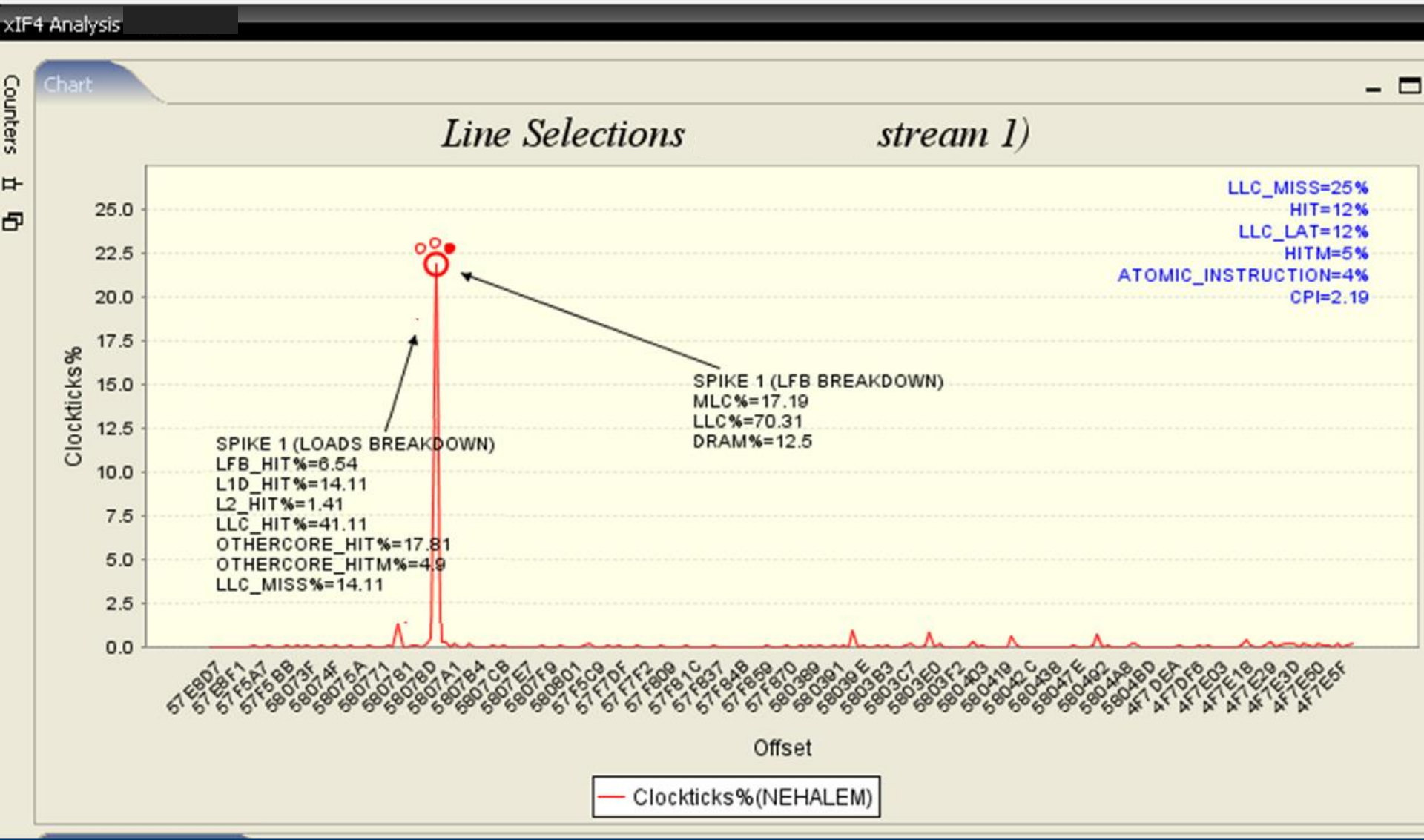
PBA Relates Static ASM with Events

Streams: Assist in Creating Larger and Larger Streams



Streams Created with Last Branch Records Produce Greater Context

Streams: Loads and LFB Breakdown



PBA Displays Where Problematic Loads are Satisfied

Theme #2 JIT Explosion: "Everyone is writing a JIT Nowadays"

