Hardware Counters Working Group Outbrief

Scalable Tools Workshop
Granlibakken 2016

HW Counter Top Down Methodology

- Intel Experts: Michael Chynoweth, Ahmad Yasin
- Very important for understanding performance
- **Descriptions exist but not for KNL or Atom**
- Top down spreadsheet
  - https://download.01.org/perfmon/TMAM_Metrics.xlsx
- Collect many counters with a multiplexed run
  - is multiplexing OK?
    - fixed counters not impacted by multiplexing (fixed counter in clocks)
      - collect multiplexed data and compare with fixed counter attribution
        - if difference is greater than 10%, then multiplexed data is suspect
          - take weighted average to compare the data use PEBS to collect cycles (consumers of expensive loads)
      - Ask Stephane about how to validate multiplexing on Linux
- Fixed cost penalties: take them with grain of salt (may be overlapped)
- Understanding memory hierarchy issues on Intel architectures
  - use PEBS to track expensive loads (e.g. L3 misses)
  - use PEBS to collect cycles (consumers of expensive loads)

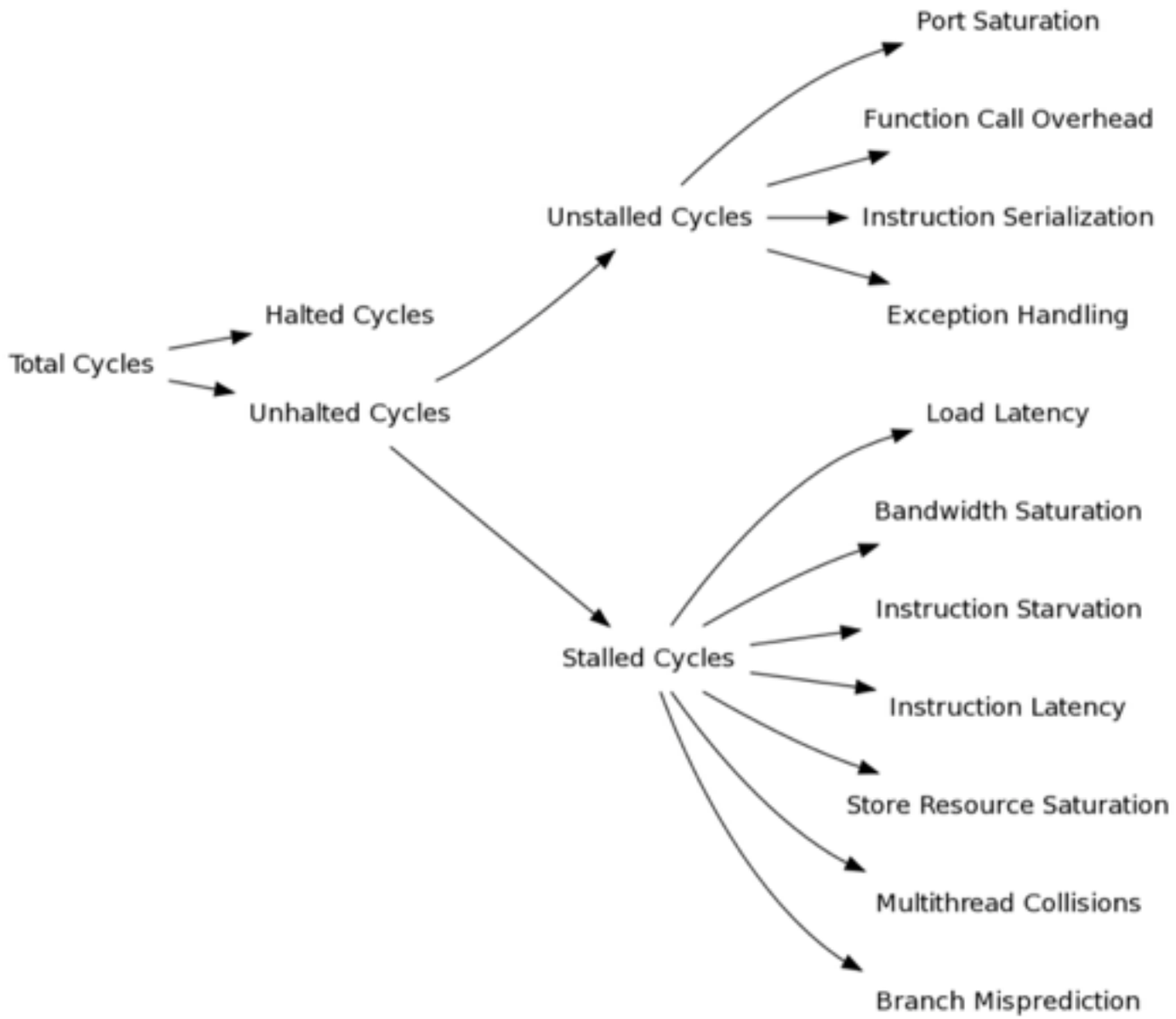
Intel Top-Down Methodology

TMAM Version 3.1 Intel Confidential

Key	Level1	Level2	Level3	Level4	SKL	Server BDX	BDW/BDW-DE	Server HSX	HSW	Server IVT	IVB	Server JKT/SNB-EP	SNB	Locate-with	Count Domain	Metric Description	Threshold
FE	Frontend_Bound												IDQ_UC	SKL/SKX/(	Slots	This category represents slots fraction where the	> 0.2
FE		Frontend_Latency					#Pipeline_Width *	IDQ_UOPS_NOT_DELIVERED.C					#Pipelin	SKL/SKX/(	Slots	This metric represents slots fraction the CPU wa	> 0.15 & P
FE			ICache_Misses		ICACHE_16B.IF	DATA_STALL / CLKS		ICACHE.IFDATA		ICACHE.IFETCH			N/A	SKL/SKX/(	Clocks	This metric represents cycles fraction the CPU w	> 0.05 & P
FE			ITLB_Misses		ICACHE_64B.IF	TAG_STALL / CLKS							#ITLB_M	SKL/SKX/(	Clocks	This metric represents cycles fraction the CPU w	> 0.05 & P
FE			Branch_Resteers		(INT_MISC.CLEAR_RESTEER_CYCLES + 10* BACLEARS.ANY) / CLK								#Avg_R	BR_MISP_	Clocks	This metric represents cycles fraction the CPU w	> 0.05 & P; $issueBad; ~overl
FE			DSB_Switches										DSB2MI	SKL/SKX/(	Clocks	This metric represents cycles fraction the CPU w	> 0.05 & P
