4. Instruction tables

Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD, and VIA CPUs

By Agner Fog. Technical University of Denmark. Copyright © 1996 – 2020. Last updated 2020-10-11.

Introduction

This is the fourth in a series of five manuals:

- 1. Optimizing software in C++: An optimization guide for Windows, Linux, and Mac platforms.
- 2. Optimizing subroutines in assembly language: An optimization guide for x86 platforms.
- 3. The microarchitecture of Intel, AMD, and VIA CPUs: An optimization guide for assembly programmers and compiler makers.
- 4. Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD, and VIA CPUs.
- 5. Calling conventions for different C++ compilers and operating systems.

The latest versions of these manuals are always available from www.agner.org/optimize. Copyright conditions are listed below.

The present manual contains tables of instruction latencies, throughputs and micro-operation breakdown and other tables for x86 family microprocessors from Intel, AMD, and VIA.

The figures in the instruction tables represent the results of my measurements rather than the official values published by microprocessor vendors. Some values in my tables are higher or lower than the values published elsewhere. The discrepancies can be explained by the following factors:

- My figures are experimental values while figures published by microprocessor vendors may be based on theory or simulations.
- My figures are obtained with a particular test method under particular conditions. It is possible that different values can be obtained under other conditions.
- Some latencies are difficult or impossible to measure accurately, especially for memory access and type conversions that cannot be chained.
- Latencies for moving data from one execution unit to another are listed explicitly in some of my tables while they are included in the general latencies in some tables published by microprocessor vendors.

Most values are the same in all microprocessor modes (real, virtual, protected, 16-bit, 32-bit, 64-bit). Values for far calls and interrupts may be different in different modes. Call gates have not been tested.

Instructions with a LOCK prefix have a long latency that depends on cache organization and possibly RAM speed. If there are multiple processors or cores or direct memory access (DMA) devices, then all locked instructions will lock a cache line for exclusive access, which may involve RAM access. A LOCK prefix typically costs more than a hundred clock cycles, even on single-processor systems. This also applies to the XCHG instruction with a memory operand.

If any text in the pdf version of this manual is unreadable, then please refer to the spreadsheet version.

Copyright notice

This series of five manuals is copyrighted by Agner Fog. Public distribution and mirroring is not allowed. Non-public distribution to a limited audience for educational purposes is allowed. A creative commons license CC-BY-SA shall automatically come into force when I die. See https://creativecommons.org/licenses/by-sa/4.0/legalcode

Definition of terms

Instruction

The instruction name is the assembly code for the instruction. Multiple instructions or multiple variants of the same instruction may be joined into the same line. Instructions with and without a 'v' prefix to the name have the same values unless otherwise noted.

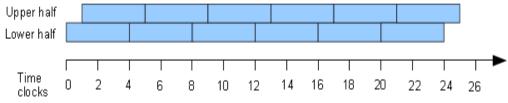
Operands

Operands can be different types of registers, memory, or immediate constants. Abbreviations used in the tables are: i = immediate constant, r = any general purpose register, r32 = 32-bit register, etc., mm = 64 bit mmx register, x or xmm = 128 bit xmm register, y = 256 bit ymm register, z = 512 bit zmm register, v = any vector register, sr = segment register, m = any memory operand including indirect operands, m64 means 64-bit memory operand, etc.

Latency

The latency of an instruction is the delay that the instruction generates in a dependency chain. The measurement unit is clock cycles. Where the clock frequency is varied dynamically, the figures refer to the core clock frequency. The numbers listed are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity may increase the latencies by possibly more than 100 clock cycles on many processors, except in move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results may give a similar delay. A missing value in the table means that the value has not been measured or that it cannot be measured in a meaningful way.

Some processors have a pipelined execution unit that is smaller than the largest register size so that different parts of the operand are calculated at different times. Assume, for example, that we have a long dependency chain of 128-bit vector instructions running in a fully pipelined 64-bit execution unit with a latency of 4. The lower 64 bits of each operation will be calculated at times 0, 4, 8, 12, 16, etc. And the upper 64 bits of each operation will be calculated at times 1, 5, 9, 13, 17, etc. as shown in the figure below. If we look at one 128-bit instruction in isolation, the latency will be 5. But if we look at a long chain of 128-bit instructions, the total latency will be 4 clock cycles per instruction plus one extra clock cycle in the end. The latency in this case is listed as 4 in the tables because this is the value it adds to a dependency chain.



Reciprocal throughput

ine throughput is the maximum number of instructions of the same kind that can be executed per clock cycle when the operands of each instruction are independent of the preceding instructions. The values listed are the reciprocals of the throughputs, i.e. the average number of clock cycles per instruction when the instructions are not part of a limiting dependency chain. For example, a reciprocal throughput of 2 for FMUL means that a new FMUL instruction can start executing 2 clock cycles after a previous FMUL. A reciprocal throughput of 0.33 for ADD means that the execution units can handle 3 integer additions per clock cycle.

The reason for listing the reciprocal values is that this makes comparisons between latency and throughput easier. The reciprocal throughput is also called issue latency. The values listed are for a single thread or a single core. A missing value in the table means that the value has not been measured.

Definition of terms

μops

Uop or μ op is an abbreviation for micro-operation. Processors with out-of-order cores are capable of splitting complex instructions into μ ops. For example, a read-modify instruction may be split into a read- μ op and a modify- μ op. The number of μ ops that an instruction generates is important when certain bottlenecks in the pipeline limit the number of μ ops per clock cycle.

Execution unit

The execution core of a microprocessor has several execution units. Each execution unit can handle a particular category of μ ops, for example floating point additions. The information about which execution unit a particular μ op goes to can be useful for two purposes. Firstly, two μ ops cannot execute simultaneously if they need the same execution unit. And secondly, some processors have a latency of an extra clock cycle when the result of a μ op executing in one execution unit is needed as input for a μ op in another execution unit.

Execution port

The execution units are clustered around a few execution ports on most Intel processors. Each µop passes through an execution port to get to the right execution unit. An execution port can be a bottleneck because it can handle only one µop at a time. Two µops cannot execute simultaneously if they need the same execution port, even if they are going to different execution units.

Instruction set

This indicates which instruction set an instruction belongs to. The instruction is only available in processors that support this instruction set. The most important instruction sets are listed on the next page. Availability in processors prior to 80386 does not apply for 32-bit and 64-bit operands. Availability in the MMX instruction set does not apply to 128-bit packed integer instructions, which require SSE2. Availability in the SSE instruction set does not apply to double precision floating point instructions, which require SSE2.

32-bit instructions are available in 80386 and later. 64-bit instructions in general purpose registers are available only under 64-bit operating systems. Instructions that use XMM registers (SSE and later), YMM registers (AVX and later), and ZMM registers (AVX512 and later) are only available under operating systems that support these register sets.

How the values were measured

The values in the tables are measured with the use of my own test programs, which are available from www.agner.org/optimize/testp.zip

The time unit for all measurements is CPU clock cycles. It is attempted to obtain the highest clock frequency if the clock frequency is varying with the workload. Many Intel processors have a performance counter named "core clock cycles". This counter gives measurements that are independent of the varying clock frequency. Where no "core clock cycles" counter is available, the "time stamp counter" is used (RDTSC instruction). In cases where this gives inconsistent results (e.g. in AMD Bobcat) it is necessary to make the processor boost the clock frequency by executing a large number of instructions (> 1 million) or turn off the power-saving features in the BIOS setup.

Instruction throughputs are measured with a long sequence of instructions of the same kind, where subsequent instructions use different registers in order to avoid dependence of each instruction on the previous one. The input registers are cleared in the cases where it is impossible to use different registers. The test code is carefully constructed in each case to make sure that no other bottleneck is limiting the throughput than the one that is being measured.

Instruction latencies are measured in a long dependency chain of identical instructions where the output of each instruction is used as input for the next instruction.

The sequence of instructions should be long, but not so long that it doesn't fit into the level-1 code cache. A typical length is 100 instructions of the same type. This sequence is repeated in a loop if a larger number of instructions is desired.

Definition of terms

It is not possible to measure the latency of a memory read or write instruction with software methods. It is only possible to measure the combined latency of a memory write followed by a memory read from the same address. What is measured here is not actually the cache access time, because in most cases the microprocessor is smart enough to make a "store forwarding" directly from the write unit to the read unit rather than waiting for the data to go to the cache and back again. The latency of this store forwarding process is arbitrarily divided into a write latency and a read latency in the tables. But in fact, the only value that makes sense to performance optimization is the sum of the write time and the read time.

A similar problem occurs where the input and the output of an instruction use different types of registers. For example, the MOVD instruction can transfer data between general purpose registers and XMM vector registers. The value that can be measured is the combined latency of data transfer from one type of registers to another type and back again (A \rightarrow B \rightarrow A). The division of this latency between the A \rightarrow B latency and the B \rightarrow A latency is sometimes obvious, sometimes based on guesswork, μ op counts, indirect evidence, or triangular sequences such as A \rightarrow B \rightarrow Memory \rightarrow A. In many cases, however, the division of the total latency between A \rightarrow B latency and B \rightarrow A latency is arbitrary. However, what cannot be measured cannot matter for performance optimization. What counts is the sum of the A \rightarrow B latency and the B \rightarrow A latency, not the individual terms.

The µop counts are usually measured with the use of the performance monitor counters (PMCs) that are built into modern microprocessors. The PMCs for VIA processors are undocumented, and the interpretation of these PMCs is based on experimentation.

The execution ports and execution units that are used by each instruction or µop are detected in different ways depending on the particular microprocessor. Some microprocessors have PMCs that can give this information directly. In other cases it is necessary to obtain this information indirectly by testing whether a particular instruction or µop can execute simultaneously with another instruction/µop that is known to go to a particular execution port or execution unit. On some processors, there is a delay for transmitting data from one execution unit (or cluster of execution units) to another. This delay can be used for detecting whether two different instructions/µops are using the same or different execution units.

Instruction sets

Instruction sets

Explanation of instruction sets for x86 processors

x86	This is the name of the common instruction set, supported by all processors in this lineage.
80186	This is the first extension to the x86 instruction set. New integer instructions: PUSH i, PUSHA, POPA, IMUL r,r,i, BOUND, ENTER, LEAVE, shifts and rotates by immediate ≠ 1.
80286	System instructions for 16-bit protected mode.
80386	The eight general purpose registers are extended from 16 to 32 bits. 32-bit addressing. 32-bit protected mode. Scaled index addressing. MOVZX, MOVSX, IMUL r,r, SHLD, SHRD, BT, BTR, BTS, BTC, BSF, BSR, SETcc.
80486	BSWAP. Later versions have CPUID.
x87	This is the floating point instruction set. Supported when a 8087 or later coprocessor is present. Some 486 processors and all processors since Pentium/ K5 have built-in support for floating point instructions without the need for a coprocessor.
80287	FSTSW AX
80387	FPREM1, FSIN, FCOS, FSINCOS.
Pentium	RDTSC, RDPMC.
PPro	Conditional move (CMOV, FCMOV) and fast floating point compare (FCOMI) instructions introduced in Pentium Pro. These instructions are not supported in Pentium MMX, but are supported in all processors with SSE and later.
MMX	Integer vector instructions with packed 8, 16 and 32-bit integers in the 64-bit MMX registers MM0 - MM7, which are aliased upon the floating point stack registers ST(0) - ST(7).
SSE	Single precision floating point scalar and vector instructions in the new 128-bit XMM registers XMM0 - XMM7. PREFETCH, SFENCE, FXSAVE, FXRSTOR, MOVNTQ, MOVNTPS. The use of XMM registers requires operating system support.
SSE2	Double precision floating point scalar and vector instructions in the 128-bit XMM registers XMM0 - XMM7. 64-bit integer arithmetics in the MMX registers. Integer vector instructions with packed 8, 16, 32 and 64-bit integers in the XMM registers. MOVNTI, MOVNTPD, PAUSE, LFENCE, MFENCE.
SSE3	FISTTP, LDDQU, MOVDDUP, MOVSHDUP, MOVSLDUP, ADDSUBPS, ADDSUPPD, HADDPS, HADDPD, HSUBPS, HSUBPD.
SSSE3	(Supplementary SSE3): PSHUFB, PHADDW, PHADDSW, PHADDD, PMADDUBSW, PHSUBW, PHSUBSW, PHSUBD, PSIGNB, PSIGNW, PSIGND, PMULHRSW, PABSB, PABSW, PABSD, PALIGNR.
64 bit	This instruction set is called x86-64, x64, AMD64 or EM64T. It defines a new 64-bit mode with 64-bit addressing and the following extensions: The general purpose registers are extended to 64 bits, and the number of general purpose registers is extended from eight to sixteen. The number of XMM registers is also extended from eight to sixteen, but the number of MMX and ST registers is still eight. Data can be addressed relative to the instruction pointer. There is no way to get access to these extensions in 32-bit mode Most instructions that involve segmentation are not available in 64 bit mode. Direct far jumps and calls are not allowed, but indirect far jumps, indirect far calls and far returns are allowed. These are used in system code for switching mode. Segment registers DS, ES, and SS cannot be used. The FS and GS segments and segment prefixes are available in 64 bit mode and are used for addressing thread environment blocks and processor environment blocks

Instruction sets

available in 64 bit mode

Instructions not The following instructions are not available in 64-bit mode: PUSHA, POPA, BOUND, INTO, BCD instructions: AAA, AAS, DAA, DAS, AAD, AAM, undocumented instructions (SALC, ICEBP, 82H alias for 80H opcode). SYSENTER, SYSEXIT, ARPL. On some early Intel processors, LAHF and SAHF are not available in 64 bit mode. Increment and decrement register instructions cannot be coded in the short one-byte opcode form because these codes have been reassigned as REX prefixes.

> Most instructions that involve segmentation are not available in 64 bit mode. Direct far jumps and calls are not allowed, but indirect far jumps, indirect far calls and far returns are allowed. These are used in system code for switching mode. PUSH CS, PUSH DS, PUSH ES, PUSH SS, POP DS, POP ES, POP SS, LDS and LES instructions are not allowed. CS, DS, ES and SS prefixes are allowed but ignored. The FS and GS segments and segment prefixes are available in 64 bit mode and are used for addressing thread environment blocks and processor environment blocks.

MPSADBW, PHMINPOSUW, PMULDQ, PMULLD, DPPS, DPPD, BLEND.., SSE4.1 PMIN.., PMAX.., ROUND.., INSERT.., EXTRACT.., PMOVSX.., PMOVZX.., PTEST, PCMPEQQ, PACKUSDW, MOVNTDQA

SSF42 CRC32, PCMPESTRI, PCMPESTRM, PCMPISTRI, PCMPISTRM, PCMPGTQ, POPCNT.

AESDEC, AESDECLAST, AESENC, AESENCLAST, AESIMC, AES AESKEYGENASSIST.

CLMUL PCLMULQDQ.

The sixteen 128-bit XMM registers are extended to 256-bit YMM registers with AVX room for further extension in the future. The use of YMM registers requires operating system support. Floating point vector instructions are available in 256bit versions. Almost all previous XMM instructions now have two versions: with and without zero-extension into the full YMM register. The zero-extension versions have three operands in most cases. Furthermore, the following instructions are added in AVX: VBROADCASTSS, VBROADCASTSD, VEXTRACTF128, VINSERTF128, VLDMXCSR, VMASKMOVPS, VMASKMOVPD, VPERMILPD, VPERMIL2PD, VPERMILPS, VPERMIL2PS, VPERM2F128, VSTMXCSR, VZEROALL, VZEROUPPER.

Integer vector instructions are available in 256-bit versions. Furthermore, the AVX2 following instructions are added in AVX2: ANDN, BEXTR, BLSI, BLSMSK, BLSR, BZHI, INVPCID, LZCNT, MULX, PEXT, PDEP, RORX, SARX, SHLX, SHRX, TZCNT, VBROADCASTI128, VBROADCASTSS, VBROADCASTSD, VEXTRACTI128, VGATHERDPD, VGATHERQPD, VGATHERDPS. VGATHERQPS, VPGATHERDD, VPGATHERQD, VPGATHERDQ, VPGATHERQQ, VINSERTI128, VPERM2I128, VPERMD, VPERMPD. VPERMPS, VPERMQ, VPMASKMOVD, VPMASKMOVQ, VPSLLVD, VPSLLVQ, VPSRAVD, VPSRLVD, VPSRLVQ,

(FMA): Fused multiply and add instructions: VFMADDxxxPD, VFMADDxxxPS, FMA3 VFMADDxxxSD, VFMADDxxxSS, VFMADDSUBxxxPD, VFMADDSUBxxxPS, VFMSUBADDxxxPD, VFMSUBADDxxxPS, VFMSUBxxxPD, VFMSUBxxxPS, VFMSUBxxxSD, VFMSUBxxxSS, VFNMADDxxxPD, VFNMADDxxPS, VFNMADDxxxSD, VFNMADDxxxSS, VFNMSUBxxxPD, VFNMSUBxxxPS, VFNMSUBxxxSD, VFNMSUBxxxSS.

FMA4 Same as Intel FMA, but with 4 different operands according to a preliminary Intel specification which is now supported only by some AMD processors. Intel's FMA specification has later been changed to FMA3, which is now also supported by AMD.

MOVBE MOVBE POPCNT POPCNT PCLMUL PCLMULQDQ

XSAVE

XSAVEOPT

RDRAND RDRAND

Instruction sets

RDSEED RDSEED

BMI1 ANDN, BEXTR, BLSI, BLSMSK, BLSR, LZCNT, TXCNT BMI2 BZHI, MULX, PDEP, PEXT, RORX, SARX, SHRX, SHLX

ADX ADCX, ADOX, CLAC

AVX512F The 256-bit YMM registers are extended to 512-bit ZMM registers. The number

of vector registers is extended to 32 in 64-bit mode, while there are still only 8 vector registers in 32-bit mode. 8 new vector mask registers k0 – k7. Masked vector instructions. Many new instructions. Single- and double precision floating point vectors are always supported. Other instructions are supported if the various optional AVX512 variants, listed below, are supported as well.

AVX512BW Vectors of 8-bit and 16-bit integers in ZMM registers.

AVX512DQ Some additional instructions with vectors of 32-bit and 64-bit integers in ZMM

registers.

AVX512VL The vector operations defined for 512-bit vectors in the various AVX512 subsets,

including masked operations, can be applied to 128-bit and 256-bit vectors as

well.

AVX512CD Conflict detection instructions

AVX512ER Approximate exponential function, reciprocal and reciprocal square root

AVX512PF Gather and scatter prefetch SHA Secure hash algorithm

MPX Memory protection extensions

SMAP CLAC, STAC

CVT16 VCVTPH2PS, VCVTPS2PH.

3DNow (AMD only. Obsolete). Single precision floating point vector instructions in the

64-bit MMX registers. Only available on AMD processors. The 3DNow

instructions are: FEMMS, PAVGUSB, PF2ID, PFACC, PFADD,

PFCMPEQ/GT/GE, PFMAX, PFMIN, PFRCP/IT1/IT2, PFRSQRT/IT1, PFSUB,

PFSUBR, PI2FD, PMULHRW, PREFETCH/W.

3DNowE (AMD only, Obsolete), PF2IW, PFNACC, PFPNACC, PI2FW, PSWAPD.

PREFETCHW This instruction has survived from 3DNow and now has its own feature name

PREFETCHWT1 PREFETCHWT1

SSE4A (AMD only). EXTRQ, INSERTQ, LZCNT, MOVNTSD, MOVNTSS, POPCNT.

(POPCNT shared with Intel SSE4.2).