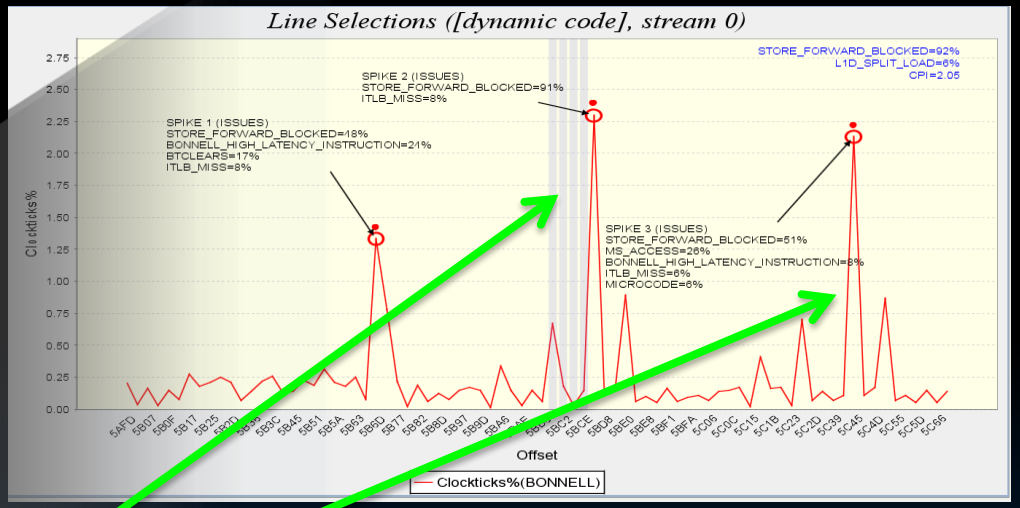
Vtune JIT APIs picked up by xIF

JIT Explosion

Javascript JITs

Win8 Apps, Dalvik

Automated Code Gen Analysis for JITs



xIF automatically identifying issues in JIT

Theme #2 Short Non-Steady State Workloads Becoming Critical

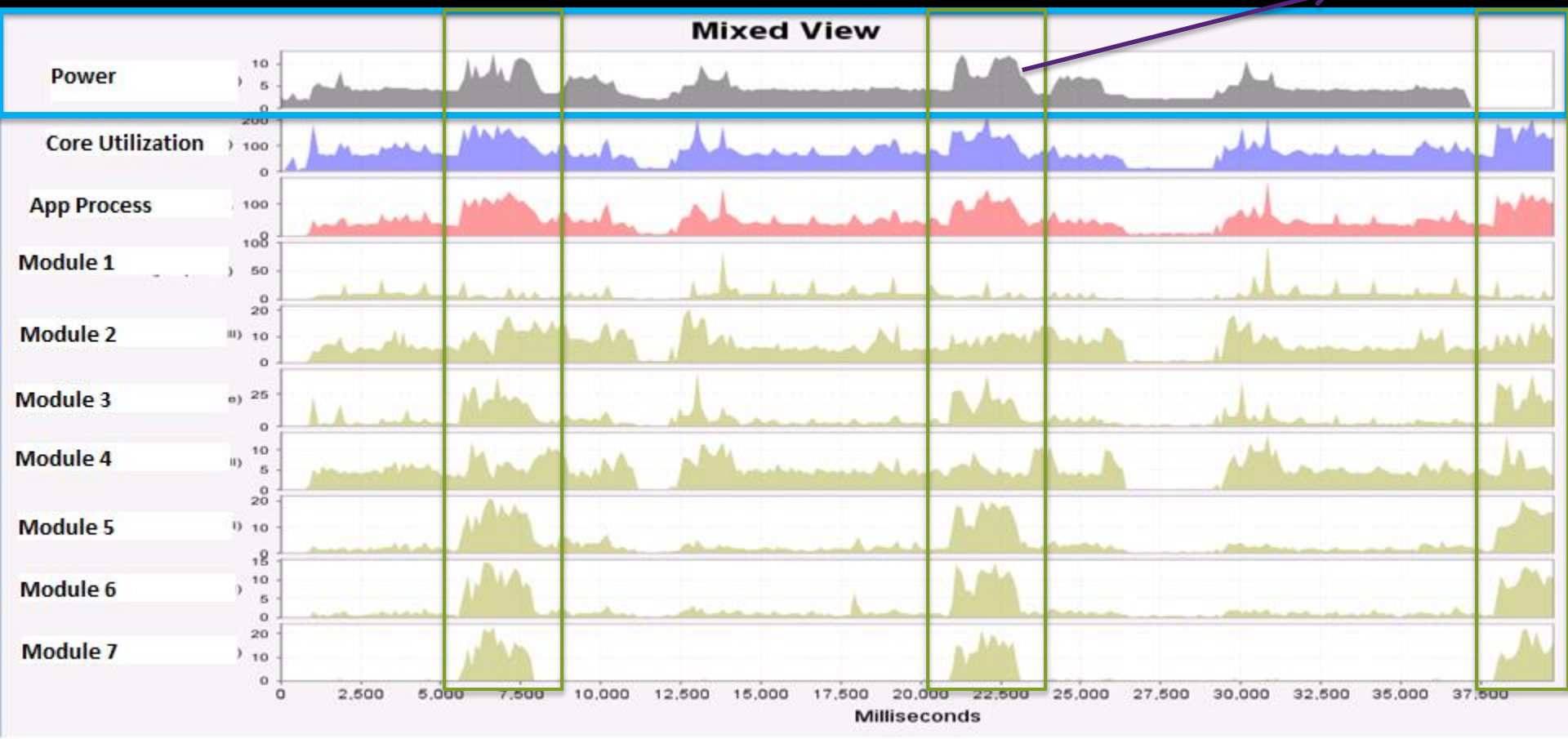
“Debugging in the Millisecond Timeframe”