FE			MS_Switches										#MS_Sv	IDQ.MS_	Clocks	This metric estimates the fraction of cycles wher	> 0.05 & P; $issueMS
BAD		Frontend_Bandwidth											Fronten	SKL/SKX/(	Slots	This metric represents slots fraction the CPU wa	> 0.1 & (IPC > 2.0) & P
BAD	Bad_Speculation												(UOPS_	ISSUED.A	Slots	This category represents slots fraction wasted di	> 0.1; $issueBad
BAD		Branch_Mispredicts											#Mispr	BR_MISP_	Slots	This metric represents slots fraction the CPU has	> 0.05 & P
BAD		Machine_Clears											Bad_Sp	MACHINE	Slots	This metric represents slots fraction the CPU has	> 0.05 & P
BE	Backend_Bound												1 - (Frontend_Bo		Slots	This category represents slots fraction where no	> 0.2
BE/Mem		Memory_Bound											#Memory_Bound		Slots	This metric represents slots fraction the Memory	> 0.2 & P
BE/Mem			L1_Bound			(CYCLE_ACTIVITY.STALLS_MEM_A		(#STALLS_MEI					N/A	SKL/SKX/(	Clocks	This metric estimates how often the CPU was sta	(> 0.1 & P) | DTLB_Load;
BE/Mem				DTLB_Load	(#Mem_STLB_	(#Mem_STLB_Hit_Cost * DTLB_LOAD_MISSES.ST							(#Mem	SKL/SKX/(	Clocks	This metric represents cycles fraction where the	> 0.1
BE/Mem				FB_Full		Load_Miss_Real_La		Load_Miss_Re		Load_Miss_Real_			Latency * L1D_I		Clocks	This metric does a *rough estimation* of how of	> 0.3; $issueBW; $issueSL
BE/Mem			L2_Bound		(1 if FB_Full < 1	(CYCLE_ACTIVITY.STALLS_L1D_MI		(CYCLE_ACTIV					N/A	SKL/SKX/(	Clocks	This metric estimates how often the CPU was sta	> 0.1 & P
BE/Mem			L3_Bound		(CYCLE_ACTIV	#Mem_L3_Hit_Fraction * CYCLE_ACTIVITY.STALLS							#Mem_	SKL/SKX/(	Clocks	This metric estimates how often the CPU was sta	> 0.1 & P
BE/Mem			MEM_Bound		CYCLE_ACTIVIT	(1 - #Mem_L3_Hit_Fraction) * CYCLE_ACTIVITY.ST							(1 - #M	SKL/SKX/(	Clocks	This metric estimates how often the CPU was sta	> 0.1 & P
BE/Mem				MEM_Bandwidth									#ORO_DRD_BW_		Clocks	This metric estimates cycles fraction where the c	> 0.1 & P; $issueBW
BE/Mem				MEM_Latency									#ORO_DRD_Any_		Clocks	This metric estimates cycles fraction where the p	> 0.1 & P
BE/Mem			Store_Bound		EXE_ACTIVITY.BOUND_ON_STORES / CLKS								RESOU	SKL/SKX/(	Clocks	This metric estimates how often CPU was stalled	> 0.2 & P