XOP

(AMD only. Obsolete). VFRCZPD, VFRCZPS, VFRCZSD, VFRCZSS, VPCMOV,

VPCOMB, VPCOMD, VPCOMQ, PCOMW, VPCOMUB, VPCOMUD, VPCOMUQ, VPCOMUW, VPHADDBD, VPHADDBQ, VPHADDBW,

VPHADDDQ, VPHADDUBD, VPHADDUBQ, VPHADDUBW, VPHADDUDQ, VPHADDUWD, VPHADDUWQ, VPHADDWD, VPHADDWQ, VPHSUBBW, VPHSUBDQ, VPHSUBWD, VPMACSDD, VPMACSDQH, VPMACSDQL,

VPMACSSDD, VPMACSSDQH, VPMACSSDQL, VPMACSSWD,

 ${\sf VPMACSSWW}, {\sf VPMACSWD}, {\sf VPMACSWW}, {\sf VPMADCSSWD}, {\sf VPMADCSWD}, \\$

VPPERM, VPROTB, VPROTD, VPROTQ, VPROTW, VPSHAB, VPSHAD,

VPSHAQ, VPSHAW, VPSHLB, VPSHLD, VPSHLQ, VPSHLW.

Microprocessor versions tested

The tables in this manual are based on testing of the following microprocessors

Processor name	Microarchitecture Code name	Family number (hex)	Model number (hex)	Comment
AMD K7 Athlon		6	6	Step. 2, rev. A5
AMD K8 Opteron		F	5	Stepping A
AMD K10 Opteron		10	2	2350, step. 1
AMD Bulldozer	Bulldozer, Zambezi	15	1	FX-6100, step 2
AMD Piledriver	Piledriver	15	2	FX-8350, step 0. And others
AMD Steamroller	Steamroller, Kaveri	15	30	A10-7850K, step 1
AMD Excavator	Bristol Ridge	15	65	A10-9700E, step 1
AMD Ryzen	Zen 1	17	1	Ryzen 7 1800X, step 1
AMD Ryzen 3700	Zen 2	17	71	Ryzen 7 3700X. Step. 0
AMD Bobcat	Bobcat	14	1	E350, step. 0
AMD Kabini	Jaguar	16	0	A4-5000, step 1
Intel Pentium	P5	5	2	·
Intel Pentium MMX	P5	5	4	Stepping 4
Intel Pentium II	P6	6	6	
Intel Pentium III	P6	6	7	
Intel Pentium 4	Netburst	F	2	Stepping 4, rev. B0
Intel Pentium 4 EM64T	Netburst, Prescott	F	4	Xeon. Stepping 1
Intel Pentium M	Dothan	6	D	Stepping 6, rev. B1
Intel Core Duo	Yonah	6	E	Not fully tested
Intel Core 2 (65 nm)	Merom	6	F	T5500, Step. 6, rev. B2
Intel Core 2 (45 nm)	Wolfdale	6	17	E8400, Step. 6
Intel Core i7	Nehalem	6	1A	i7-920, Step. 5, rev. D0
Intel 2nd gen. Core	Sandy Bridge	6	2A	i5-2500, Step 7
Intel 3rd gen. Core	Ivy Bridge	6	3A	i7-3770K, Step 9
Intel 4th gen. Core	Haswell	6	3C	i7-4770K, step. 3
Intel 5th gen. Core	Broadwell	6	56	D1540, step 2
Intel 6th gen. Core	Skylake	6	5E	Step. 3
Intel 7th gen. Core	Skylake-X	6	55	Step. 4
Intel 9th gen. Core	Coffee Lake	6	9E	Step. B
Intel Atom 330	Diamondville	6	1C	Step. 2
Intel Bay Trail	Silvermont	6	37	Step. 3
Intel Apollo Lake	Goldmont	6	5C	Step. 9
Intel Gemini Lake	Goldmont Plus	6	7A	Step. 1
Intel Xeon Phi	Knights Landing	6	57	Step. 1
VIA Nano L2200		6	F	Step. 2
VIA Nano L3050	Isaiah	6	F	Step. 8 (prerelease sample)

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB,

JNE, etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, xmm = 128 bit xmm register, sr = segment register, m = any memory operand including indirect operands, m64 means 64-bit memory oper-

and, etc.

Ops: Number of macro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 macro-operations use microcode.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latency listed does not include the memory oper-

and where the operand is listed as register or memory (r/m).

Reciprocal throughput: This is also called issue latency. This value indicates the average number of

clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However, the throughput may be limited by other bottlenecks in the

pipeline.

Execution unit: Indicates which execution unit is used for the macro-operations. ALU means

any of the three integer ALU's. ALU0_1 means that ALU0 and ALU1 are both used. AGU means any of the three integer address generation units. FADD means floating point adder unit. FMUL means floating point multiplier unit. FMISC means floating point store and miscellaneous unit. FA/M means FADD or FMUL is used. FANY means any of the three floating point units can be used. Two macro-operations can execute simultaneously if they go to different

execution units.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions						
MOV	r,r	1	1	1/3	ALU	
MOV	r,i	1	1	1/3	ALU	
						Any addr. mode. Add 1 clk if code segment base ≠
MOV	r8,m8	1	4	1/2	ALU, AGU	0
MOV	r16,m16	1	4	1/2	ALU, AGU	do.
MOV	r32,m32	1	3	1/2	AGU	do.
MOV	m8,r8H	1	8	1/2	AGU	AH, BH, CH, DH
MOV	m8,r8L	1	2	1/2	AGU	Any other 8-bit register Any addressing
MOV	m16/32,r	1	2	1/2	AGU	mode
MOV	m,i	1	2	1/2	AGU	
MOV	r,sr	1	2	1		
MOV	sr,r/m	6	9-13	8		
MOVZX, MOVSX	r,r	1	1	1/3	ALU	
MOVZX, MOVSX	r,m	1	4	1/2	ALU, AGU	

CMOVcc	r,r	1	1	1/3	ALU	
CMOVcc	r,m	1		1/2	ALU, AGU	
XCHG	r,r	3	2	1	ALU	
ACITO	1,1	3		'	ALU	Tii
XCHG		2	46	16	ALLI ACII	Timing depends
	r,m	3	16	16	ALU, AGU	on hw
XLAT		2	5		ALU, AGU	
PUSH	r	1		1	ALU, AGU	
PUSH	i	1		1	ALU, AGU	
PUSH	m	2		1	ALU, AGU	
PUSH	sr	2		1	ALU, AGU	
PUSHF(D)		1		1	ALU, AGU	
PUSHA(D)		9		4	ALU, AGU	
1 1						
POP	r	2		1	ALU, AGU	
POP	m	3		1	ALU, AGU	
POP	DS/ES/FS/GS	6		10	ALU, AGU	
POP	SS	9		18	ALU, AGU	
POPF(D)		2		1	ALU, AGU	
POPA(D)		9		4	ALU, AGU	
LEA	r16,[m]	2	3	1	AGU	Any addr. size
LEA	r32,[m]	1	2	1/3	AGU	Any addr. size
	132,[11]					Ally addit Size
LAHF		4	3	2	ALU	
SAHF		2	2	2	ALU	
SALC		1	1	1	ALU	
LDS, LES,	r,m	10		9		
BSWAP	r	1	1	1/3	ALU	
Arithmetic instructions						
ADD, SUB	r,r/i	1	1	1/3	ALU	
ADD, SUB		1	1	1/2		
	r,m	-			ALU, AGU	
ADD, SUB	m,r	1	7	2.5	ALU, AGU	
ADC, SBB	r,r/i	1	1	1/3	ALU	
ADC, SBB	r,m	1	1	1/2	ALU, AGU	
ADC, SBB	m,r/i	1	7	2.5	ALU, AGU	
CMP	r,r/i	1	1	1/3	ALU	
CMP	r,m	1		1/2	ALU, AGU	
INC, DEC, NEG	r	1	1	1/3	ALU	
INC, DEC, NEG	m i	1	7	3	ALU, AGU	
	'''	9	5	5		
AAA, AAS					ALU	
DAA		12	6	6	ALU	
DAS		16	7	7	ALU	
AAD		4	5		ALU0	
AAM		31	13		ALU	
MUL, IMUL	r8/m8	3	3	2	ALU0	
						latency ax=3,
MUL, IMUL	r16/m16	3	3	2	ALU0_1	dx=4
MUL, IMUL	r32/m32	3	4	3	ALU0_1	
IMUL	r16,r16/m16	2	3	2	ALU0	
IMUL	r32,r32/m32	2	4	2.5	ALU0	
IMUL	· ·	2	4	2.3 1	ALU0	
	r16,(r16),i					
IMUL	r32,(r32),i	2	5	2	ALU0	
IMUL	r16,m16,i	3		2	ALU0	
IMUL	r32,m32,i	3		2	ALU0	
DIV	r8/m8	32	24	23	ALU	
	'		'			,

DIV	r16/m16	47	24	23	ALU
DIV	r32/m32	79	40	40	ALU
IDIV	r8	41	17	17	ALU
IDIV	r16	56	25	25	ALU
IDIV	r32	88	41	41	ALU
IDIV	m8	42	17	17	ALU
IDIV	m16	57	25	25	ALU
IDIV	m32	89	41	41	ALU
CBW, CWDE		1	1	1/3	ALU
CWD, CDQ		1	1	1/3	ALU
3112, 324				.,,	,
Logic instructions			_		
AND, OR, XOR	r,r	1	1	1/3	ALU
AND, OR, XOR	r,m	1	1	1/2	ALU, AGU
AND, OR, XOR	m,r	1	7	2.5	ALU, AGU
TEST	r,r	1	1	1/3	ALU
TEST	r,m	1	1	1/2	ALU, AGU
NOT	r	1	1	1/3	ALU
NOT	m	1	7	2.5	ALU, AGU
SHL, SHR, SAR	r,i/CL	1	1	1/3	ALU
ROL, ROR	r,i/CL	1	1	1/3	ALU
RCL, RCR		1	1	1/3	ALU
1	r,1				
RCL	r,i	9	4	4	ALU
RCR	r,i	7	3	3	ALU
RCL	r,CL	9	3	3	ALU
RCR	r,CL	7	3	3	ALU
SHL,SHR,SAR,ROL,ROR	m,i /CL	1	7	3	ALU, AGU
	· ·	1	7	4	
RCL, RCR	m,1				ALU, AGU
RCL	m,i	10	5	4	ALU, AGU
RCR	m,i	9	8	4	ALU, AGU
RCL	m,CL	9	6	4	ALU, AGU
RCR	m,CL	8	7	3	ALU, AGU
SHLD, SHRD	r,r,i	6	4	2	ALU
				3	
SHLD, SHRD	r,r,cl	7	4		ALU
SHLD, SHRD	m,r,i/CL	8	7	3	ALU, AGU
BT	r,r/i	1	1	1/3	ALU
BT	m,i	1		1/2	ALU, AGU
ВТ	m,r	5		2	ALU, AGU
BTC, BTR, BTS	r,r/i	2	2	1	ALU
			1		
BTC	m,i	5	7	2	ALU, AGU
BTR, BTS	m,i	4	7	2	ALU, AGU
BTC, BTR, BTS	m,r	8	6	3	ALU, AGU
BSF	r,r	19	7	7	ALU
BSR	r,r	23	9	9	ALU
BSF		20	8	8	ALU, AGU
	r,m				
BSR	r,m	23	10	10	ALU, AGU
SETcc	r	1	1	1/3	ALU
SETcc	m	1		1/2	ALU, AGU
CLC, STC		1		1/3	ALU
CMC		1	1	1/3	ALU
			'		
CLD		2		1	ALU
STD		3		2	ALU

Control transfer instruction	ons					
JMP	short/near	1		2	ALU	
	_					low values = real
JMP	far	16-20	23-32	_		mode
JMP	r	1		2	ALU	
JMP	m(near)	1		2	ALU, AGU	
JMP	(for)	17-21	25-33			low values = real
	m(far)	1	25-33	1/2 2	A1.11	mode
Jcc	short/near	1		1/3 - 2	ALU	rcp. t.= 2 if jump
J(E)CXZ	short	2	0.4	1/3 - 2	ALU	rcp. t.= 2 if jump
LOOP	short	7	3-4	3-4	ALU	
CALL	near	3	2	2	ALU	
CALL	far	16-22	23-32			low values = real mode
CALL	r	4	3	3	ALU	mode
CALL	m(near)	5	3	3	ALU, AGU	
CALL	m(near)		3	3	ALU, AGU	low values = real
CALL	m(far)	16-22	24-33			mode
RETN	()	2	3	3	ALU	
RETN	i	2	3	3	ALU	
			-			low values = real
RETF		15-23	24-35			mode
						low values = real
RETF	i	15-24	24-35			mode
IRET		32	81			real mode
INT	İ	33	42			real mode
						values are for no
BOUND	m	6		2		jump
INTO		2		2		values are for no
INTO						jump
String instructions						
LODS		4	2	2		
REP LODS		5	2	2		values per count
STOS		4	2	2		'
REP STOS		3	1	1		values per count
MOVS		7	3	3		•
REP MOVS		4	1-4	1-4		values per count
SCAS		5	2	2		
REP SCAS		5	2	2		values per count
CMPS		7	6	6		Tanada par adam
REP CMPS		6	3-4	3-4		values per count
Other						
NOP (90)		1	0	1/3	ALU	
Long NOP (0F 1F)		1	0	1/3	ALU	
ENTER		i,0	12	12	12	
				_		3 ops, 5 clk if 16
LEAVE		3		3		bit
CLI		8-9		5		
STI		16-17		27		
CPUID		19-28	44-74			
RDTSC		5		11		
RDPMC		9		11		

Floating point x87 instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions						
FLD	r	1	2	1/2	FA/M	
FLD	m32/64	1	4	1/2	FANY	
FLD	m80	7	16	4		
FBLD	m80	30	41	39		
FST(P)	r	1	2	1/2	FA/M	
FST(P)	m32/64	1	3	1	FMISC	
FSTP	m80	10	7	5	1 WIGO	
FBSTP	m80	260	'	188		
FXCH				0.4		
	r	1	0		ENHOO	
FILD	m	1	9	1	FMISC	
FIST(P)	m	1	7	1	FMISC, FA/M	
FLDZ, FLD1		1		1	FMISC	
						Low latency im- mediately after
FCMOVcc	st0,r	9	6	5	FMISC, FA/M	FCOMI
FFREE	r	1		1/3	FANY	
FINCSTP, FDECSTP		1	0	1/3	FANY	
				.,,	. ,	Low latency im-
						mediately after
FNSTSW	AX	2	6-12	12	FMISC. ALU	FCOM FTST
FSTSW	AX	3	6-12	12	FMISC, ALU	
FNSTSW	m16	2	•	8	FMISC, ALU	do.
FNSTCW	m16	3		1	FMISC, ALU	40.
NOTOW	11110	3		'	I WIGO, ALO	faster if
FLDCW	m16	14		42	FMISC, ALU	unchanged
Arithmetic instructions						
FADD(P),FSUB(R)(P)	r/m	1	4	1	FADD	
FIADD,FISUB(R)	m	2	4	1-2	FADD,FMISC	
FMUL(P)	r/m	1	4	1	FMUL	
FIMUL		2	4	2		
IIVIUL	m	2	4	_	FMUL,FMISC	
FDIV(R)(P)	r/m	1	11-25	8-22	FMUL	Low values are for round divisors
FIDIV(R)(F)		1		9-23		
` '	m	2	12-26	1 .	FMUL,FMISC	do.
FABS, FCHS		1	2	1	FMUL	
FCOM(P), FUCOM(P)	r/m	1	2	1	FADD	
FCOMPP, FUCOMPP		1	2	1	FADD	
FCOMI(P)	r	1	3	1	FADD	
FICOM(P)	m	2		1	FADD, FMISC	
-TST		1	2	1	FADD	
FXAM		2		2	FMISC, ALU	
FRNDINT		5	10	3		
FPREM		1	7-10	8	FMUL	
FPREM1		1	8-11	8	FMUL	
Math						
FSQRT		1	35	12	FMUL	
FSIN		44	90-100			

FCOS		51	90-100			
FSINCOS		76	100-150			
FPTAN		46	100-200			
FPATAN		72	160-170			
FSCALE		5	8			
FXTRACT		7	11			
F2XM1		8	27			
FYL2X		49	126			
FYL2XP1		63	147			
Other						
FNOP		1	0	1/3	FANY	
(F)WAIT		1	0	1/3	ALU	
FNCLEX		7		24	FMISC	
FNINIT		25		92	FMISC	
FNSAVE		76		147		
FRSTOR		65		120		
1 1 1 2 1 2 1 1	I I					
FXSAVE		44		59		

Integer MMX instructions

Integer MMX instruct	1	0		D ! :	F	N. 4
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions						
MOVD	r32, mm	2	7	2	FMICS, ALU	
MOVD	mm, r32	2	9	2	FANY, ALU	
MOVD	mm,m32	1		1/2	FANY	
MOVD	m32, r	1		1	FMISC	
MOVQ	mm,mm	1	2	1/2	FA/M	
MOVQ	mm,m64	1		1/2	FANY	
MOVQ	m64,mm	1		1	FMISC	
MOVNTQ	m,mm	1		2	FMISC	
PACKSSWB/DW						
PACKUSWB	mm,r/m	1	2	2	FA/M	
PUNPCKH/LBW/WD	mm,r/m	1	2	2	FA/M	
PSHUFW	mm,mm,i	1	2	1/2	FA/M	
MASKMOVQ	mm,mm	32		24		
PMOVMSKB	r32,mm	3		3	FADD	
PEXTRW	r32,mm,i	2	5	2	FMISC, ALU	
PINSRW	mm,r32,i	2	12	2	FA/M	
Arithmetic instructions						
PADDB/W/D PADDSB/W						
PADDUSB/W PSUBB/W/						
D PSUBSB/W PSUBUSB/	mm r/m	1	2	1/2	FA/M	
N DOMDEO/OT BANAD	mm,r/m	1	2 2			
PCMPEQ/GT B/W/D	mm,r/m	l	2	1/2	FA/M	
PMULLW PMULHW PMULHUW	mm,r/m	1	3	1	FMUL	
PMADDWD	mm,r/m	1	3	1	FMUL	
PAVGB/W	mm,r/m	1	2	1/2	FA/M	
PMIN/MAX SW/UB	mm,r/m	1	2	1/2	FA/M	
PSADBW	mm,r/m	'1	3	1/2	FADD	
r SADBW	111111,1/111	1	3	1	FADD	

AMD K7										
Logic PAND PANDN POR PXOR PSLL/RLW/D/Q PSRAW/ D	mm,r/m mm,i/mm/m	1	2	1/2 1/2	FA/M FA/M					
Other EMMS		1		1/3	FANY					

Floating point XMM instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
		<u> </u>		throughput	unit	
Move instructions						
MOVAPS	r,r	2	2	1	FA/M	
MOVAPS	r,m	2		2	FMISC	
MOVAPS	m,r	2		2	FMISC	
MOVUPS	r,r	2	2	1	FA/M	
MOVUPS	r,m	5		2		
MOVUPS	m,r	5		2		
MOVSS	r,r	1	2	1	FA/M	
MOVSS	r,m	2	4	1	FANY FMISC	
MOVSS	m,r	1	3	1	FMISC	
MOVHLPS, MOVLHPS	r,r	1	2	1/2	FA/M	
MOVHPS, MOVLPS	r,m	1		1/2	FMISC	
MOVHPS, MOVLPS	m,r	1		1	FMISC	
MOVNTPS	m,r	2		4	FMISC	
MOVMSKPS	r32,r	3		2	FADD	
SHUFPS	r,r/m,i	3	3	3	FMUL	
UNPCK H/L PS	r,r/m	2	3	3	FMUL	
	1,,,,	-			102	
Conversion						
CVTPI2PS	xmm,mm	1	4		FMISC	
CVT(T)PS2PI	mm,xmm	1	6		FMISC	
CVTSI2SS	xmm,r32	4		10	FMISC	
CVT(T)SS2SI	r32,xmm	2		3	FMISC	
011(1)00201	102,7	-				
Arithmetic						
ADDSS SUBSS	r,r/m	1	4	1	FADD	
ADDPS SUBPS	r,r/m	2	4	2	FADD	
MULSS	r,r/m	1	4	1	FMUL	
MULPS	r,r/m	2	4	2	FMUL	
	, ,					Low values are
						for round divi-
						sors, e.g. powers
DIVSS	r,r/m	1	11-16	8-13	FMUL	of 2.
DIVPS	r,r/m	2	18-30	18-30	FMUL	do.
RCPSS	r,r/m	1	3	1	FMUL	
RCPPS	r,r/m	2	3	2	FMUL	
MAXSS MINSS	r,r/m	1	2	1	FADD	
MAXPS MINPS	r,r/m	2	2	2	FADD	
CMPccSS	r,r/m	1	2	1	FADD	

CMPccPS	r,r/m	2	2	2	FADD	
COMISS UCOMISS	r,r/m	1	2	1	FADD	
Logic	-					
ANDPS/D ANDNPS/D ORPS/D XORPS/D	r,r/m	2	2	2	FMUL	
Math						
SQRTSS	r,r/m	1	19	16	FMUL	
SQRTPS	r,r/m	2	36	36	FMUL	
RSQRTSS	r,r/m	1	3	1	FMUL	
RSQRTPS	r,r/m	2	3	2	FMUL	
Other						
LDMXCSR	m	8		9		
STMXCSR	m	3		10		

3DNow instructions (obsolete)

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move and convert instruc	ctions					
PREFETCH(W)	m	1		1/2	AGU	
PF2ID	mm,mm	1	5	1	FMISC	
PI2FD	mm,mm	1	5	1	FMISC	
PF2IW	mm,mm	1	5	1	FMISC	3DNow E
PI2FW	mm,mm	1	5	1	FMISC	3DNow E
PSWAPD	mm,mm	1	2	1/2	FA/M	3DNow E
Integer instructions						
PAVGUSB	mm,mm	1	2	1/2	FA/M	
PMULHRW	mm,mm	1	3	1	FMUL	
Floating point instruction	 IS					
PFADD/SUB/SUBR	mm,mm	1	4	1	FADD	
PFCMPEQ/GE/GT	mm,mm	1	2	1	FADD	
PFMAX/MIN	mm,mm	1	2	1	FADD	
PFMUL	mm,mm	1	4	1	FMUL	
PFACC	mm,mm	1	4	1	FADD	
PFNACC, PFPNACC	mm,mm	1	4	1	FADD	3DNow E
PFRCP	mm,mm	1	3	1	FMUL	
PFRCPIT1/2	mm,mm	1	4	1	FMUL	
PFRSQRT	mm,mm	1	3	1	FMUL	
PFRSQIT1	mm,mm	1	4	1	FMUL	
Other						
FEMMS	mm,mm	1		1/3	FANY	

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB, JNE,

etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, xmm = 128 bit xmm register, sr = segment register, m = any memory operand including indirect operands, m64 means 64-bit memory operand, etc.