DEMO TOUCH DEBUG

Theme #2 Power:

“Power and performance are two sides of the same coin”

Power Correlation using Intel® PBA



- Package power intermittently jumping high

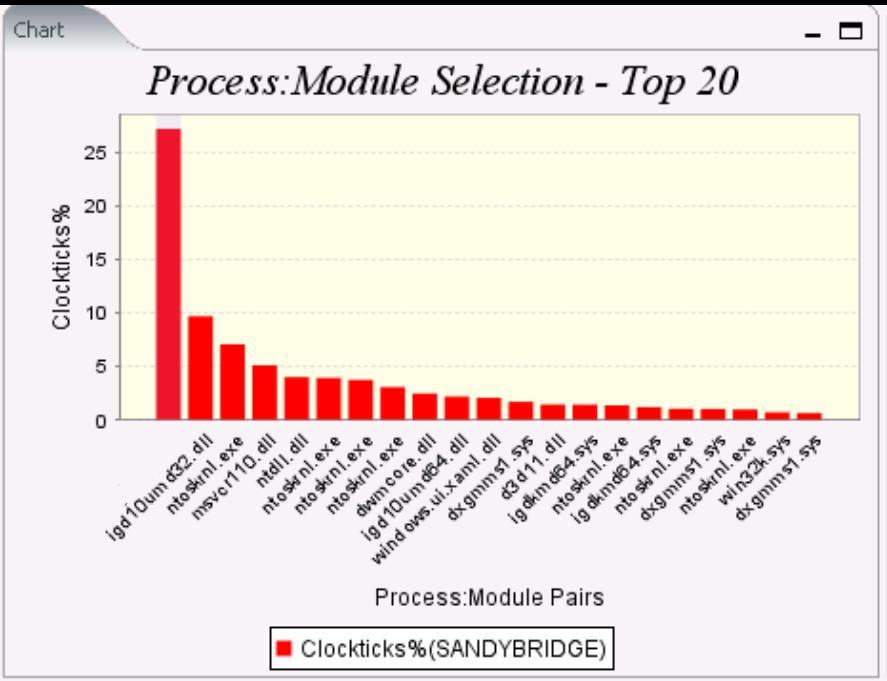
Automatically Determine Causes of High Power