BE/Core		Core_Bound											Backend_Bound -		Slots	This metric represents slots fraction where Core	> 0.2 & P
BE/Core			Divider		ARITH.DIVIDER	ARITH.FPU_DIV_AC	10 * ARITH.DIVIDER_UOPS / C			ARITH.F			SKL/SKX/(		Clocks	This metric represents cycles fraction where the	> 0.2 & P
BE/Core			Ports_Utilization		(#Backend_Bo	(#Backend_Bound_Cycles - RESOURCE_STALLS.SE							(#Backend_Boun		Clocks	This metric estimates cycles fraction the CPU pe	> 0.2 & P
RET	Retiring												UOPS_RETIRED.RI		Slots	by useful work i.e. issued uops that eventually	(> 0.7 | Microcode_Sequence
RET		Base											Retiring	INST_RET	Slots	This metric represents slots fraction where the C	> 0.6 & P
RET			FP_Arith				FP_Scalar + FP_Vec	N/A					FP_Scalar + FP_Ve		Uops	This metric represents overall arithmetic floating	> 0.2 & P
RET				FP_Scalar			(FP_ARITH_INST_R	N/A			(FP_COMP_OI		(FP_COMP_OPS_		Uops	This metric represents arithmetic floating-point i	> 0.1 & P
RET				FP_Vector			(FP_ARITH_INST_R	N/A			(FP_COMP_OI		(FP_COMP_OPS_		Uops	This metric represents arithmetic floating-point i	> 0.2 & P
RET			Other				1 - FP_Arith	N/A					1 - FP_Arith		Uops	This metric represents non-floating-point (FP) uc	> 0.3 & P
RET		Microcode_Sequencer											#Retire_	IDQ.MS_I	Slots	This metric represents slots fraction the CPU wa	> 0.05; $issueMS

Gooda

- Gooda contains lists of interesting events for analyzing performance of several architectures
- List of penalties for understanding performance on different architectures
- https://github.com/David-Levinthal/gooda



Hardware Counter High Level Issues

- Intel needs constructive feedback
  - people care about performance counters
  - changing list of events makes it hard to use
- Validating events for processors is difficult
  - finite resources for validation
  - some events go private when not useful

Recommendations for Tuning

- Use auto feedback driven optimization with Intel and GCC compilers
- Data that can be collected with perf
- perf record -e branch_instruction_retired_taken -b (uses LBR) -o -
  - puts results in a pipe mode
  - use scripts to convert this to a gcov file