Ops: Number of macro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 macro-operations use microcode.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latency listed does not include the memory operand where the oper-

and is listed as register or memory (r/m).

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However, the throughput may be limited by other bottlenecks in the pipeline.

Execution unit: Indicates which execution unit is used for the macro-operations. ALU means any

of the three integer ALU's. ALU0_1 means that ALU0 and ALU1 are both used. AGU means any of the three integer address generation units. FADD means floating point adder unit. FMUL means floating point multiplier unit. FMISC means floating point store and miscellaneous unit. FA/M means FADD or FMUL is used. FANY means any of the three floating point units can be used. Two macro-opera-

tions can execute simultaneously if they go to different execution units.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions				tinougnput	dine	
MOV	r,r	1	1	1/3	ALU	
MOV	r,i	1	1	1/3	ALU	
MOV	r8,m8	1	4	1/2	ALU, AGU	Any addressing mode.
MOV	r16,m16	1	4	1/2	ALU, AGU	Add 1 clock if code
MOV	r32,m32	1	3	1/2	AGU	segment base ≠ 0
MOV	r64,m64	1	3	1/2	AGU	
MOV	m8,r8H	1	8	1/2	AGU	AH, BH, CH, DH
						Any other 8-bit regis-
MOV	m8,r8L	1	3	1/2	AGU	ter
MOV	m16/32/64,r	1	3	1/2	AGU	Any addressing mode
MOV	m,i	1	3	1/2	AGU	
MOV	m64,i32	1	3	1/2	AGU	
MOV	r,sr	1	2	1/2-1		
MOV	sr,r/m	6	9-13	8		
MOVNTI	m,r	1		2-3	AGU	
MOVZX, MOVSX	r,r	1	1	1/3	ALU	
MOVZX, MOVSX	r,m	1	4	1/2	ALU, AGU	
MOVSXD	r64,r32	1	1	1/3	ALU	

MOVSXD	r64,m32	1		1/2	ALU, AGU	
CMOVcc	· ·	1	1	1/2	ALU, AGU	
CMOVcc	r,r		1			
	r,m	1		1/2	ALU, AGU	
XCHG	r,r	3	2	1	ALU	Tii
XCHG	r m	3	16	16	ALU, AGU	Timing depends on hw
XLAT	r,m	2	5	10		IIVV
	_			4	ALU, AGU	
PUSH	r	1	1	1	ALU, AGU	
PUSH	i	1	1	1	ALU, AGU	
PUSH	m	2	1	1	ALU, AGU	
PUSH	sr	2	1	1	ALU, AGU	
PUSHF(D/Q)		5	2	2	ALU, AGU	
PUSHA(D)		9	4	4	ALU, AGU	
POP	r	2	1	1	ALU, AGU	
POP	m	3	1	1	ALU, AGU	
POP	DS/ES/FS/GS	4-6	8	8	ALU, AGU	
POP	SS	7-9	28	28	ALU, AGU	
POPF(D/Q)		25	10	10	ALU, AGU	
POPA(D)		9	4	4	ALU, AGU	
LEA	r16,[m]	2	3	1	AGU	Any address size
LEA	r32,[m]	1	2	1/3	AGU	Any address size
LEA	r64,[m]	1	2	1/3	AGU	Any address size
LAHF	104,[111]	4	3	2	ALU	Ally addices size
SAHF			1	1/3	ALU	
		1				
SALC		1	1	1/3	ALU	
LDS, LES,	r,m	10		9		
BSWAP	r	1	1	1/3	ALU	
PREFETCHNTA	m	1		1/2	AGU	
PREFETCHT0/1/2	m	1		1/2	AGU	
SFENCE		6		8		
LFENCE		1		5		
MFENCE		7		16		
IN	r,i/DX	270				
OUT	i/DX,r	300				
Arithmetic instruction	S					
ADD, SUB	r,r/i	1	1	1/3	ALU	
ADD, SUB	r,m	1	1	1/2	ALU, AGU	
ADD, SUB	m,r	1	7	2.5	ALU, AGU	
ADC, SBB	r,r/i	1	1	1/3	ALU	
ADC, SBB	r,m	1	1	1/2	ALU, AGU	
ADC, SBB	m,r/i	1	7	2.5	ALU, AGU	
CMP	r,r/i	1	1	1/3	ALU	
CMP	r,m	1	'	1/2	ALU, AGU	
		-	1	1/2	ALU, AGU ALU	
INC, DEC, NEG	r	1	1			
INC, DEC, NEG	m	1	7	3	ALU, AGU	
AAA, AAS		9	5	5	ALU	
DAA		12	6	6	ALU	
DAS		16	7	7	ALU	
AAD		4	5		ALU0	
AAM		31	13		ALU	
MUL, IMUL	r8/m8	1	3	1	ALU0	
MUL, IMUL	r16/m16	3	3-4	2	ALU0_1	latency ax=3, dx=4

NALII INALII	r22/m22	2	l 3	4	ALLIO 1	1
MUL, IMUL	r32/m32	2 2	3	1 2	ALU0_1	leteness messed melse-E
MUL, IMUL	r64/m64		4-5		ALU0_1	latency rax=4, rdx=5
IMUL	r16,r16/m16	1	3	1	ALU0	
IMUL	r32,r32/m32	1	3	1	ALU0	
IMUL	r64,r64/m64	1	4	2	ALU0_1	
IMUL	r16,(r16),i	2	4	1	ALU0	
IMUL	r32,(r32),i	1	3	1	ALU0	
IMUL	r64,(r64),i	1	4	2	ALU0	
IMUL	r16,m16,i	3		2	ALU0	
IMUL	r32,m32,i	3		2	ALU0	
IMUL	r64,m64,i	3		2	ALU0_1	
DIV	r8/m8	31	15	15	ALU	
DIV	r16/m16	46	23	23	ALU	
DIV	r32/m32	78	39	39	ALU	
DIV	r64/m64	143	71	71	ALU	
IDIV	r8	40	17	17	ALU	
IDIV	r16	55	25	25	ALU	
IDIV	r32	87	41	41	ALU	
IDIV	r64	152	73	73	ALU	
IDIV	m8	41	17	17	ALU	
IDIV	m16	56	25	25	ALU	
IDIV	m32	88	41	41	ALU	
IDIV	m64	153	73	73	ALU	
CBW, CWDE, CDQE		1	1	1/3	ALU	
CWD, CDQ, CQO		1	1	1/3	ALU	
Logic instructions						
AND, OR, XOR	r,r	1	1	1/3	ALU	
AND, OR, XOR	r,m	1	1	1/2	ALU, AGU	
AND, OR, XOR	m,r	1	7	2.5	ALU, AGU	
TEST	r,r	1	1	1/3	ALU	
TEST	r,m	1	1	1/2	ALU, AGU	
NOT	r	1	1	1/3	ALU	
NOT	m	1	7	2.5	ALU, AGU	
SHL, SHR, SAR	r,i/CL	1	1	1/3	ALU	
ROL, ROR	r,i/CL	1	1	1/3	ALU	
RCL, RCR	r,1	1	1	1/3	ALU	
RCL	r,i	9	3	3	ALU	
RCR	r,i	7	3	3	ALU	
RCL	r,CL	9	4	4	ALU	
RCR	r,CL	7	3	3	ALU	
SHL,SHR,SAR,ROL,R	01	4	_			
OR	m,i /CL	1	7	3	ALU, AGU	
RCL, RCR	m,1	1	7	4	ALU, AGU	
RCL	m,i	10	9	4	ALU, AGU	
RCR	m,i	9	8	4	ALU, AGU	
RCL	m,CL	9	7	4	ALU, AGU	
RCR	m,CL	8	8	3	ALU, AGU	
SHLD, SHRD	r,r,i	6	3	3	ALU	
SHLD, SHRD	r,r,cl	7	3	3	ALU	
SHLD, SHRD	m,r,i/CL	8	6	3	ALU, AGU	
BT	r,r/i	1	1	1/3	ALU	
BT	m,i	1		1/2	ALU, AGU	

STD 2 1/3 ALU	BT BTC, BTR, BTS BTC BTR, BTS BTC BTR, BTS BSF BSF BSF BSF BSF BSF BSF CC CLC, STC CMC CLD	m,r r,r/i m,i m,i m,r m,r r16/32,r r64,r r,r r16,m r32,m r64,m r,m r	5 2 5 4 8 8 21 22 28 20 22 25 28 1 1 1	2 7 7 5 8 9 10 8 9 10 10	2 1 2 2 5 3 8 9 10 8 9 10 10 1/3 1/2 1/3 1/3	ALU, AGU ALU, AGU ALU, AGU ALU, AGU ALU, AGU ALU ALU ALU ALU, AGU ALU ALU	
	STD		2		1/3	ALU	
Control transfer instructions	Control transfer instru	 ctions					
JMP short/near 1 2 ALU			1		2	ALU	
low values = real	18.45		40.00	00.00			I I
JMP far 16-20 23-32 mode JMP r 1 2 ALU	1			23-32	2	A1.11	mode
JMP r 1 2 ALU JMP m(near) 1 2 ALU, AGU		· ·					
low values = real	OWN	m(near)	'			ALO, AGO	low values = real
JMP m(far) 17-21 25-33 mode	JMP	m(far)	17-21	25-33			
Jccshort/near11/3 - 2ALUrecip. thrp.= 2 if jump		short/near					recip. thrp.= 2 if jump
J(E/R)CXZ short 2 1/3 - 2 ALU recip. thrp.= 2 if jump							recip. thrp.= 2 if jump
LOOP short 7 3-4 ALU	1	short					
CALL near 3 2 2 ALU	CALL	near	3	2	2	ALU	
CALL far 16-22 23-32 low values = real mode	CALL	far	16-22	23-32			
CALL r 4 3 3 ALU	1				3	ΔΙΙΙ	mode
CALL m(near) 5 3 ALU, AGU							
low values = real	J. 1.22	(, ,_0,,,,00	low values = real
CALL m(far) 16-22 24-33 mode		m(far)	16-22	24-33			mode
RETN 2 3 3 ALU							
RETN i 2 3 3 ALU	RETN	i	2	3	3	ALU	
RETF 15-23 24-35 low values = real mode	RETF		15-23	24-35			mode
	RETE	i	15_2/	24-35			I I
IRET 32 81 real mode	1	'					
INT i 33 42 real mode	1	i					
		·					l odi modo
BOUND m 6 2 values are for no jump	BOUND	m	6		2		values are for no jump
INTO 2 values are for no jump	INTO		2		2		values are for no jump
String instructions	String instructions						
LODS 4 2 2		1	4	2	2		
REP LODS 5 2 2 values are per count							values are per count

STOS		4	2	2		
REP STOS	•	1.5 - 2	0.5 - 1	0.5 - 1		values are per count
MOVS		7	3	3		
REP MOVS		3	1-2	1-2		values are per count
SCAS		5	2	2		
REP SCAS		5	2	2		values are per count
CMPS		2	3	3		
REP CMPS		6	2	2		values are per count
Other						
NOP (90)		1	0	1/3	ALU	
Long NOP (0F 1F)		1	0	1/3	ALU	
ENTER		i,0	12	12	12	
LEAVE		2		3		3 ops, 5 clk if 16 bit
CLI		8-9		5		
STI		16-17		27		
CPUID		22-50	47-164			
RDTSC		6	10	7		
RDPMC		9	12	7		

Floating point x87 instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions						
FLD	r	1	2	1/2	FA/M	
FLD	m32/64	1	4	1/2	FANY	
FLD	m80	7	16	4		
FBLD	m80	30	41	39		
FST(P)	r	1	2	1/2	FA/M	
FST(P)	m32/64	1	3	1	FMISC	
FSTP	m80	10	7	5		
FBSTP	m80	260	173	160		
FXCH	r	1	0	0.4		
FILD	m	1	9	1	FMISC	
FIST(P)	m	1	7	1	FMISC, FA/M	
FLDZ, FLD1		1		1	FMISC	
						Low latency immedi-
FCMOVcc	st0,r	9	4-15	4	FMISC, FA/M	ately after FCOMI
FFREE	r	1		2	FANY	
FINCSTP, FDECSTP		1	0	1/3	FANY	
						Low latency immediately after FCOM
FNSTSW	AX	2	6-12	12	FMISC, ALU	FTST
FSTSW	AX	3	6-12	12	FMISC, ALU	do.
FNSTSW	m16	2		8	FMISC, ALU	do.
FNSTCW	m16	3		1	FMISC, ALU	
FLDCW	m16	18		50	FMISC, ALU	faster if unchanged
Arithmetic instruction	s					
FADD(P),FSUB(R)(P)	r/m	1	4	1	FADD	
FIADD,FISUB(R)	m	2	4	1-2	FADD,FMISC	
FMUL(P)	r/m	1	4	1	FMUL	

FIMUL	m	2	4	2	FMUL,FMISC	
ED IV ((D) (D)	,		44.05	0.00	5.4. "	Low values are for
FDIV(R)(P)	r/m	1	11-25	8-22	FMUL	round divisors
FIDIV(R)	m	2	12-26	9-23	FMUL,FMISC	do.
FABS, FCHS		1	2	1	FMUL	
FCOM(P), FUCOM(P)	r/m	1	2	1	FADD	
FCOMPP, FUCOMPP		1	2	1	FADD	
FCOMI(P)	r	1	3	1	FADD	
FICOM(P)	m	2		1	FADD, FMISC	
FTST		1	2	1	FADD	
FXAM		2		1	FMISC, ALU	
FRNDINT		5	10	3		
FPREM		1	7-10	8	FMUL	
FPREM1		1	8-11	8	FMUL	
Math						
FSQRT		1	27	12	FMUL	
FLDPI, etc.		1		1	FMISC	
FSIN		66	140-190			
FCOS		73	150-190			
FSINCOS		98	170-200			
FPTAN		67	150-180			
FPATAN		97	217			
FSCALE		5	8			
FXTRACT		7	12	7		
F2XM1		53	126			
FYL2X		72	179			
FYL2XP1		75	175			
Other						
FNOP		1	0	1/3	FANY	
(F)WAIT		1	0	1/3	ALU	
FNCLEX		8		27	FMISC	
FNINIT		26		100	FMISC	
FNSAVE		77		171		
FRSTOR		70		136		
FXSAVE		61		56		
FXRSTOR		101		95		

Integer MMX and XMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions	·					
MOVD	r32, mm	2	4	2	FMICS, ALU	
MOVD	mm, r32	2	9	2	FANY, ALU	
MOVD	mm,m32	1		1/2	FANY	
MOVD	r32, xmm	3	2	2	FMISC, ALU	
MOVD	xmm, r32	3	3	2		
MOVD	xmm,m32	2		1	FANY	
MOVD	m32, r	1		1	FMISC	
						Moves 64 bits.Name
MOVD (MOVQ)	r64,mm/xmm	2	4	2	FMISC, ALU	of instruction differs

				-		
MOVD (MOVQ)	mm,r64	2	9	2	FANY, ALU	do.
MOVD (MOVQ)	xmm,r64	3	9	2	FANY, ALU	do.
MOVQ	mm,mm	1	2	1/2	FA/M	
MOVQ	xmm,xmm	2	2	1	FA/M, FMISC	
MOVQ	mm,m64	1		1/2	FANY	
MOVQ	xmm,m64	2		1	FANY, FMISC	
MOVQ	m64,mm/x	1		1	FMISC	
MOVDQA	xmm,xmm	2	2	1	FA/M	
MOVDQA	xmm,m	2	_	2	FMISC	
MOVDQA	m,xmm	2		2	FMISC	
MOVDQU	xmm,m	4		2	1 111100	
MOVDQU	m,xmm	5		2		
MOVDQ2Q	mm,xmm	1	2	1/2	FA/M	
MOVQ2DQ	xmm,mm	2	2	1	FA/M, FMISC	
MOVNTQ	· ·	1		2	FMISC	
MOVNTDQ	m,mm	2		3	FMISC	
	m,xmm	2		3	FIVIISC	
PACKSSWB/DW PACKUSWB	mm,r/m	1	2	2	FA/M	
PACKSSWB/DW	111111,1/111	'		2	FAVIVI	
PACKUSWB	xmm,r/m	3	3	2	FA/M	
PUNPCKH/LBW/WD/	AIIIII,1/111	0		_	1 / VIVI	
DQ	mm,r/m	1	2	2	FA/M	
PUNPCKH/LBW/WD/	111111,17111	•	_	_	170101	
DQ	xmm,r/m	2	2	2	FA/M	
PUNPCKHQDQ	xmm,r/m	2	2	1	FA/M	
PUNPCKLQDQ	xmm,r/m	1	2	1/2	FA/M	
PSHUFD	xmm,xmm,i	3	3	1.5	FA/M	
PSHUFW	mm,mm,i	1	2	1/2	FA/M	
PSHUFL/HW	xmm,xmm,i	2	2	1	FA/M	
MASKMOVQ	mm,mm	32		13	I AVIVI	
MASKMOVDQU		64		26		
PMOVMSKB	xmm,xmm		2		FADD	
	r32,mm/xmm	1	2 5	1		
PEXTRW	r32,mm/x,i	2		2	FMISC, ALU	
PINSRW	mm,r32,i	2	12	2	FA/M	
PINSRW	xmm,r32,i	3	12	3	FA/M	
Arithmetic instruction	S					
PADDB/W/D/Q PADDSB/W						
PADDUSB/W PSUBB/						
W/D/Q PSUBSB/W						
PSUBUSB/W	mm,r/m	1	2	1/2	FA/M	
PADDB/W/D/Q	, ,			-		
PADDSB/W ADDUSB/						
W PSUBB/W/D/Q						
PSUBSB/W						
PSUBUSB/W	xmm,r/m	2	2	1	FA/M	
PCMPEQ/GT B/W/D	mm,r/m	1	2	1/2	FA/M	
PCMPEQ/GT B/W/D	xmm,r/m	2	2	1	FA/M	
PMULLW PMULHW						
PMULHUW	,		_		F. 41 11	
PMULUDQ	mm,r/m	1	3	1	FMUL	
PMULLW PMULHW						
PMULHUW	vmm r/m	2	3	2		
PMULUDQ	xmm,r/m	2	၂		FMUL	