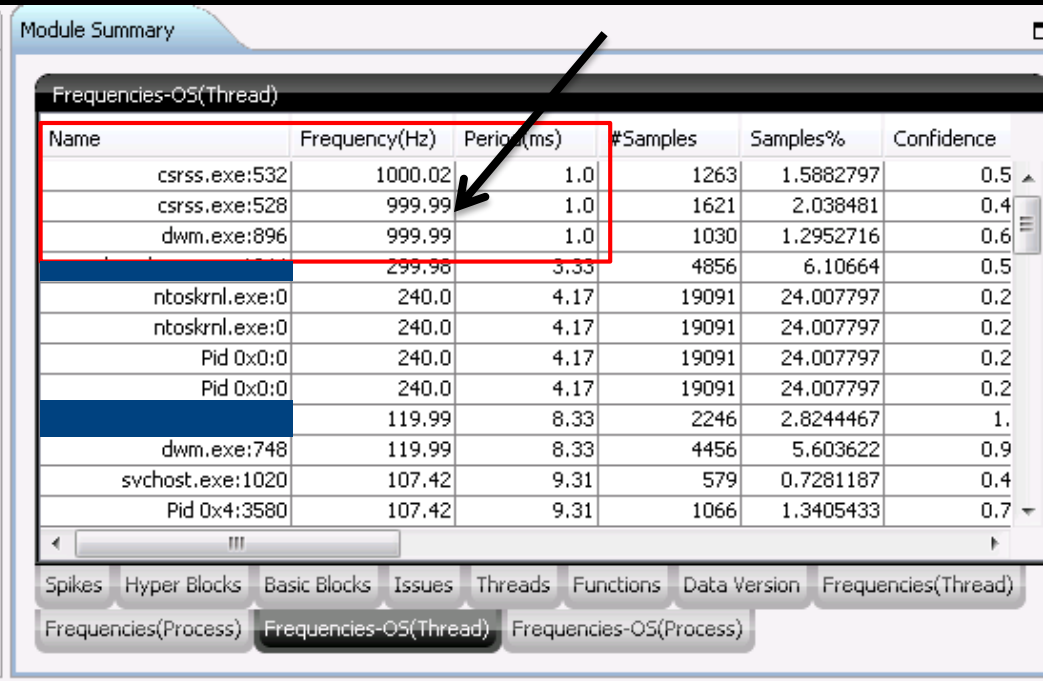
Module_Name	Clocktick High Power%	Clockticks Low Power%
AppProcess:mshtml.dll	15.3	9.39
AppProcess:ntoskrnl.exe	10.64	5.12
AppProcess:ntdll.dll	7.92	2.5
AppProcess:oleaut32.dll	5.95	0.15
AppProcess:igd10umd64.dll	5.57	7.23
AppProcess:msvcrt.dll	4.89	1.3

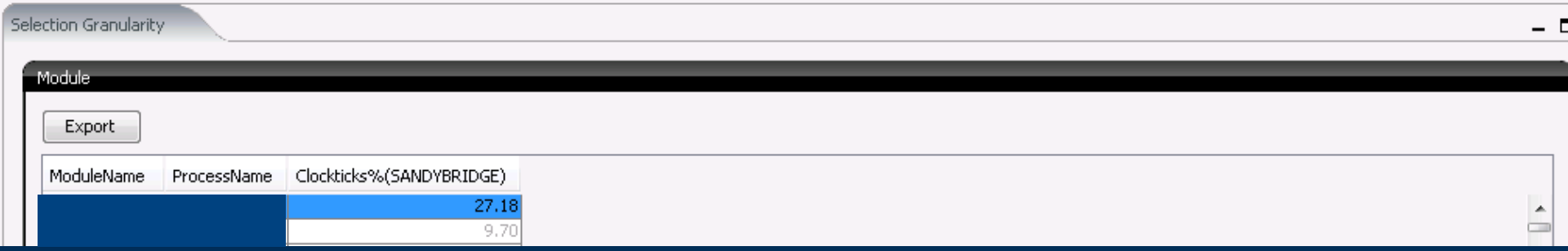
Loading Library is Causing High Power

Shifting to Power Analysis

1ms activities
wasting power!







Calculating Frequency of Activities is Powerful for Performance

CERN Collider Proposal

Conclusion

- Intel® PBA post-processes data collected using Intel® VTune to automatically call out micro-architectural and SoC issues
- Issue identification capabilities increased by order of magnitude using streams
- PBA provides extensibility by specifying rules for various architectures in a configuration file
- Overtime views, UX and power analysis are new capabilities that augment existing functionalities