PMADDWD	mm,r/m	1	3	1	FMUL	
PMADDWD	xmm,r/m	2	3	2	FMUL	
PAVGB/W	mm,r/m	1	2	1/2	FA/M	
PAVGB/W	xmm,r/m	2	2	1	FA/M	
PMIN/MAX SW/UB	mm,r/m	1	2	1/2	FA/M	
PMIN/MAX SW/UB	xmm,r/m	2	2	1	FA/M	
PSADBW	mm,r/m	1	3	1	FADD	
PSADBW	xmm,r/m	2	3	2	FADD	
Logic						
PAND PANDN POR						
PXOR	mm,r/m	1	2	1/2	FA/M	
PAND PANDN POR	_	_	_			
PXOR	xmm,r/m	2	2	1	FA/M	
PSLL/RL W/D/Q		4		4.00	E A /B 4	
PSRAW/D	mm,i/mm/m	1	2	1/2	FA/M	
PSLL/RL W/D/Q PSRAW/D	x,i/x/m	2	2	1	FA/M	
PSLLDQ, PSRLDQ	·	2	2	1	FA/M	
F SLLDQ, FSKLDQ	xmm,i	2		ı		
Other						
EMMS		1		1/3	FANY	

Floating point XMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions				tinougnput	unit	
MOVAPS/D	r,r	2	2	1	FA/M	
MOVAPS/D	r,m	2		2	FMISC	
MOVAPS/D	m,r	2		2	FMISC	
MOVUPS/D	r,r	2	2	1	FA/M	
MOVUPS/D	r,m	4		2		
MOVUPS/D	m,r	5		2		
MOVSS/D	r,r	1	2	1	FA/M	
MOVSS/D	r,m	2	4	1	FANY FMISC	
MOVSS/D	m,r	1	3	1	FMISC	
MOVHLPS,						
MOVLHPS	r,r	1	2	1/2	FA/M	
MOVHPS/D,						
MOVLPS/D	r,m	1		1	FMISC	
MOVHPS/D,		1		4	ENTION	
MOVLPS/D	m,r	1		1	FMISC	0050
MOVDDUP	r,r	2	2 2	1		SSE3
MOVSH/LDUP	r,r	2	2	2	EN 4100	SSE3
MOVNTPS/D	m,r	2		3	FMISC	
MOVMSKPS/D	r32,r	1	8	1	FADD	
SHUFPS/D	r,r/m,i	3	3	2	FMUL	
UNPCK H/L PS/D	r,r/m	2	3	3	FMUL	
Conversion						
CVTPS2PD	r,r/m	2	4	2	FMISC	
CVTPD2PS	r,r/m	4	8	3	FMISC	

CVTSD2SS CVTSS2SD CVTDQ2PS CVTDQ2PD CVT(T)PS2DQ CVT(T)PD2DQ CVTPI2PS CVTPI2PD CVT(T)PS2PI CVT(T)PD2PI CVT(T)PD2PI CVTSI2SS	r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m xmm,mm xmm,mm mm,xmm mm,xmm	3 1 2 2 2 4 1 2 1 3 3	8 2 5 5 5 8 4 5 6 8 14	8 1 2 2 2 3 1 2 1 2	FMISC FMISC FMISC FMISC FMISC FMISC FMISC FMISC FMISC FMISC	
CVTSI2SD CVT(T)SD2SI	xmm,r32 r32,xmm	2	12 10	2 2	FMISC FMISC	
CVT(T)SS2SI	r32,xmm	2	9	2	FMISC	
Arithmetic ADDSS/D SUBSS/D	r,r/m	1	4	1	FADD	
ADDPS/D SUBPS/D	r,r/m	2	4	2	FADD	
HADDPS/D HSUBPS/	, ,					
D	r,r/m	2	4	2	FADD	SSE3
MULSS/D	r,r/m	1 2	4	1 2	FMUL	
MULPS/D	r,r/m	2	4	2	FMUL	Low values are for
						round divisors, e.g.
DIVSS	r,r/m	1	11-16	8-13	FMUL	powers of 2.
DIVPS	r,r/m	2	18-30	18-30	FMUL	do.
DIVSD	r,r/m	1	11-20	8-17	FMUL	do.
DIVPD	r,r/m	2	16-34	16-34	FMUL	do.
RCPSS	r,r/m	1	3	1	FMUL	
RCPPS	r,r/m	2	3	2	FMUL	
MAXSS/D MINSS/D	r,r/m	1	2	1	FADD	
MAXPS/D MINPS/D	r,r/m	2	2	2	FADD	
CMPccSS/D	r,r/m	1	2	1	FADD	
CMPccPS/D	r,r/m	2	2	2	FADD	
COMISS/D UCOMISS/	w w/saa	4	_	4	EADD	
D	r,r/m	1	2	1	FADD	
Logic						
ANDPS/D ANDNPS/D						
ORPS/D XORPS/D	r,r/m	2	2	2	FMUL	
Math						
SQRTSS	r,r/m	1	19	16	FMUL	
SQRTPS	r,r/m	2	36	36	FMUL	
SQRTSD	r,r/m	1	27	24	FMUL	
SQRTPD	r,r/m	2	48	48	FMUL	
RSQRTSS	r,r/m	1	3	1	FMUL	
RSQRTPS	r,r/m	2	3	2	FMUL	
	,,,,,,,	_		_		
Other						
LDMXCSR	m	8		9		
STMXCSR	m	3		10		

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB,

JNE, etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, xmm = 128 bit xmm register, sr = segment register, m = any memory operand including indirect operands, m64 means 64-bit memory operand, etc.

Ops: Number of macro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 macro-operations use microcode.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latency listed does not include the memory operand where the oper-

and is listed as register or memory (r/m).

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However,

the throughput may be limited by other bottlenecks in the pipeline.

Execution unit: Indicates which execution unit is used for the macro-operations. ALU means any

of the three integer ALU's. ALU0_1 means that ALU0 and ALU1 are both used. AGU means any of the three integer address generation units. FADD means floating point adder unit. FMUL means floating point multiplier unit. FMISC means floating point store and miscellaneous unit. FA/M means FADD or FMUL is used. FANY means any of the three floating point units can be used. Two macro-opera-

tions can execute simultaneously if they go to different execution units.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
mstruction	Operanus	Ops	Latericy	throughput		Notes
Move instructions				tinougnput	ume	
MOV	r,r	1	1	1/3	ALU	
MOV	r,i	1	1	1/3	ALU	
MOV	r8,m8	1	4	1/2	ALU, AGU	Any addr. mode. Add
MOV	r16,m16	1	4	1/2	ALU, AGU	1 clock if code seg-
MOV	r32,m32	1	3	1/2	AGU	ment base ≠ 0
MOV	r64,m64	1	3	1/2	AGU	
MOV	m8,r8H	1	8	1/2	AGU	AH, BH, CH, DH
MOV	m8,r8L	1	3	1/2	AGU	Any other 8-bit reg.
MOV	m16/32/64,r	1	3	1/2	AGU	Any addressing mode
MOV	m,i	1	3	1/2	AGU	
MOV	m64,i32	1	3	1/2	AGU	
MOV	r,sr	1	3-4	1/2		
MOV	sr,r/m	6	8-26	8		from AMD manual
MOVNTI	m,r	1		1	AGU	
MOVZX, MOVSX	r,r	1	1	1/3	ALU	
MOVZX, MOVSX	r,m	1	4	1/2	ALU, AGU	
MOVSXD	r64,r32	1	1	1/3	ALU	

MOVSXD	r64,m32	1	4	1/2	ALU, AGU	
CMOVcc	r,r	1	1	1/3	ALU	
CMOVcc	r,m	1	4	1/2	ALU, AGU	
XCHG	r,r	2	1	1	ALU	
XCHG	r,m	2	21	19	ALU, AGU	Timing depends on hw
XLAT	,	2	5	5	ALU, AGU	
PUSH	r	1	•	1/2	ALU, AGU	
PUSH	i i	1		1/2	ALU, AGU	
PUSH	m	2		1/2	ALU, AGU	
PUSH	sr	2		1	ALU, AGU	
PUSHF(D/Q)		9	_	3	ALU, AGU	
PUSHA(D)		9	6	6	ALU, AGU	
POP	r	1		1/2	ALU, AGU	
POP	m	3	3	1	ALU, AGU	
POP	DS/ES/FS/GS	6	10	8	ALU, AGU	
POP	SS	10	26	16	ALU, AGU	
POPF(D/Q)		28	16	11	ALU, AGU	
POPA(D)		9	6	6	ALU, AGU	
LEA	r16,[m]	2	3	1	ALU, AGU	Any address size
LEA	r32/64,[m]	1	1	1/3	ALU	≤ 2 source operands
LEA	r32/64,[m]		2	1/3	AGU	W. scale or 3 opr.
LAHF	132/04,[111]	4	3	2		vv. scale of 3 opt.
					ALU	
SAHF		1	1	1/3	ALU	
SALC		1	1	1	ALU	
LDS, LES,	r,m	10		10		
BSWAP	r	1	1	1/3	ALU	
PREFETCHNTA	m	1		1/2	AGU	
PREFETCHT0/1/2	m	1		1/2	AGU	
PREFETCH(W)	m	1		1/2	AGU	3DNow
SFENCE		6		8		
LFENCE		1		1		
MFENCE		4		33		
IN	r,i/DX	~270				
OUT	i/DX,r	~300				
001	1/07,1	300				
Arithmetic instruction	 S					
ADD, SUB	r,r/i	1	1	1/3	ALU	
ADD, SUB	r,m	1		1/2	ALU, AGU	
ADD, SUB	m,r	1	4	1	ALU, AGU	
ADC, SBB	r,r/i	1 1	1	1/3	ALU	
ADC, SBB	r,m	1	'	1/2	ALU, AGU	
ADC, SBB	· ·	1	4	1/2	ALU, AGU	
The state of the s	m,r/i	-				
CMP	r,r/i	1	1	1/3	ALU	
CMP	r,m	1		1/2	ALU, AGU	
INC, DEC, NEG	r	1	1	1/3	ALU	
INC, DEC, NEG	m	1	7	2	ALU, AGU	
AAA, AAS		9	5	5	ALU	
DAA		12	6	6	ALU	
DAS		16	7	7	ALU	
AAD		4	5	5	ALU0	
AAM		30	13	13	ALU	
MUL, IMUL	r8/m8	1	3	1	ALU0	
MUL, IMUL	r16/m16	3	3	2	ALU0_1	latency ax=3, dx=4
, ·····	1	1	•	_		

MUL, IMUL	r32/m32	2	3	1	ALU0_1	
MUL, IMUL	r64/m64	2	4	1 2	ALU0_1 ALU0_1	latency rax=4, rdx=5
IMUL	r16,r16/m16	1	3	1	ALUU_1 ALU0	laterity rax-4, rux-5
IMUL	r32,r32/m32	1	3	1	ALU0	
IMUL	· ·	-	4	2		
	r64,r64/m64	1 2	4		ALU0_1	
IMUL	r16,(r16),i			1	ALU0	
IMUL	r32,(r32),i	1	3	1	ALU0	
IMUL	r64,(r64),i	1	4	2	ALU0	
IMUL	r16,m16,i	3		2	ALU0	
IMUL	r32,m32,i	3		2	ALU0	
IMUL	r64,m64,i	3		2	ALU0_1	
DIV	r8/m8		17	17	ALU	
IDIV	r8		19	19	ALU	
IDIV	m8		22	22	ALU	
DIV	r16/m16		15-30	15-30	ALU	Depends on number
DIV	r32/m32		15-46	15-46	ALU	of significant bits in
DIV	r64/m64		15-78	15-78	ALU	absolute value of dividend. See AMD soft-
IDIV	r16/m16		24-39	24-39	ALU	ware optimization
IDIV	r32/m32		24-55	24-55	ALU	guide.
IDIV	r64/m64		24-87	24-87	ALU	guido.
CBW, CWDE, CDQE		1	1	1/3	ALU	
CWD, CDQ, CQO		1	1	1/3	ALU	
Logic instructions						
AND, OR, XOR	r,r	1	1	1/3	ALU	
AND, OR, XOR	r,m	1		1/2	ALU, AGU	
AND, OR, XOR	m,r	1	4	1	ALU, AGU	
TEST	r,r	1	1	1/3	ALU	
TEST	r,m	1		1/2	ALU, AGU	
NOT	r	1	1	1/3	ALU	
NOT	m	1	7	1	ALU, AGU	
SHL, SHR, SAR	r,i/CL	1	1	1/3	ALU	
ROL, ROR	r,i/CL	1	1	1/3	ALU	
RCL, RCR	r,1	1	1	1	ALU	
RCL	r,i	9	3	3	ALU	
RCR	r,i	7	3	3	ALU	
RCL	r,CL	9	4	4	ALU	
RCR	r,CL	7	3	3	ALU	
SHL,SHR,SAR,ROL,RO		1	7	1	ALU, AGU	
RCL, RCR	m,1	1	7	1	ALU, AGU	
RCL	m,i	10	7	5	ALU, AGU	
RCR	m,i	9	7	6	ALU, AGU	
RCL	m,CL	9	8	6	ALU, AGU	
RCR	m,CL	8	7	5	ALU, AGU	
SHLD, SHRD	r,r,i	6	3	2	ALU	
SHLD, SHRD	r,r,cl	7	3	3	ALU	
SHLD, SHRD	m,r,i/CL	8	7.5	6	ALU, AGU	
BT	r,r/i	1	1	1/3	ALU	
BT	m,i	1		1/2	ALU, AGU	
BT	m,r	5	7	2	ALU, AGU	
BTC, BTR, BTS	r,r/i	2	2	1/3	ALU	
BTC	m,i	5	9	1.5	ALU, AGU	
BTR, BTS	m,i	4	9	1.5	ALU, AGU	
5111, 515	111,1	-т	, ,	1.0	/LO, /\OU	1

BTC	m,r	8	8	10	ALU, AGU	
BTR, BTS	m,r	8	8	7	ALU, AGU	
BSF		6	4	3	ALU	
BSR	r,r			3		
	r,r	7	4		ALU	
BSF	r,m	7	7	3	ALU, AGU	
BSR	r,m	8	7	3	ALU, AGU	
POPCNT	r,r/m	1	2	1	ALU	SSE4.A / SSE4.2
LZCNT	r,r/m	1	2	1	ALU	SSE4.A, AMD only
SETcc	r	1	1	1/3	ALU	
SETcc	m	1		1/2	ALU, AGU	
CLC, STC		1		1/3	ALU	
CMC		1	1	1/3	ALU	
			I			
CLD		1		1/3	ALU	
STD		2		2/3	ALU	
Control transfer instru	 uctions					
JMP	short/near	1		2	ALU	
JMP	far	16-20	23-32	_	/	low values = real mode
JMP		10-20	20-02	2	ALU	iow values – real mode
	r					
JMP	m(near)	1		2	ALU, AGU	
JMP	m(far)	17-21	25-33			low values = real mode
Jcc	short/near	1		1/3 - 2	ALU	recip. thrp.= 2 if jump
J(E/R)CXZ	short	2		2/3 - 2	ALU	recip. thrp.= 2 if jump
LOOP	short	7		3	ALU	
CALL	near	3	2	2	ALU	
CALL	far	16-22	23-32			low values = real mode
CALL	r	4	3	3	ALU	
CALL	m(near)	5	3	3	ALU, AGU	
CALL	m(far)	16-22	24-33		7120,7100	low values = real mode
RETN	iii(iai)	2		2	A111	low values – real mode
			3	3	ALU	
RETN	i	2	3	3	ALU	
RETF	_	15-23	24-35			low values = real mode
RETF	i	15-24	24-35			low values = real mode
IRET		32	81			real mode
INT	i	33	42			real mode
BOUND	m	6		2		values are for no jump
INTO		2		2		values are for no jump
String instructions	-		^	_		
LODS		4	2	2		
REP LODS		5	2	2		values are per count
STOS		4	2	2		
REP STOS		2	1	1		values are per count
MOVS		7	3	3		
REP MOVS		3	1	1		values are per count
SCAS		5	2	2		
REP SCAS		5	2	2		values are per count
CMPS		7	3	3		Description of the second
REP CMPS		3	1	1		values are per count
Other						
NOP (90)		1	0	1/3	ALU	
Long NOP (0F 1F)		1	0	1/3	ALU	

ENTER	i,0	12		12	
LEAVE	2		3		3 ops, 5 clk if 16 bit
CLI	8-9		5		
STI	16-17		27		
CPUID	22-50	47-164			
RDTSC	30		67		
RDPMC	13		5		

Floating point x87 instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions				ougput		
FLD	r	1	2	1/2	FA/M	
FLD	m32/64	1	4	1/2	FANY	
FLD	m80	7	13	4		
FBLD	m80	20	94	30		
FST(P)	r	1	2	1/2	FA/M	
FST(P)	m32/64	1	2	1	FMISC	
FSTP	m80	10	8	7		
FBSTP	m80	218	167	163		
FXCH	r	1	0	1/3		
FILD	m	1	6	1	FMISC	
FIST(P)	m	1	4	1	FMISC	
FLDZ, FLD1		1		1	FMISC	
						Low latency immedi-
FCMOVcc	st0,r	9			FMISC, FA/M	ately after FCOMI
FFREE	r	1		1/3	FANY	
FINCSTP, FDECSTP		1	0	1/3	FANY	
ENGTOW/				40	EN4100 ALLI	Low latency immediately
FNSTSW	AX	2		16	FMISC, ALU	after FCOM FTST
FSTSW	AX	3		14	FMISC, ALU	do.
FNSTSW	m16	2		9	FMISC, ALU	do.
FNSTCW	m16	3		2	FMISC, ALU	f t : f
FLDCW	m16	12		14	FMISC, ALU	faster if unchanged
Arithmetic instruction	 S					
FADD(P),FSUB(R)(P)	r/m	1	4	1	FADD	
FIADD,FISUB(R)	m	2		4	FADD,FMISC	
FMUL(P)	r/m	1	4	1	FMUL	
FIMUL	m	2		4	FMUL,FMISC	
FDIV(R)(P)	r/m	1	?	24	FMUL	
FIDIV(R)	m	2	31	24	FMUL,FMISC	
FABS, FCHS		1	2	2	FMUL	
FCOM(P), FUCOM(P)	r/m	1		1	FADD	
FCOMPP, FUCOMPP		1		1	FADD	
FCOMI(P)	r	1		1	FADD	
FICOM(P)	m	2		1	FADD, FMISC	
FTST		1		1	FADD	
FXAM		2		1	FMISC, ALU	
FRNDINT		6		37		
FPREM		1		7	FMUL	
FPREM1		1		7	FMUL	

		1			
Math					
FSQRT		1	35	35	FMUL
FLDPI, etc.		1		1	FMISC
FSIN		45	~51?		
FCOS		51	~90?		
FSINCOS		76	~125?		
FPTAN		45	~119		
FPATAN		9	151?	45?	
FSCALE		5	9	29	
FXTRACT		11	9	41	
F2XM1		8	65	30?	
FYL2X		8	13	30?	
FYL2XP1		12	114	44?	
Other					
FNOP		1	0	1/3	FANY
(F)WAIT		1	0	1/3	ALU
FNCLEX		8		28	FMISC
FNINIT		26		103	FMISC
FNSAVE	m	77	162	149	
FRSTOR	m	70	133	149	
FXSAVE	m	61	63	58	
FXRSTOR	m	85	89	79	

Integer MMX and Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
	Operando	Оро	Lutonoy	throughput		110100
Move instructions						
MOVD	r32, mm	1	3	1	FADD	
MOVD	mm, r32	2	6	3		
MOVD	mm,m32	1	4	1/2	FANY	
MOVD	r32, xmm	1	3	1	FADD	
MOVD	xmm, r32	2	6	3		
MOVD	xmm,m32	1	2	1/2		
MOVD	m32,mm/x	1	2	1	FMISC	
						Moves 64 bits.Name
MOVD (MOVQ)	r64,mm/x	1	3	1	FADD	of instruction differs
MOVD (MOVQ)	mm,r64	2	6	3		do.
MOVD (MOVQ)	xmm,r64	2	6	3	FMUL, ALU	do.
MOVQ	mm,mm	1	2	1/2	FA/M	
MOVQ	xmm,xmm	1	2.5	1/3	FANY	
MOVQ	mm,m64	1	4	1/2	FANY	
MOVQ	xmm,m64	1	2	1/2	?	
MOVQ	m64,mm/x	1	2	1	FMISC	
MOVDQA	xmm,xmm	1	2.5	1/3	FANY	
MOVDQA	xmm,m	1	2	1/2	?	
MOVDQA	m,xmm	2	2	1	FMUL,FMISC	
MOVDQU	xmm,m	1	2	1/2		
MOVDQU	m,xmm	3	3	2		
MOVDQ2Q	mm,xmm	1	2	1/3	FANY	
MOVQ2DQ	xmm,mm	1	2	1/3	FANY	

MOVNTQ	m,mm	1		1	FMISC	
MOVNTDQ	m,xmm	2		1	FMUL,FMISC	
PACKSSWB/DW	111,7111111	_		•	I WOL,I WIGO	
PACKUSWB	mm,r/m	1	2	1/2	FA/M	
PACKSSWB/DW	,,,,,	•	_	.,_	', ', ', ', '	
PACKUSWB	xmm,r/m	1	3	1/2	FA/M	
PUNPCKH/LBW/WD/	7,.,	-		.,_		
DQ	mm,r/m	1	2	1/2	FA/M	
PUNPCKH/LBW/WD/	,,,,,,,,		_			
DQ	xmm,r/m	1	3	1/2	FA/M	
PUNPCKHQDQ	xmm,r/m	1	3	1/2	FA/M	
PUNPCKLQDQ	xmm,r/m	1	3	1/2	FA/M	
PSHUFD	xmm,xmm,i	1	3	1/2	FA/M	
PSHUFW	mm,mm,i	1	2	1/2	FA/M	
PSHUFL/HW	xmm,xmm,i	1	2	1/2	FA/M	
MASKMOVQ	mm,mm	32		13	I AVIVI	
MASKMOVDQU	xmm,xmm	64		24		
PMOVMSKB	r32,mm/xmm	1	3	1	FADD	
	I '				FADD	
PEXTRW	r32,mm/x,i	2	6	1	Ε Δ / Δ A	
PINSRW	mm/x,r32,i	2	9	3	FA/M	0054.4. AMD
INSERTQ	xmm,xmm	3	6	2	FA/M	SSE4.A, AMD only
INSERTQ	xmm,xmm,i,i	3	6	2	FA/M	SSE4.A, AMD only
EXTRQ	xmm,xmm	1	2	1/2	FA/M	SSE4.A, AMD only
EXTRQ	xmm,xmm,i,i	1	2	1/2	FA/M	SSE4.A, AMD only
Arithmetic instruction	S					
PADDB/W/D/Q PADDSB/W						
PADDSB/W PADDUSB/W PSUBB/						
W/D/Q PSUBSB/W						
PSUBUSB/W	mm/xmm,r/m	1	2	1/2	FA/M	
PCMPEQ/GT B/W/D	mm/xmm,r/m	1	2	1/2	FA/M	
PMULLW PMULHW	,,,,,,,,	-	_			
PMULHUW						
PMULUDQ	mm/xmm,r/m	1	3	1	FMUL	
PMADDWD	mm/xmm,r/m	1	3	1	FMUL	
PAVGB/W	mm/xmm,r/m	1	2	1/2	FA/M	
PMIN/MAX SW/UB	mm/xmm,r/m	1	2	1/2	FA/M	
PSADBW	mm/xmm,r/m	1	3	1	FADD	
	,					
Logic						
PAND PANDN POR						
PXOR	mm/xmm,r/m	1	2	1/2	FA/M	
PSLL/RL W/D/Q						
PSRAW/D	mm,i/mm/m	1	2	1/2	FA/M	
PSLL/RL W/D/Q						
PSRAW/D	x,i/mm/x	1	3	1/2	FA/M	
PSLLDQ, PSRLDQ	xmm,i	1	3	1/2	FA/M	
Other		4		4.10	FAND	
EMMS		1		1/3	FANY	

Floating point XMM instructions

Move instructions MOVAPS/D MOVAPS/D MOVAPS/D MOVUPS/D MOVUPS/D MOVUPS/D MOVSS/D MOVSS/D MOVSS/D MOVHLPS, MOVHPS/D,	r,r r,m m,r r,r r,m m,r r,r r,m	Ops 1 1 2 1 1 3 1 1	2.5 2 2 2.5 2 3	Reciprocal throughput 1/2 1/2 1 1/2 1/2 1/2 1/2	FANY ? FMUL,FMISC FANY	Notes
MOVAPS/D MOVAPS/D MOVAPS/D MOVUPS/D MOVUPS/D MOVUPS/D MOVSS/D MOVSS/D MOVSS/D MOVHLPS, MOVHLPS, MOVHPS/D,	r,m m,r r,r r,m m,r r,r r,m	1 2 1 1 3 1	2 2 2.5 2 3	1/2 1 1/2	? FMUL,FMISC	
MOVAPS/D MOVAPS/D MOVUPS/D MOVUPS/D MOVUPS/D MOVSS/D MOVSS/D MOVSS/D MOVHLPS, MOVHLPS, MOVHPS/D,	r,m m,r r,r r,m m,r r,r r,m	1 2 1 1 3 1	2 2 2.5 2 3	1/2 1 1/2	? FMUL,FMISC	
MOVAPS/D MOVUPS/D MOVUPS/D MOVUPS/D MOVSS/D MOVSS/D MOVSS/D MOVSS/D MOVHLPS, MOVHLPS, MOVHPS/D,	m,r r,r r,m m,r r,r r,m	2 1 1 3 1	2 2.5 2 3	1 1/2	FMUL,FMISC	
MOVUPS/D MOVUPS/D MOVUPS/D MOVSS/D MOVSS/D MOVSS/D MOVHLPS, MOVLHPS MOVHPS/D,	r,r r,m m,r r,r r,m	1 1 3 1	2.5 2 3	1/2	· ·	
MOVUPS/D MOVUPS/D MOVSS/D MOVSS/D MOVSS/D MOVHLPS, MOVHLPS MOVHPS/D,	r,m m,r r,r r,m	1 3 1	2 3		FANY	
MOVUPS/D MOVSS/D MOVSS/D MOVSS/D MOVHLPS, MOVHLPS MOVHPS/D,	m,r r,r r,m	3 1	3	1/2		
MOVSS/D MOVSS/D MOVSS/D MOVHLPS, MOVLHPS MOVHPS/D,	r,r r,m	1			?	
MOVSS/D MOVSS/D MOVHLPS, MOVLHPS MOVHPS/D,	r,m	-	1 -	2	FMISC	
MOVSS/D MOVHLPS, MOVLHPS MOVHPS/D,		1	2	1/2	FA/M	
MOVHLPS, MOVLHPS MOVHPS/D,	m,r		2	1/2	?	
MOVLHPS MOVHPS/D,		1	2	1	FMISC	
MOVHPS/D,						
	r,r	1	3	1/2	FA/M	
MOVLPS/D	r,m	1	4	1/2	FA/M	
MOVHPS/D,						
MOVLPS/D	m,r	1		1	FMISC	
MOVNTPS/D	m,r	2		3	FMUL,FMISC	
MOVNTSS/D	m,r	1		1	FMISC	SSE4.A, AMD only
MOVMSKPS/D	r32,r	1	3	1	FADD	
SHUFPS/D	r,r/m,i	1	3	1/2	FA/M	
UNPCK H/L PS/D	r,r/m	1	3	1/2	FA/M	
Conversion						
CVTPS2PD	r,r/m	1	2	1	FMISC	
CVTPD2PS	r,r/m	2	7	1		
CVTSD2SS	r,r/m	3	8	2		
CVTSS2SD	r,r/m	3	7	2		
CVTDQ2PS	r,r/m	1	4	1	FMISC	
CVTDQ2PD	r,r/m	1	4	1	FMISC	
CVT(T)PS2DQ	r,r/m	1	4	1	FMISC	
CVT(T)PD2DQ	r,r/m	2	7	1		
CVTPI2PS	xmm,mm	2	7	1		
CVTPI2PD	xmm,mm	1	4	1	FMISC	
CVT(T)PS2PI	mm,xmm	1	4	1	FMISC	
CVT(T)PD2PI	mm,xmm	2	7	1		
CVTSI2SS	xmm,r32	3	14	3		
CVTSI2SD	xmm,r32	3	14	3		
CVT(T)SD2SI	r32,xmm	2	8	1	FADD,FMISC	
CVT(T)SS2SI	r32,xmm	2	8	1	FADD,FMISC	
Arithmetic						
ADDSS/D SUBSS/D	r,r/m	1	4	1	FADD	
ADDPS/D SUBPS/D	r,r/m	1	4	1	FADD	
MULSS/D	r,r/m	1	4	1	FMUL	
MULPS/D	r,r/m	1	4	1	FMUL	
DIVSS	r,r/m	1	16	13	FMUL	
DIVPS	r,r/m	1	18	15	FMUL	
DIVSD	r,r/m	1	20	17	FMUL	
DIVPD	r,r/m	1	20	17	FMUL	
RCPSS RCPPS	r,r/m	1	3	1	FMUL	

MAXSS/D MINSS/D MAXPS/D MINPS/D	r,r/m r,r/m	1	2 2	1	FADD FADD	
CMPccSS/D	r,r/m	1	2	1	FADD	
CMPccPS/D	r,r/m	1	2	1	FADD	
COMISS/D UCOMISS/	1,1/111		_	'	INDD	
D D	r,r/m	1		1	FADD	
Logic						
ANDPS/D ANDNPS/D	_					
ORPS/D XORPS/D	r,r/m	1	2	1/2	FA/M	
Math						
SQRTSS	r,r/m	1	19	16	FMUL	
SQRTPS	r,r/m	1	21	18	FMUL	
SQRTSD	r,r/m	1	27	24	FMUL	
SQRTPD	r,r/m	1	27	24	FMUL	
RSQRTSS	r,r/m	1	3	1	FMUL	
RSQRTPS	r,r/m	1	3	1	FMUL	
Other						
LDMXCSR	m	12	12	10		
STMXCSR	m	3	12	11		

Obsolete 3DNow instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution	Notes
Move and convert ins	tructions			tiirougiiput	unit	
PF2ID	mm,mm	1	5	1	FMISC	
PI2FD	mm,mm	1	5	1	FMISC	
PF2IW	mm,mm	1	5	1	FMISC	3DNow extension
PI2FW	mm,mm	'1	5	1	FMISC	3DNow extension
PSWAPD	1	1	2	1/2	FA/M	3DNow extension
PSWAPD	mm,mm	1		1/2	FAVIVI	SDINOW extension
Integer instructions						
PAVGUSB	mm,mm	1	2	1/2	FA/M	
PMULHRW	mm,mm	1	3	1	FMUL	
	,					
Floating point instruct	tions					
PFADD/SUB/SUBR	mm,mm	1	4	1	FADD	
PFCMPEQ/GE/GT	mm,mm	1	2	1	FADD	
PFMAX/MIN	mm,mm	1	2	1	FADD	
PFMUL	mm,mm	1	4	1	FMUL	
PFACC	mm,mm	1	4	1	FADD	
PFNACC, PFPNACC	mm,mm	1	4	1	FADD	3DNow extension
PFRCP	mm,mm	1	3	1	FMUL	
PFRCPIT1/2	mm,mm	1	4	1	FMUL	
PFRSQRT	mm,mm	1	3	1	FMUL	
PFRSQIT1	mm,mm	1	4	1	FMUL	
	,					
Other						
FEMMS	mm,mm	1		1/3	FANY	

Thank you to Xucheng Tang for doing the measurements on the K10.

AMD Bulldozer

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB, JNE,

etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, x = 128 bit xmm register, y = 256 bit ymm register, m = any memory operand including indirect operands, m64 means 64-bit memory operand, etc.

Ops: Number of macro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 macro-operations use microcode.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latency listed does not include the memory operand where the listing

for register and memory operand are joined (r/m).

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However, the throughput may be limited by other bottlenecks in the pipeline.

Execution pipe: Indicates which execution pipe or unit is used for the macro-operations:

Integer pipes:

EX0: integer ALU, division

EX1: integer ALU, multiplication, jump EX01: can use either EX0 or EX1 AG01: address generation unit 0 or 1 Floating point and vector pipes:

P0: floating point add, mul, div, convert, shuffle, shift

P1: floating point add, mul, div, shuffle, shift

P2: move, integer add, boolean P3: move, integer add, boolean, store

P01: can use either P0 or P1 P23: can use either P2 or P3

Two macro-operations can execute simultaneously if they go to different

execution pipes

Domain: Tells which execution unit domain is used:

ivec: integer vector execution unit. fp: floating point execution unit. fma: floating point multiply/add subunit.

inherit: the output operand inherits the domain of the input operand.

ivec/fma means the input goes to the ivec domain and the output comes from the

fma domain.

There is an additional latency of 1 clock cycle if the output of an ivec instruction goes to the input of a fp or fma instruction, and when the output of a fp or fma instruction goes to the input of an ivec or store instruction. There is no latency between the fp and fma units. All other latencies after memory load and before mem-

ory store instructions are included in the latency counts.

An fma instruction has a latency of 5 if the output goes to another fma instruction, 6 if the output goes to an fp instruction, and 6+1 if the output goes to an ivec or

store instruction.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
		-		throughput	pipes	

Move instructions						
MOV	r,r	1	1	0.5	EX01	
MOV	r,i	1	1	0.5	EX01	
MOV	r,m	1	4	0.5	AG01	all addr. modes
MOV	m,r	1	4	1	EX01 AG01	all addr. modes
MOV	m,i	1		1		
MOVNTI	m,r	1	5	2		
MOVZX, MOVSX	r,r	1	1	0.5	EX01	
MOVSX	r,m	1	5	0.5	EX01	
MOVZX	r,m	1	4	0.5	EX01	
MOVSXD	r64,r32	1	1	0.5	EX01	
MOVSXD	r64,m32	1	5	0.5	EX01	
CMOVcc	r,r	1	1	0.5	EX01	
CMOVcc	r,m	1		0.5	EX01	
XCHG		2	1	1	EX01	
ACIIG	r,r		'	'	EXUI	Timing depends on
XCHG	r,m	2	~50	~50	EX01	Timing depends on hw
XLAT	1,	2	6	2	2701	1111
PUSH	r	1		1		
PUSH	i	1		1		
PUSH	m	2		1.5		
PUSHF(D/Q)	111	8		4		
` '		9		9		
PUSHA(D)						
POP	r	1		1		
POP	m	2		1		
POPF(D/Q)		34		19		
POPA(D)	40.5.1	14	0.0	8	E)/0.4	
LEA	r16,[m]	2	2-3		EX01	any addr. size
LEA	r32,[m]	2	2-3		EX01	16 bit addr. size
	-20/04 []	_	_	0.5	EV04	scale factor > 1
LEA	r32/64,[m]	1	2	0.5	EX01	or 3 operands
LEA	r32/64,[m]	1	1	0.5	EX01	all other cases
LAHF		4	3	2		
SAHF		2	2	1		
SALC		1	1	1	5)/0/	
BSWAP	r	1	1	0.5	EX01	
PREFETCHNTA	m	1		0.5		
PREFETCHT0/1/2	m	1		0.5		
PREFETCH/W	m	1		0.5		AMD 3DNow
SFENCE		6		89		
LFENCE		1		0.25		
MFENCE		6		89		
Arithmetic instructions	3					
ADD, SUB	r,r	1	1	0.5	EX01	
ADD, SUB	r,i	1	1	0.5	EX01	
ADD, SUB	r,m	1		0.5	EX01	
ADD, SUB	m,r	1	7-8	1	EX01	
ADD, SUB	m,i	1	7-8	1	EX01	
ADC, SBB	r,r	1	1		EX01	
ADC, SBB	r,i	1	1		EX01	
ADC, SBB	r,m	1	1	1	EX01	
ADC, SBB	m,r	1	9	1	EX01	

ADC, SBB	m,i	1	9	1	EX01
CMP	r,r	1	1	0.5	EX01
CMP	r,i	1	1	0.5	EX01
CMP	r,m	1		0.5	EX01
INC, DEC, NEG	r	1	1	0.5	EX01
INC, DEC, NEG	m	1	7-8	1	EX01
AAA, AAS		10	6	-	
DAA		16	9		
DAS		20	10		
AAD		4	6		
AAM		9	20	20	
MUL, IMUL	r8/m8	1	4	2	EX1
MUL, IMUL	r16/m16	2	4	2	EX1
MUL, IMUL	r32/m32	1	4	2	EX1
MUL, IMUL	r64/m64	1	6	4	EX1
IMUL	r16,r16/m16	1	4	2	EX1
IMUL	r32,r32/m32	1	4	2	EX1
IMUL		1	6	4	
	r64,r64/m64	2	5	2	EX1
IMUL	r16,(r16),i			2	EX1
IMUL	r32,(r32),i	1	4		EX1
IMUL	r64,(r64),i	1	6	4	EX1
IMUL	r16,m16,i	2		2	EX1
IMUL	r32,m32,i	2		2	EX1
IMUL	r64,m64,i	2		4	EX1
DIV	r8/m8	14	20	20	EX0
DIV	r16/m16	18	15-27	15-28	EX0
DIV	r32/m32	16	16-43	16-43	EX0
DIV	r64/m64	16	16-75	16-75	EX0
IDIV	r8/m8	33	23	20	EX0
IDIV	r16/m16	36	23-33	20-27	EX0
IDIV	r32/m32	36	22-48	20-43	EX0
IDIV	r64/m64	36	22-79	20-75	EX0
CBW, CWDE, CDQE		1	1		EX01
CDQ, CQO		1	1	0.5	EX01
CWD		2	1	1	EX01
Logic instructions					
AND, OR, XOR	r,r	1	1	0.5	EX01
AND, OR, XOR	r,i	1	1	0.5	EX01
AND, OR, XOR	r,m	1		0.5	EX01
AND, OR, XOR	m,r	1	7-8	1	EX01
AND, OR, XOR	m,i	1	7-8	1	EX01
TEST	r,r	1	1	0.5	EX01
TEST	r,i	1	1	0.5	EX01
TEST	m,r	1		0.5	EX01
TEST	m,i	1		0.5	EX01
NOT	r	1	1	0.5	EX01
NOT	m	1	7	1	EX01
SHL, SHR, SAR	r,i/CL	1	1	0.5	EX01
ROL, ROR	r,i/CL	1	1	0.5	EX01
RCL	r,1	1	1		EX01
RCL	r,i	16	8		EX01
RCL	r,cl	17	9		EX01
1 -	.,		-	I	

RCR	r,1	1	1		EX01	
RCR	r,i	15	8		EX01	
RCR	r,cl	16	8		EX01	
SHLD, SHRD		6	3	3	EX01	
	r,r,i					
SHLD, SHRD	r,r,cl	7	4	3.5	EX01	
SHLD, SHRD	m,r,i/CL	8		3.5	EX01	
BT	r,r/i	1	1	0.5	EX01	
BT	m,i	1		0.5	EX01	
BT	m,r	7		3.5	EX01	
BTC, BTR, BTS	r,r/i	2	2	1	EX01	
BTC, BTR, BTS	m,i	4		2	EX01	
BTC, BTR, BTS	m,r	10		5	EX01	
BSF		6	2	3	EX01	
1	r,r		3			
BSF	r,m	8	4	4	EX01	
BSR	r,r	7	4	4	EX01	
BSR	r,m	9		5	EX01	
LZCNT	r,r	1	2	2	EX0	SSE4.A
POPCNT	r,r/m	1	4	2	EX1	SSE4.2
SETcc	r	1	1	0.5	EX01	
SETcc	m	1		1	EX01	
CLC, STC		1		0.5	EX01	
CMC		1	1	0.5	EX01	
1			1	_		
CLD		2		3		
STD		2		4		
POPCNT	r16/32,r16/32	1	4	2		SSE4A
POPCNT	r64,r64	1	4	4		SSE4A
LZCNT	r,r	2	2	2		SSE4A
EXTRQ	x,i,i	1	3	1	P1	SSE4A
EXTRQ	x,x	1	3	1	P1	SSE4A
INSERTQ	x,x,i,i	1	3	1	P1	SSE4A
INSERTQ	X,X,1,1	1	3	1	P1	SSE4A
INSLITTE	^,^	ı	3	l l	Г	JOL4A
	1					
Control transfer instru				_		
JMP	short/near	1		2	EX1	
JMP	r	1		2	EX1	
JMP	m	1		2	EX1	
Jcc	short/near	1		1-2	EX1	2 if jumping
fused CMP+Jcc	short/near	1		1-2	EX1	2 if jumping
J(E/R)CXZ	short	1		1-2	EX1	2 if jumping
LOOP	short	1		1-2	EX1	2 if jumping
LOOPE LOOPNE	short	1		1-2	EX1	2 if jumping
1						Z II jumping
CALL	near	2		2	EX1	
CALL	r	2		2	EX1	
CALL	m	3		2	EX1	
RET		1		2	EX1	
RET	i	4		2-3	EX1	
BOUND	m	11		5		for no jump
INTO		4		24		for no jump
		•		-		,
String instructions						
LODS	1	3		3		
REP LODS		6n		3n		
STOS		3		3		

REP STOS		2n		2n		small n
REP STOS		3 per 16B		3 per 16B		best case
MOVS		5 pcr 10B		3		best ease
REP MOVS		2n		2n		small n
REP MOVS		4 per 16B		3 per 16B		best case
SCAS		3		3		bost dasc
REP SCAS		7n		4n		
CMPS		6		3		
REP CMPS		9n		4n		
TALL CIVIL C						
Synchronization						
LOCK ADD	m,r	1	~55			
XADD	m,r	4	10			
LOCK XADD	m,r	4	~51			
CMPXCHG	m8,r8	5	15			
LOCK CMPXCHG	m8,r8	5	~51			
CMPXCHG	m,r16/32/64	6	14			
LOCK CMPXCHG	m,r16/32/64	6	~52			
CMPXCHG8B	m64	18	15			
LOCK CMPXCHG8B	m64	18	~53			
CMPXCHG16B	m128	22	52			
LOCK CMPXCHG16B	m128	22	~94			
Other						
NOP (90)		1		0.25	none	
Long NOP (0F 1F)		1		0.25	none	
PAUSE		40		43		
ENTER	a,0	13		22		
ENTER	a,b	11+5b		16+4b		
LEAVE		2		4		
CPUID		37-63		112-280		
RDTSC		36		42		
RDPMC	*20 -0	22	•	300		
CRC32	r32,r8	3	3	2		
CRC32	r32,r16	5	5	5		
CRC32	r32,r32	5	6	6		
XGETBV		4		31		

Floating point x87 instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes		
Move instructions								
FLD	r	1	2	0.5	P01	fp		
FLD	m32/64	1	8	1		fp		
FLD	m80	8	14	4		fp		
FBLD	m80	60	61	40	P0 P1 P2 P3	fp		
FST(P)	r	1	2	0.5	P01	fp		
FST(P)	m32/64	2	8	1		fp		
FSTP	m80	13	9	20		fp		
FBSTP	m80	239	240	244	P0 P1 F3	fp		
FXCH	r	1	0	0.5	P01	inherit		
FILD	m	1	12	1	F3	fp		

FIST(P)	m	2	8	1	P0 F3	fp
FLDZ, FLD1		1		0.5	P01	fp
FCMOVcc	st0,r	8	3	3	P0 P1 F3	fp
FFREE	r	1		0.25	none	
FINCSTP, FDECSTP		1	0	0.25	none	inherit
FNSTSW	AX	4	~13	22	P0 P2 P3	
FNSTSW	m16	3	~13	19	P0 P2 P3	
FLDCW	m16	1		3		
FNSTCW	m16	3		2		
Arithmetic instructions	5					
FADD(P),FSUB(R)(P)	r/m	1	5-6	1	P01	fma
FIADD,FISUB(R)	m	2		2	P01	fma
FMUL(P)	r/m	1	5-6	1	P01	fma
FIMUL	m	2		2	P01	fma
FDIV(R)(P)	r	1	10-42	5-18	P01	fp
FDIV(R)	m	2			P01	fp
FIDIV(R)	m	2			P01	fp
FABS, FCHS		1	2	0.5	P01	fp
FCOM(P), FUCOM(P)	r/m	1		0.5	P01	fp
FCOMPP, FUCOMPP	.,	1		0.5	P01	fp
FCOMI(P)	r	2	2	1	P0 P1 F3	fp
FICOM(P)	m	2	_	1	P01	fp
FTST		1		0.5	P01	fp
FXAM		1	~20	0.5	P01	fp
FRNDINT		1	4	1	P0	fp
FPREM		1	19-62	'	P0	fp
FPREM1		1	19-62		P0	fp
FFICIVII		ı	19-03		FU	ıρ
Math						
FSQRT		1	10-53		P01	
FLDPI, etc.		1		0.5	P01	
FSIN		10-162	65-210	65-210	P0 P1 P3	
FCOS		160-170	~160	~160	P0 P1 P3	
FSINCOS		12-166	95-160	95-160	P0 P1 P3	
FPTAN		11-190	95-245	95-245	P0 P1 P3	
FPATAN		10-355	60-440	60-440	P0 P1 P3	
FSCALE		8	52	23 . 10	P0 P1 P3	
FXTRACT		12	10	5	P0 P1 P3	
F2XM1		10	64-71	J	P0 P1 P3	
FYL2X		10-175	0		P0 P1 P3	
FYL2XP1		10-175			P0 P1 P3	
1 1 2 2 7 11 1		10 170			101110	
Other						
FNOP		1		0.25	none	
(F)WAIT		1		0.25	none	
FNCLEX		18		57	P0	
FNINIT		31		170	P0	
FNSAVE	m864	103	300	300	P0 P1 P2 P3	
FRSTOR	m864	76	312	312	P0 P3	

Integer vector instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions					-	
MOVD	r32/64, mm/x	1	8	1		
MOVD	mm/x, r32/64	2	10	1		
MOVD	mm/x,m32	1	6	0.5		
MOVD	m32,mm/x	1	5	1		
MOVQ	mm/x,mm/x	1	2	0.5	P23	
MOVQ	mm/x,m64	1	6	0.5		
MOVQ	m64,mm/x	1	5	1	P3	
MOVDQA	xmm,xmm	1	0	0.25	none	inherit domain
MOVDQA	xmm,m	1	6	0.5		
MOVDQA	m,xmm	1	5	1	P3	
VMOVDQA	ymm,ymm	2	2	0.5	P23	
VMOVDQA	ymm,m256	2	6	1	1 20	
VMOVDQA	m256,ymm	4	5	3	P3	
MOVDQU	xmm,xmm	1	0	0.25	none	inherit domain
MOVDQU		1	6	0.25	none	minent domain
	xmm,m	•			Da	
MOVDQU	m,xmm	1	5	1	P3	
LDDQU	xmm,m	1	6	0.5		
VMOVDQU	ymm,m256	2	6	1-2	D0 D0	
VMOVDQU	m256,ymm	8	6	10	P2 P3	
MOVDQ2Q	mm,xmm	1	2	0.5	P23	
MOVQ2DQ	xmm,mm	1	2	0.5	P23	
MOVNTQ	m,mm	1	6	2	P3	
MOVNTDQ	m,xmm	1	6	2	P3	
MOVNTDQA	xmm,m	1	6	0.5		
PACKSSWB/DW	mm/x,r/m	1	2	1	P1	
PACKUSWB	mm/x,r/m	1	2	1	P1	
PUNPCKH/LBW/WD/						
DQ	mm/x,r/m	1	2	1	P1	
PUNPCKHQDQ	xmm,r/m	1	2	1	P1	
PUNPCKLQDQ	xmm,r/m	1	2	1	P1	
PSHUFB	mm/x,r/m	1	3	1	P1	
PSHUFD	xmm,xmm,i	1	2	1	P1	
PSHUFW	mm,mm,i	1	2	1	P1	
PSHUFL/HW	xmm,xmm,i	1	2	1	P1	
PALIGNR	mm/x,r/m,i	1	2	1	P1	
PBLENDW	xmm,r/m	1	2	0.5	P23	SSE4.1
MASKMOVQ	mm,mm	31	38	37	P3	
MASKMOVDQU	xmm,xmm	64	48	61	P1 P3	
PMOVMSKB	r32,mm/x	2	10	1	P1 P3	
PEXTRB/W/D/Q	r,x/mm,i	2	10	1	P1 P3	AVX
PINSRB/W/D/Q	x/mm,r,i	2	12	2	P1	
PMOVSXBW/BD/BQ/	^//////////////////////////////////////	4	14		ЕТ	
WD/WQ/DQ	xmm,xmm	1	2	1	P1	SSE4.1
PMOVZXBW/BD/BQ/	AIIIII,AIIIII	Ī	_	'		JOL7.1
WD/WQ/DQ	xmm,xmm	1	2	1	P1	SSE4.1
VPCMOV	x,x,x,x/m	1	2	1	P1	AMD XOP
VPCMOV	y,y,y,y/m	2	2	2	P1	AMD XOP
VPPERM	x,x,x,x/m	1	2	1	P1	AMD XOP
VIII LINIVI	۸,۸,۸,۸/۱۱۱	I		1	ГΙ	AIVID AUF

Arithmetic instructions	•					
PADDB/W/D/Q/SB/SW/	•					
USB/USW	mm/x,r/m	1	2	0.5	P23	
PSUBB/W/D/Q/SB/SW/	111111/2,1/111	'	2	0.5	1 23	
USB/USW	mm/x,r/m	1	2	0.5	P23	
PHADD/SUB(S)W/D	•	3	5	2	P1 P23	SSSE3
` ,	X,X	4		2		
PHADD/SUB(S)W/D	x,m		5		P1 P23	SSSE3
PCMPEQ/GT B/W/D	mm/x,r/m	1	2	0.5	P23	00544
PCMPEQQ	mm/x,r/m	1	2	0.5	P23	SSE4.1
PCMPGTQ	mm/x,r/m	1	2	0.5	P23	SSE4.2
PMULLW PMULHW				_		
PMULHUW PMULUDQ	mm/x,r/m	1	4	1	P0	
PMULLD	xmm,r/m	1	5	2	P0	SSE4.1
PMULDQ	xmm,r/m	1	4	1	P0	SSE4.1
PMULHRSW	mm/x,r/m	1	4	1	P0	SSSE3
PMADDWD	mm/x,r/m	1	4	1	P0	
PMADDUBSW	mm/x,r/m	1	4	1	P0	
PAVGB/W	mm/x,r/m	1	2	0.5	P23	
PMIN/MAX SB/SW/ SD						
UB/UW/UD	mm/x,r/m	1	2	0.5	P23	
PHMINPOSUW	xmm,r/m	2	4	1	P1 P23	SSE4.1
PABSB/W/D	mm/x,r/m	1	2	0.5	P23	SSSE3
PSIGNB/W/D	mm/x,r/m	1	2	0.5	P23	SSSE3
PSADBW	mm/x,r/m	2	4	1	P23	33323
MPSADBW	x,x,i	8	8	4	P1 P23	SSE4.1
WII SABBVV	۸,۸,۱		O	7	11123	AMD XOP
VPCOMB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
VI GOIVIB/VV/B/Q	λ,λ,λ/111,1	'	2	0.5	1 20	AMD XOP
VPCOMUB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
VPHADDBW/BD/BQ/	λ,λ,λ/111,1		_	0.0	1 20	latority of it is o, i
WD/WQ/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPHADDUBW/BD/BQ/	7,7,7,111		_	0.0	. 20	7 11112 7101
WD/WQ/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPHSUBBW/WD/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPMACSWW/WD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSDD	x,x,x/m,x	1	5	2	P0	AMD XOP
VPMACSDQH/L	x,x,x/m,x x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSSWW/WD	x,x,x/m,x x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSSWW/WB	x,x,x/m,x	1	5	2	P0	AMD XOP
VPMACSSDD VPMACSSDQH/L		1	4	1	P0	AMD XOP
VPMADCSWD	x,x,x/m,x	1 .			P0	
	x,x,x/m,x	1	4	1		AMD XOP
VPMADCSSWD	x,x,x/m,x	1	4	1	P0	AMD XOP
Logic						
PAND PANDN POR						
PXOR	mm/x,r/m	1	2	0.5	P23	
PSLL/RL W/D/Q						
PSRAW/D	mm/x,r/m	1	3	1	P1	
PSLL/RL W/D/Q						
PSRAW/D	mm/x,i	1	2	1	P1	
PSLLDQ, PSRLDQ	xmm,i	1	2	1	P1	
PTEST	xmm,r/m	2		1	P1 P3	SSE4.1
VPROTB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
VPROTB/W/D/Q	x,x,i	1	2	1	P1	AMD XOP
VPSHAB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
V. 011/10/11/0/Q	A,A,A/111	' '		1	' '	/ WID // OI

VPSHLB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
String instructions						
PCMPESTRI	x,x,i	27	17	10	P1 P2 P3	SSE4.2
PCMPESTRM	x,x,i	27	10	10	P1 P2 P3	SSE4.2
PCMPISTRI	x,x,i	7	14	3	P1 P2 P3	SSE4.2
PCMPISTRM	x,x,i	7	7	4	P1 P2 P3	SSE4.2
Encryption						
PCLMULQDQ	x,x/m,i	5	12	7	P1	pclmul
AESDEC	x,x	2	5	2	P01	aes
AESDECLAST	x,x	2	5	2	P01	aes
AESENC	x,x	2	5	2	P01	aes
AESENCLAST	x,x	2	5	2	P01	aes
AESIMC	x,x	1	5	1	P0	aes
AESKEYGENASSIST	x,x,i	1	5	1	P0	aes
Other						
EMMS		1		0.25		

Floating point XMM and YMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes
Move instructions						
MOVAPS/D MOVUPS/						
D	X,X	1	0	0.25	none	inherit domain
VMOVAPS/D	y,y	2	2	0.5	P23	ivec
MOVAPS/D MOVUPS/						
D	x,m128	1	6	0.5		
VMOVAPS/D	050			4.0		
VMOVUPS/D	y,m256	2	6	1-2		
MOVAPS/D MOVUPS/	100 v	4	_	4	Do	
D VAAOVA BOVD	m128,x	1	5	1	P3	
VMOVAPS/D	m256,y	4	5	3	P3	
VMOVUPS/D	m256,y	8	6	10	P2 P3	£
MOVSS/D	X,X	1	2	0.5	P01	fp
MOVSS/D	x,m32/64	1	6	0.5		
MOVSS/D	m32/64,x	1	5	1		
MOVHPS/D MOVLPS/	v m61	4	7	1		
D MOVHPS/D	x,m64 m64,x	1 2	8	1	P1 P3	
	,			1	-	
MOVLPS/D	m64,x	1	7	1	P3	i
MOVLHPS MOVHLPS	X,X	1	2	1	P1	ivec
MOVMSKPS/D	r32,x	2	10	1	P1 P3	
VMOVMSKPS/D	r32,y	_			D 0	
MOVNTPS/D	m128,x	1	6	2	P3	
VMOVNTPS/D	m256,y	_			Б0	00544
MOVNTSS/SD	m,x	1		4	P3	SSE4A
SHUFPS/D	x,x/m,i	1	2	1	P1	ivec
VSHUFPS/D	y,y,y/m,i	2	2	2	P1	ivec
VPERMILPS/PD	x,x,x/m	1	3	1	P1	ivec
VPERMILPS/PD	y,y,y/m	2	3	2	P1	ivec

VPERMILPS/PD	x,x/m,i	1	2	1	P1	ivec
VPERMILPS/PD	y,y/m,i	2	2	2	P1	ivec
VPERM2F128	y,y,y,i	8	4	3	P23	ivec
VPERM2F128	y,y,m,i	10		4	P23	ivec
BLENDPS/PD	x,x/m,i	1	2	0.5	P23	ivec
VBLENDPS/PD	y,y,y/m,i	2	2	1	P23	ivec
BLENDVPS/PD	x,x/m,xmm0	1	2	1	P1	ivec
VBLENDVPS/PD	y,y,y/m,y	2	2	2	P1	ivec
MOVDDUP	X,X	1	2	1	P1	ivec
MOVDDUP	x,m64	1		0.5		
VMOVDDUP	y,y	2	2	2	P1	ivec
VMOVDDUP	y,m256	2		1		
VBROADCASTSS	x,m32	1	6	0.5		
VBROADCASTSS	y,m32	2	6	0.5	P23	
VBROADCASTSD	y,m64	2	6	0.5	P23	
VBROADCASTF128	y,m128	2	6	0.5	P23	
MOVSH/LDUP	x,x	1	2	1	P1	ivec
MOVSH/LDUP	x,m128	1	_	0.5		1,00
VMOVSH/LDUP	у,у	2	2	2	P1	ivec
VMOVSH/LDUP	y,m256	2		1		1700
UNPCKH/LPS/D	x,x/m	1	2	1	P1	ivec
VUNPCKH/LPS/D	y,y,y/m	2	2	2	P1	ivec
EXTRACTPS	r32,x,i	2	10	1	P1 P3	1000
EXTRACTPS	m32,x,i	2	14	1	P1 P3	
VEXTRACTF3		1	2	1	P23	ivec
VEXTRACTF128	x,y,i m128,y,i	2	7	1	P23	ivec
INSERTPS		1	2	1	P1	
INSERTPS	X,X,İ	1		1	P1	
VINSERTF128	x,m32,i	2	2	1	P23	ivoo
VINSERTF128	y,y,x,i	2	9	1	P23	ivec
VMASKMOVPS/D	y,y,m128,i	1	9	0.5	P01	
VMASKMOVPS/D	x,x,m128	2	9	1	P01	
VMASKMOVPS/D	y,y,m256	2 18	22	7	P0 P1 P2 P3	
	m128,x,x					
VMASKMOVPS/D	m256,y,y	34	25	13	P0 P1 P2 P3	
Conversion						
CVTPD2PS	X,X	2	7	1	P01	fp
VCVTPD2PS		4	7	2	P01	fp
CVTPS2PD	x,y x,x	2	7	1	P01	fp
VCVTPS2PD		4	7	2	P01	fp
CVTSD2SS	y,x	1	4	1	P0	fp
CVTSD2SS CVTSS2SD	X,X	1	4	1	P0	
CVT3323D CVTDQ2PS	X,X	1	4	1	P0	fp fo
VCVTDQ2PS	X,X	2	4	2	P0	fp fo
	y,y		4	1	P0	fp
CVT(T) PS2DQ	X,X	1 2	4	2	P0 P0	fp fo
VCVT(T) PS2DQ CVTDQ2PD	y,y	2	7	1	P0 P01	fp fo
VCVTDQ2PD	X,X	4	8	2	P01	fp fp
	y,x					fp fn
CVT(T)PD2DQ	X,X	2	7	1	P01	fp
VCVT(T)PD2DQ	x,y	4	7	2	P01	fp
CVTPI2PS	x,mm	1	4	1	P0	fp
CVT(T)PS2PI	mm,x	1	4	1	P0	fp
CVTPI2PD	x,mm	2	7	1	P0 P1	fp

CVT(T) PD2PI	mm,x	2	7	1	P0 P1	fp
CVT(1) FB2F1	x,r32	2	14	1	P0	fp
CVTSI2SS CVT(T)SS2SI	r32,x	2	13	1	P0	
CVT(1)33231 CVTSI2SD	x,r32/64	2	14	1	P0	fp fp
	•	2		-	P0 P0	fp
CVT(T)SD2SI	r32/64,x	2	13	1	PU	fp
Arithmetic						
ADDSS/D SUBSS/D	x,x/m	1	5-6	0.5	P01	fma
ADDPS/D SUBPS/D	x,x/m	1	5-6	0.5	P01	fma
	, -					
VADDPS/D VSUBPS/D	y,y,y/m	2	5-6	1	P01	fma
ADDSUBPS/D	x,x/m	1	5-6	0.5	P01	fma
VADDSUBPS/D	y,y,y/m	2	5-6	1	P01	fma
			40		D04 D4	/5
HADDPS/D HSUBPS/D	X,X	3	10	2	P01 P1	ivec/fma
HADDPS/D HSUBPS/D	x,m128	4		2	P01 P1	ivec/fma
VHADDPS/D						
VHSUBPS/D	y,y,y	8	10	4	P01 P1	ivec/fma
VHADDPS/D						
VHSUBPS/D	y,y,m	10		4	P01 P1	ivec/fma
MULSS MULSD	x,x/m	1	5-6	0.5	P01	fma
MULPS MULPD	x,x/m	1	5-6	0.5	P01	fma
VMULPS VMULPD	y,y,y/m	2	5-6	1	P01	fma
DIVSS DIVPS	x,x/m	1	9-24	4.5-9.5	P01	fp
VDIVPS	y,y,y/m	2	9-24	9-19	P01	fp
DIVSD DIVPD	x,x/m	1	9-27	4.5-11	P01	fp
VDIVPD	y,y,y/m	2	9-27	9-22	P01	fp
RCPSS/PS	x,x/m	1	5	1	P01	fp
VRCPPS	y,y/m	2	5	2	P01	fp
CMPSS/D	3.3					,
CMPPS/D	x,x/m	1	2	0.5	P01	fp
VCMPPS/D	y,y,y/m	2	2	1	P01	fp
COMISS/D UCOMISS/						
D	x,x/m	2		1	P01 P3	fp
MAXSS/SD/PS/PD						
MINSS/SD/PS/PD	x,x/m	1	2	0.5	P01	fp
VMAXPS/D VMINPS/D	y,y,y/m	2	2	1	P01	fp
ROUNDSS/SD/PS/PD	x,x/m,i	1	4	1	P0	fp
VROUNDSS/SD/PS/	71,74111,1				. •	
PD	y,y/m,i	2	4	2	P0	fp
DPPS	x,x,i	16	25	6	P01 P23	fma
DPPS	x,m128,i	18		7	P01 P23	fma
VDPPS	y,y,y,i	25	27	13	P01 P3	fma
VDPPS	y,m256,i	29		13	P01 P3	fma
DPPD	x,x,i	15	15	5	P01 P23	fma
DPPD	x,m128,i	17		6	P01 P23	fma
VFMADDSS/SD	x,x,x,x/m	1	5-6	0.5	P01	AMD FMA4
VFMADDPS/PD	x,x,x,x/m	1	5-6	0.5	P01	AMD FMA4
VFMADDPS/PD	y,y,y,y/m	2	5-6	1	P01	AMD FMA4
All other FMA4 instruction		I	J 0-0	'	1 01	AMD FMA4
Math						

SQRTSS/PS	x,x/m	1	14-15	4.5-12	P01	fp
VSQRTPS	y,y/m	2	14-15	9-24	P01	fp
SQRTSD/PD	x,x/m	1	24-26	4.5-16.5	P01	fp
VSQRTPD	y,y/m	2	24-26	9-33	P01	fp
RSQRTSS/PS	x,x/m	1	5	1	P01	fp
VRSQRTPS	y,y/m	2	5	2	P01	fp
VFRCZSS/SD/PS/PD	x,x	2	10	2	P01	AMD XOP
VFRCZSS/SD/PS/PD	x,m	3	10	2	P01	AMD XOP
Lawia						
Logic AND/ANDN/OR/XORPS/						
PD	x,x/m	1	2	0.5	P23	ivec
VAND/ANDN/OR/	24,24		_	0.0	0	
XORPS/PD	y,y,y/m	2	2	1	P23	ivec
Other						
VZEROUPPER		9		4		32 bit mode
VZEROUPPER		16		5		64 bit mode
VZEROALL		17		6	P2 P3	32 bit mode
VZEROALL		32		10	P2 P3	64 bit mode
LDMXCSR	m32	1	10	4	P0 P3	-
STMXCSR	m32	2	19	19	P0 P3	
FXSAVE	m4096	67	136	136	P0 P1 P2 P3	
FXRSTOR	m4096	116	176	176	P0 P1 P2 P3	
XSAVE	m	122	196	196	P0 P1 P2 P3	
XRSTOR	m	177	250	250	P0 P1 P2 P3	

AMD Piledriver

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB, JNE,

etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, x = 128 bit xmm register, y = 256 bit ymm register, m = 256 and memory operand including indirect operands, m = 256 bit ymm register, m = 256 bit

Ops: Number of macro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 macro-operations use microcode.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latency listed does not include the memory operand where the listing

for register and memory operand are joined (r/m).

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However, the throughput may be limited by other bottlenecks in the pipeline.

Execution pipe: Indicates which execution pipe or unit is used for the macro-operations:

Integer pipes:

EX0: integer ALU, division

EX1: integer ALU, multiplication, jump EX01: can use either EX0 or EX1 AG01: address generation unit 0 or 1 Floating point and vector pipes:

P0: floating point add, mul, div, convert, shuffle, shift

P1: floating point add, mul, div, shuffle, shift

P2: move, integer add, boolean P3: move, integer add, boolean, store

P01: can use either P0 or P1 P23: can use either P2 or P3

Two macro-operations can execute simultaneously if they go to different

execution pipes

Domain: Tells which execution unit domain is used:

ivec: integer vector execution unit. fp: floating point execution unit. fma: floating point multiply/add subunit.

inherit: the output operand inherits the domain of the input operand.

ivec/fma means the input goes to the ivec domain and the output comes from the

fma domain.

There is an additional latency of 1 clock cycle if the output of an ivec instruction goes to the input of a fp or fma instruction, and when the output of a fp or fma instruction goes to the input of an ivec or store instruction. There is no latency between the fp and fma units. All other latencies after memory load and before mem-

ory store instructions are included in the latency counts.

An fma instruction has a latency of 5 if the output goes to another fma instruction, 6 if the output goes to an fp instruction, and 6+1 if the output goes to an ivec or

store instruction.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
		-		throughput	pipes	

Move instructions						
MOV	r8,r8	1	1	0.5	EX01	
MOV	r16,r16	1	1	0.5	EX01	
MOV	r32,r32	1	1	0.3	EX01 or AG01	
MOV	r64,r64	1	1	0.3	EX01 or AG01	
MOV	r,i	1	1	0.5	EX01	
MOV	r,m	1	4	0.5	AG01	all addr. modes
MOV	m,r	1	4	1	EX01 AG01	all addr. modes
MOV	m,i	1	7	1	LXUTAGUT	all addi. Illodes
MOVNTI	m,r	1	4	2		
MOVZX, MOVSX	r16,r8	1	1	1	EX01	
MOVZX, MOVSX	r32,r	1	1	0.5	EX01	
MOVZX, MOVSX	r64,r	1	1	0.5	EX01	
MOVSX MOVSX		1	5	0.5	EX01	
MOVZX	r,m	1	4	0.5	EX01	
	r,m	1	1	0.5		
MOVSXD	r64,r32	•			EX01	
MOVSXD	r64,m32	1	5	0.5	EX01	
CMOVcc	r,r	1	1	0.5	EX01	
CMOVcc	r,m	1	_	0.5	EX01	
XCHG	r8,r8	2	1	1	EX01	
XCHG	r16,r16	2	1	1	EX01	
XCHG	r32,r32	2	1	0.5	EX01	
XCHG	r64,r64	2	1	0.5	EX01	
VCHC	r no	2	- 40	~40	EV04	Timing depends on
XCHG	r,m	2 2	~40		EX01	hw
XLAT	_		6	2		
PUSH	r i	1		1		
PUSH		1		1		
PUSH	m	2		1		
PUSHF(D/Q)		8		4		
PUSHA(D)	_	9		9		
POP	r	1		1		
POP (D/O)	m	2		1		
POPF(D/Q)		34		18		
POPA(D)	40.5.1	14	0.0	8	E)/04	
LEA	r16,[m]	2	2-3		EX01	any addr. size
LEA	r32,[m]	2	2-3		EX01	16 bit addr. size
I = A	r20/64 [m]	4	2	0.5	EV04	scale factor > 1
LEA	r32/64,[m]	1	2	0.5	EX01	or 3 operands
LEA	r32/64,[m]	1	1	0.5	EX01	all other cases
LAHF		4	3	2		
SAHF		2	2	1		
SALC		1	1	1	F)/0.4	
BSWAP	r	1	1	0.5	EX01	
PREFETCHNTA	m	1		0.5		
PREFETCHT0/1/2	m	1		0.5		DDEEETS: "
PREFETCH/W	m	1 -		0.5		PREFETCHW
SFENCE		7		81		
LFENCE		1		0.25		
MFENCE		7		81		
Arithmetic instructions						
Arithmetic instructions ADD, SUB	r,r	1	1	0.5	EX01	
, 100, 000	1,1	'	'	0.0	L/(01	ı l

ADD, SUB						
ADD, SUB	ADD SUB	ri	1	1	0.5	FX01
ADD, SUB	1			'		
ADD, SUB			-			
ADC, SBB		m,r	1	7-8	1	
ADC, SBB ADC, SBB ADC, SBB ADC, SBB R, M, 1 1 9 1 EX01 ADC, SBB ADC, SBB M, 1 1 9 1 EX01 ADC, SBB M, 1 1 9 1 EX01 ADC, SBB M, 1 1 9 1 EX01 CMP F, 1 1 1 0.5 EX01 CMP F, 1 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 INC, DEC, NEG R 1 1 7-8 1 EX01 AAA, AAS INC, DEC, NEG R 1 0 6 DAA INC, DEC, NEG R 1 1 7-8 1 EX01 AAA, AAS INC, DEC, NEG R 1 1 4 2 EX1 MUL, IMUL R8/M8 1 4 2 EX1 MUL, IMUL R16/M16 2 4 2 EX1 MUL, IMUL R16/M16 2 4 2 EX1 MUL, IMUL R16/M16 1 4 2 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R18/M18 9 IT-22 IN-22 EX0 DIV R16/M16 7 IN-26 IN-25 EX0 DIV R16/M16 7 IN-26 IN-26 IN-26 DIV R16/M16 7 IN-26 IN-26 DIV R16/M16 7 IN-26 IN-26 DIV R16/M16 7 IN-26 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16	ADD, SUB	m,i	1	7-8	1	EX01
ADC, SBB ADC, SBB ADC, SBB ADC, SBB R, M, 1 1 9 1 EX01 ADC, SBB ADC, SBB M, 1 1 9 1 EX01 ADC, SBB M, 1 1 9 1 EX01 ADC, SBB M, 1 1 9 1 EX01 CMP F, 1 1 1 0.5 EX01 CMP F, 1 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 CMP R, 1 1 0.5 EX01 INC, DEC, NEG R 1 1 7-8 1 EX01 AAA, AAS INC, DEC, NEG R 1 0 6 DAA INC, DEC, NEG R 1 1 7-8 1 EX01 AAA, AAS INC, DEC, NEG R 1 1 4 2 EX1 MUL, IMUL R8/M8 1 4 2 EX1 MUL, IMUL R16/M16 2 4 2 EX1 MUL, IMUL R16/M16 2 4 2 EX1 MUL, IMUL R16/M16 1 4 2 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 1 6 4 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R16/R16/M16 2 2 EX1 IMUL R18/M18 9 IT-22 IN-22 EX0 DIV R16/M16 7 IN-26 IN-25 EX0 DIV R16/M16 7 IN-26 IN-26 IN-26 DIV R16/M16 7 IN-26 IN-26 DIV R16/M16 7 IN-26 IN-26 DIV R16/M16 7 IN-26 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16 7 IN-26 DIV R16/M16	I		1	1		FX01
ADC, SBB ADC, SBB ADC, SBB Mr ADC, SBB ADC, SBB ADC, SCO ADC, SCO ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SBB ADC, SCO	-		<u>-</u>			
ADC, SBB ADC, SBC ADC, SCD ADD, SCD ADC, SCD ADD, SCD ADC, SCD ADD, SCD ADC, SCD ADD, SCD ADD, SCD ADD, SCD ADD, SCD ADC, SCD ADD, SCD ADC, SCD ADD,		-				
ADC, SBB		r,m	1		1	EX01
CMP r, r 1 1 0.5 EX01 CMP r, i 1 1 0.5 EX01 CMP r, m 1 0.5 EX01 INC, DEC, NEG r 1 1 0.5 EX01 INC, DEC, NEG m 1 7-8 1 EX01 AAA, AAS 10 6 8 8 10 6 DAA 16 9 9 8 8 10 15 15 MUL, IMUL r8/m8 1 4 6 6 6 6 AAM 10 15 15 15 15 15 15 14 2 EX1 14 1 6 4 EX1 14 1 6 4 1 <t< td=""><td>ADC, SBB</td><td>m,r</td><td>1</td><td>9</td><td>1</td><td>EX01</td></t<>	ADC, SBB	m,r	1	9	1	EX01
CMP r, r 1 1 0.5 EX01 CMP r, i 1 1 0.5 EX01 CMP r, m 1 0.5 EX01 INC, DEC, NEG r 1 1 0.5 EX01 INC, DEC, NEG m 1 7-8 1 EX01 AAA, AAS 10 6 8 8 10 6 DAA 16 9 9 8 8 10 15 15 MUL, IMUL r8/m8 1 4 6 6 6 6 AAM 10 15 15 15 15 15 15 14 2 EX1 14 1 6 4 EX1 14 1 6 4 1 <t< td=""><td>ADC. SBB</td><td>m.i</td><td>1</td><td>9</td><td>1</td><td>EX01</td></t<>	ADC. SBB	m.i	1	9	1	EX01
CMP r,i 1 1 0.5 EX01 CMP r,m 1 0.5 EX01 CMP m,i 1 0.5 EX01 INC, DEC, NEG r 1 1 0.5 EX01 INC, DEC, NEG m 1 7-8 1 EX01 AAA, AAS 10 6 0 0 0 0 AAA, AAS 10 6 0						
CMP CMP r,m m,i 1 m,i 0.5 m,i EX01 0.5 EX01 INC, DEC, NEG EX01 INC, DEC, NEG EX01 INC, DEC, NEG M 1 m 7-8 INC, DEC, NEG 1 m						
CMP		r,ı		1		
INC, DEC, NEG	CMP	r,m	1		0.5	EX01
INC, DEC, NEG	CMP	m,i	1		0.5	EX01
INC, DEC, NEG	INC DEC NEG		1	1		FX01
AAA, AAS DAA DAS DAS AAD AAD AAD AAD AAD AAD			-			
DAA 16 9 DAS 20 10 AAD 4 6 AAM 10 15 15 MUL, IMUL r16/m16 2 4 2 EX1 MUL, IMUL r32/m32 1 4 2 EX1 MUL, IMUL r32/m32 1 4 2 EX1 MUL, IMUL r16,r16/m16 1 4 2 EX1 MUL, IMUL r16,r16/m16 1 4 2 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r32,r32/m32 1 4 2 EX1 IMUL r16,(r16),i 2 5 2 EX1 IMUL r16,(r64),i 1 6 4 EX1 IMUL r16,m16,i 2 2 EX1 IMUL r16,m64,i 2 2 EX1 IMUL r16,m64,i 2 1		111			'	EXUI
DAS AAD AAM AAM AAM BUL, IMUL BUL BUL, IMUL BUL BUL, IMUL BUL BUL, IMUL BUL BUL, IMUL BUL BUL BUL BUL BUL BUL BUL BUL BUL B						
AAD AAM AAM AAM AAM AAM AAM AAM AAM AAM	DAA		16	9		
AAD AAM AAM AAM AAM AAM AAM AAM AAM AAM	DAS		20	10		
AAM MUL, IMUL r8/m8 1 4 2 EX1 MUL, IMUL r16/m16 2 4 2 EX1 MUL, IMUL r16,r16/m64 1 6 4 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r32,r32/m32 1 IMUL r64,r64/m64 1 IMUL r16,r16,ii 2 5 2 EX1 IMUL r16,r16,ii 1 4 2 EX1 IMUL r16,r16,ii 1 4 2 EX1 IMUL r16,r16,ii 1 4 2 EX1 IMUL r16,r16,ii 1 6 4 EX1 IMUL r16,m16,ii 2 EX1 IMUL r16,m16,ii 2 EX1 IMUL r16,m16,ii 2 EX1 IMUL r16,m64,ii 2 EX1 IMUL r16,m64,ii 2 EX1 IMUL r16,m64,ii 2 EX1 IMUL r16,m64,ii 2 EX1 IMUL r16/m16 r13-26 I3-25 EX0 DIV r16/m16 r13-26 I3-25 EX0 DIV r16/m16 r13-26 I3-25 EX0 DIV r16/m16 r13-26 I3-25 EX0 DIV r16/m16 r13-26 I3-25 EX0 DIV r32/m32 IDIV r64/m64 IDIV r64/m64 IDIV r32/m32 IDIV r32/m32 IDIV r32/m32 IDIV r32/m32 IDIV r32/m32 IDIV r32/m32 IDIV r32/m32 IDIV r32/m32 IDIV R16-m16 IDI						
MUL, IMUL r8/m8 1 4 2 EX1 MUL, IMUL r16/m16 2 4 2 EX1 MUL, IMUL r32/m32 1 4 2 EX1 MUL, IMUL r64/m64 1 6 4 EX1 MUL, IMUL r16,r16/m16 1 4 2 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r64,r64/m64 1 6 4 EX1 IMUL r16,r16),i 2 5 2 EX1 IMUL r16,r16),i 2 5 2 EX1 IMUL r16,r16,i 1 6 4 EX1 IMUL r16,m16,i 2 1 2					4.5	
MUL, IMUL r16/m16 2 4 2 EX1 MUL, IMUL r32/m32 1 4 2 EX1 MUL, IMUL r64/m64 1 6 4 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r32,r32/m32 1 4 2 EX1 IMUL r16,(r16),i 2 5 2 EX1 IMUL r16,(r16),i 2 5 2 EX1 IMUL r16,(r16),i 1 6 4 EX1 IMUL r16,(r16),i 1 1 1						
MUL, IMUL r32/m32 1 4 2 EX1 MUL, IMUL r64/m64 1 6 4 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r16,r64/m64 1 6 4 EX1 IMUL r64,r64/m64 1 6 4 EX1 IMUL r16,(r16),i 2 5 2 EX1 IMUL r32,(r32),i 1 4 2 EX1 IMUL r64,(r64),i 1 6 4 EX1 IMUL r16,m16,i 2 2 EX1 IMUL r32,m32,i 2 2 EX1 IMUL r34,m64,i 2 4 EX1 IMUL r64,m64,i 2 4 EX1 IMUL r64,m64,i 2 13-25 EX0 DIV r16/m16 7 13-26 13-25 EX0 DIV r64/m64 2	MUL, IMUL	r8/m8		4		EX1
MUL, IMUL r64/m64 1 6 4 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r32,r32/m32 1 4 2 EX1 IMUL r64,r64/m64 1 6 4 EX1 IMUL r16,r616,i 2 5 2 EX1 IMUL r32,(r32),i 1 4 2 EX1 IMUL r64,(r64),i 1 6 4 EX1 IMUL r16,m16,i 2 2 EX1 IMUL r32,m32,i 2 2 EX1 IMUL r64,m64,i 2 4 EX1 IMUL r64,m64,i 2 4 EX1 IMUL r64,m64,i 2 4 EX1 DIV r16/m16 7 13-26 13-25 EX0 DIV r64/m64 2 13-71 13-71 EX0 IDIV r8/m8 9 <td>MUL, IMUL</td> <td>r16/m16</td> <td>2</td> <td>4</td> <td>2</td> <td>EX1</td>	MUL, IMUL	r16/m16	2	4	2	EX1
MUL, IMUL r64/m64 1 6 4 EX1 IMUL r16,r16/m16 1 4 2 EX1 IMUL r32,r32/m32 1 4 2 EX1 IMUL r64,r64/m64 1 6 4 EX1 IMUL r16,r616,i 2 5 2 EX1 IMUL r32,(r32),i 1 4 2 EX1 IMUL r64,(r64),i 1 6 4 EX1 IMUL r16,m16,i 2 2 EX1 IMUL r32,m32,i 2 2 EX1 IMUL r64,m64,i 2 4 EX1 IMUL r64,m64,i 2 4 EX1 IMUL r64,m64,i 2 4 EX1 DIV r16/m16 7 13-26 13-25 EX0 DIV r64/m64 2 13-71 13-71 EX0 IDIV r8/m8 9 <td></td> <td>r32/m32</td> <td>1</td> <td>4</td> <td>2</td> <td>FX1</td>		r32/m32	1	4	2	FX1
IMUL						
IMUL						
IMUL						
IMUL	IMUL	r32,r32/m32	1	4		EX1
IMUL	IMUL	r64,r64/m64	1	6	4	EX1
IMUL	IMUI	r16 (r16) i	2	5	2	FX1
IMUL		, ,				
IMUL						
IMUL				6		
IMUL	IMUL	r16,m16,i			2	EX1
DIV r8/m8 9 17-22 13-22 EX0 DIV r16/m16 7 13-26 13-25 EX0 DIV r32/m32 2 12-40 12-40 EX0 DIV r64/m64 2 13-71 13-71 EX0 IDIV r8/m8 9 17-21 13-18 EX0 IDIV r16/m16 7 13-26 13-25 EX0 IDIV r32/m32 2 13-40 13-40 EX0 IDIV r64/m64 2 13-71 13-71 EX0 CBW, CWDE, CDQE 1 1 EX01 EX01 CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 EX01 CWD 2 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 7-8 1 EX01	IMUL	r32,m32,i	2		2	EX1
DIV r8/m8 9 17-22 13-22 EX0 DIV r16/m16 7 13-26 13-25 EX0 DIV r32/m32 2 12-40 12-40 EX0 DIV r64/m64 2 13-71 13-71 EX0 IDIV r8/m8 9 17-21 13-18 EX0 IDIV r16/m16 7 13-26 13-25 EX0 IDIV r32/m32 2 13-40 13-40 EX0 IDIV r64/m64 2 13-71 13-71 EX0 CBW, CWDE, CDQE 1 1 EX01 EX01 CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 EX01 CWD 2 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 7-8 1 EX01	IMUI	r64 m64 i	2		4	FX1
DIV r16/m16 7 13-26 13-25 EX0 DIV r32/m32 2 12-40 12-40 EX0 DIV r64/m64 2 13-71 13-71 EX0 IDIV r8/m8 9 17-21 13-18 EX0 IDIV r16/m16 7 13-26 13-25 EX0 IDIV r32/m32 2 13-40 13-40 EX0 IDIV r64/m64 2 13-71 13-71 EX0 CBW, CWDE, CDQE 1 1 EX01 EX01 CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 EX01 CWD 2 1 1 0.5 EX01 CWD 7,r 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 7-8 1 EX01 A				17 22		
DIV r32/m32 2 12-40 12-40 EX0 DIV r64/m64 2 13-71 13-71 EX0 IDIV r8/m8 9 17-21 13-18 EX0 IDIV r16/m16 7 13-26 13-25 EX0 IDIV r32/m32 2 13-40 13-40 EX0 IDIV r64/m64 2 13-71 13-71 EX0 CBW, CWDE, CDQE 1 1 0.5 EX01 CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 EX01 Logic instructions 7,r 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 7-8 1 EX01 TEST						
DIV r64/m64 2 13-71 13-71 EX0 IDIV r8/m8 9 17-21 13-18 EX0 IDIV r16/m16 7 13-26 13-25 EX0 IDIV r32/m32 2 13-40 13-40 EX0 CBW, CWDE, CDQE 1 1 EX0 EX01 CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 EX01 Logic instructions 7,r 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01 <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>						
IDIV	DIV	r32/m32		12-40	12-40	EX0
IDIV	DIV	r64/m64	2	13-71	13-71	EX0
IDIV	IDIV	r8/m8	9	17-21	13-18	EX0
IDIV IDIV						
IDIV CBW, CWDE, CDQE CDQ, CQO CWD						
CBW, CWDE, CDQE 1 1 1 0.5 EX01 CDQ, CQO 2 1 1 0.5 EX01 CWD 2 1 1 EX01 Logic instructions AND, OR, XOR r,r 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01						
CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 EX01 Logic instructions		r64/m64		13-71	13-71	
CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 EX01 Logic instructions	CBW, CWDE, CDQE		1	1		EX01
CWD 2 1 1 EX01 Logic instructions AND, OR, XOR AND, OR, XOR r,r 1 1 0.5 EX01 AND, OR, XOR r,m 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01	CDQ, CQO		1	1	0.5	EX01
Logic instructions r,r 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01	1 '					
AND, OR, XOR r,r 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01			_	'	'	
AND, OR, XOR r,r 1 1 0.5 EX01 AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01						
AND, OR, XOR r,i 1 1 0.5 EX01 AND, OR, XOR r,m 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01	-				_	
AND, OR, XOR r,m 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01	AND, OR, XOR	r,r	1	1	0.5	EX01
AND, OR, XOR r,m 1 0.5 EX01 AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01	AND, OR, XOR	r,i	1	1	0.5	EX01
AND, OR, XOR m,r 1 7-8 1 EX01 AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01			1			FX01
AND, OR, XOR m,i 1 7-8 1 EX01 TEST r,r 1 1 0.5 EX01				7.0		
TEST r,r 1 1 0.5 EX01						
TEST r,i 1 1 0.5 EX01		r,r		1		
	TEST	r,i	1	1	0.5	EX01

TEST	m,r	1		0.5	EX01	
TEST	m,i	1		0.5	EX01	
NOT	r	1	1	0.5	EX01	
NOT	m	1	7-8	1	EX01	
ANDN		1	1	0.5	EX01	BMI1
SHL, SHR, SAR	r,r,r	1	1	0.5	EX01	DIVILI
	r,i/CL					
ROL, ROR	r,i/CL	1	1	0.5	EX01	
RCL	r,1	1	1		EX01	
RCL	r,i	16	7		EX01	
RCL	r,cl	17	7		EX01	
RCR	r,1	1	1		EX01	
RCR	r,i	15	7		EX01	
RCR	r,cl	16	6		EX01	
SHLD, SHRD	r,r,i	6	3	3	EX01	
SHLD, SHRD	r,r,cl	7	3	3	EX01	
SHLD, SHRD	m,r,i/CL	8		3.5	EX01	
BT	r,r/i	1	1	0.5	EX01	
BT	m,i	1		0.5	EX01	
BT	m,r	7		4	EX01	
BTC, BTR, BTS	r,r/i	2	2	1	EX01	
BTC, BTR, BTS	m,i	4	20		EX01	
BTC, BTR, BTS	m,r	10	21		EX01	
BSF	r,r	6	3	3	EX01	
BSF	r,m	8	4	4	EX01	
BSR	r,r	7	4	4	EX01	
BSR	r,m	9		5	EX01	
SETcc	r	1	1	0.5	EX01	
SETcc	m	1	-	1	EX01	
CLC, STC		1		0.5	EX01	
CMC		1	1		EX01	
CLD		2	•	3	27101	
STD		2		4		
POPCNT	r16/32,r16/32	1	4	2		SSE4.2
POPCNT	r64,r64	1	4	4		SSE4.2
LZCNT	r,r	1	2	2	EX0	LZCNT
TZCNT	r,r	2	2	2	LAU	BMI1
BEXTR	r,r,r	2	2	0.67		BMI1
BEXTR	r,r,i	2	2	0.67		AMD TBM
BLSI		2	2	1		BMI1
BLSMSK	r,r	2	2	1		BMI1
BLSR	r,r	2	2	1		BMI1
BLCFILL	r,r	2	2	1		AMD TBM
1	r,r	2				AMD TBM
BLCI	r,r		2	1		
BLCIC	r,r	2	2	1		AMD TBM
BLCMSK	r,r	2	2	1		AMD TBM
BLCS	r,r	2	2	1		AMD TBM
BLSFILL	r,r	2	2	1		AMD TBM
BLSI	r,r	2	2	1		AMD TBM
BLSIC	r,r	2	2	1		AMD TBM
T1MSKC	r,r	2	2	1		AMD TBM
TZMSK	r,r	2	2	1		AMD TBM
Control transfer instru	ctions					

JMP JMP JMP Jcc fused CMP+Jcc J(E/R)CXZ LOOP LOOPE LOOPNE CALL CALL CALL RET RET BOUND INTO	short/near r m short/near short short short short r near r m	1 1 1 1 1 1 1 2 2 2 3 1 4 11 4		2 2 1-2 1-2 1-2 1-2 2 2 2 2 2 2 2	EX1 EX1 EX1 EX1 EX1 EX1 EX1 EX1 EX1 EX1	2 if jumping 2 if jumping 2 if jumping 2 if jumping 2 if jumping for no jump
String instructions LODS REP LODS REP LODS STOS REP STOS REP STOS MOVS REP MOVS REP MOVS REP MOVS SCAS REP SCAS CMPS REP CMPS	m8/m16 m32/m64	3 6n 6n 3 1n 3 per 16B 5 1-3n 4.5 pr 16B 3 7n 6 9n		3 3n 2.5n 3 1n 3 per 16B 3 1n 3 per 16B 3 3-4n 3		small n best case small n best case
Synchronization LOCK ADD XADD LOCK XADD CMPXCHG LOCK CMPXCHG CMPXCHG LOCK CMPXCHG CMPXCHGBB LOCK CMPXCHG8B LOCK CMPXCHG8B LOCK CMPXCHG16B LOCK CMPXCHG16B	m,r m,r m,r8/16 m,r8/16 m,r32/64 m,r32/64 m64 m64 m128 m128	1 4 4 5 5 6 6 18 18 22 22	~40 20 ~39 23 ~40 20 ~40 25 ~42 66 ~80			
Other NOP (90) Long NOP (0F 1F) PAUSE ENTER ENTER LEAVE CPUID XGETBV	a,0 a,b	1 40 13 20+3b 2 38-64 4		0.25 0.25 40 21 16+4b 4 105-271	none none	

RDTSC		36		42	
RDPMC		21		310	
CRC32	r32,r8	3	3	2	
CRC32	r32,r16	5	5	5	
CRC32	r32,r32	5	6	6	

Floating point x87 instructions								
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes		
Move instructions								
FLD	r	1	2	0.5	P01	fp		
FLD	m32/64	1	7	1		fp		
FLD	m80	8	20	4		fp		
FBLD	m80	60	64	35	P0 P1 P2 P3	fp		
FST(P)	r	1	2	0.5	P01	fp		
FST(P)	m32/64	2	7	1		fp		
FSTP	m80	13	22	20		fp		
FBSTP	m80	239	220		P0 P1 F3	fp		
FXCH	r	1	0	0.5	P01	inherit		
FILD	m	1	11	1	F3	fp		
FIST(T)(P)	m	2	7	1	P0 F3	fp		
FLDZ, FLD1		1		0.5	P01	fp		
FCMOVcc	st0,r	8	3	3	P0 P1 F3	fp		
FFREE	r	1		0.25	none			
FINCSTP, FDECSTP		1	0	0.25	none	inherit		
FNSTSW	AX	3		19	P0 P2 P3			
FNSTSW	m16	2		17	P0 P2 P3			
FLDCW	m16	1		3				
FNSTCW	m16	2		2				
Arithmetic instructions	 							
FADD(P),FSUB(R)(P)	r/m	1	5-6	1	P01	fma		
FIADD,FISUB(R)	m	2		2	P01	fma		
FMUL(P)	r/m	1	5-6	1	P01	fma		
FIMUL	m	2		2	P01	fma		
FDIV(R)(P)	r	1	9-40	4-16	P01	fp		
FDIV(R)	m	1			P01	fp		
FIDIV(R)	m	2			P01	fp		
FABS, FCHS		1	2	0.5	P01	fp		
FCOM(P), FUCOM(P)	r/m	1		0.5	P01	fp		
FCOMPP, FUCOMPP		1		0.5	P01	fp		
FCOMI(P)	r	2	2	1	P0 P1 F3	fp		
FICOM(P)	m	2		1	P01	fp		
FTST		1		0.5	P01	fp		
FXAM		1	~20	0.5	P01	fp		
FRNDINT		1	4	1	P0	fp		
FPREM		1	17-60		P0	fp		
FPREM1		1	17-60		P0	fp		
Math								
FSQRT		1	14-50	5-20	P01			
FLDPI, etc.		1		0.5	P01			

FSIN		10-162	60-210	60-146	P0 P1 P3	
FCOS		160-170	~154	~154	P0 P1 P3	
FSINCOS		12-166	86-141	86-141	P0 P1 P3	
FPTAN		11-190	166-231	86-204	P0 P1 P3	
FPATAN		10-355	60-352	60-352	P0 P1 P3	
FSCALE		8	44	5	P0 P1 P3	
FXTRACT		12	7	5	P0 P1 P3	
F2XM1		10	60-73		P0 P1 P3	
FYL2X		10-176			P0 P1 P3	
FYL2XP1		10-176			P0 P1 P3	
Other						
FNOP		1		0.25	none	
(F)WAIT		1		0.25	none	
FNCLEX		18		54	P0	
FNINIT		31		134	P0	
FNSAVE	m864	103	300	300	P0 P1 P2 P3	
FRSTOR	m864	76	236	236	P0 P3	

Integer vector instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions					• •	
MOVD	r32/64, mm/x	1	8	1		P3
MOVD	mm/x, r32/64	2	10	1		
MOVD	mm/x,m32	1	6	0.5		
MOVD	m32,mm/x	1	5	1		P3
MOVQ	mm/x,mm/x	1	2	0.5	P23	
MOVQ	mm/x,m64	1	6	0.5		
MOVQ	m64,mm/x	1	5	1	P3	
MOVDQA	xmm,xmm	1	0	0.25	none	inherit domain
MOVDQA	xmm,m	1	6	0.5		
MOVDQA	m,xmm	1	5	1	P3	
VMOVDQA	ymm,ymm	2	2	0.5	P23	
VMOVDQA	ymm,m256	2	6	1		
VMOVDQA	m256,ymm	4	11	17	P3	
MOVDQU	xmm,xmm	1	0	0.25	none	inherit domain
MOVDQU	xmm,m	1	6	0.5		
MOVDQU	m,xmm	1	5	1	P3	
LDDQU	xmm,m	1	6	0.5		
VMOVDQU	ymm,m256	2	6	1		
VMOVDQU	m256,ymm	8	14	20	P2 P3	
MOVDQ2Q	mm,xmm	1	2	0.5	P23	
MOVQ2DQ	xmm,mm	1	2	0.5	P23	
MOVNTQ	m,mm	1	5	2	P3	
MOVNTDQ	m,xmm	1	5	2	P3	
MOVNTDQA	xmm,m	1	6	0.5		
PACKSSWB/DW	mm/x,r/m	1	2	1	P1	
PACKUSWB	mm/x,r/m	1	2	1	P1	
PUNPCKH/LBW/WD/ DQ	mm/x,r/m	1	2	1	P1	

	I	ı	T.	ı	1	1
PUNPCKHQDQ	xmm,r/m	1	2	1	P1	
PUNPCKLQDQ	xmm,r/m	1	2	1	P1	
PSHUFB	mm/x,r/m	1	3	1	P1	
PSHUFD	xmm,xmm,i	1	2	1	P1	
PSHUFW	mm,mm,i	1	2	1	P1	
PSHUFL/HW	xmm,xmm,i	1	2	1	P1	
PALIGNR	mm/x,r/m,i	1	2	1	P1	
PBLENDW	xmm,r/m	1	2	0.5	P23	SSE4.1
MASKMOVQ	mm,mm	31	36	59	P3	
MASKMOVDQU	xmm,xmm	64	59	92	P1 P3	
PMOVMSKB	r32,mm/x	2	10	1	P1 P3	
PEXTRB/W/D/Q	r,x/mm,i	2	10	1	P1 P3	SSE4.1
PINSRB/W/D/Q	x/mm,r,i	2	12	2	P1	
EXTRQ	x,i,i	1	3	1	P1	AMD SSE4A
EXTRQ	x,x	1	1	1	P1	AMD SSE4A
INSERTQ	x,x,i,i	1	1	1	P1	AMD SSE4A
INSERTQ	x,x,	1	1	1	P1	AMD SSE4A
PMOVSXBW/BD/BQ/	7,7	•	•			7 11112 002 17 (
WD/WQ/DQ	x,x	1	2	1	P1	SSE4.1
PMOVZXBW/BD/BQ/	71,71	·	_			
WD/WQ/DQ	x,x	1	2	1	P1	SSE4.1
VPCMOV	x,x,x,x/m	1	2	1	P1	AMD XOP
VPCMOV	y,y,y,y/m	2	2	2	P1	AMD XOP
VPPERM	x,x,x,x/m	1	2	1	P1	AMD XOP
=	23,23,23,24	·	_			7 2 7 1 3 .
Arithmetic instructions	 2					
PADDB/W/D/Q/SB/SW/	1					
USB/USW	mm/x,r/m	1	2	0.5	P23	
PSUBB/W/D/Q/SB/SW/	· ·	-	_			
USB/USW	mm/x,r/m	1	2	0.5	P23	
PHADD/SUB(S)W/D	x,x	3	5	2	P1 P23	SSSE3
PHADD/SUB(S)W/D	x,m	4	5	2	P1 P23	SSSE3
PCMPEQ/GT B/W/D	mm/x,r/m	1	2	0.5	P23	
PCMPEQQ	mm/x,r/m	1	2	0.5	P23	SSE4.1
PCMPGTQ	mm/x,r/m	1	2	0.5	P23	SSE4.2
PMULLW PMULHW	, ,				-	
PMULHUW PMULUDQ	mm/x,r/m	1	4	1	P0	
PMULLD	x,r/m	1	5	2	P0	SSE4.1
PMULDQ	x,r/m	1	4	1	P0	SSE4.1
PMULHRSW	mm/x,r/m	1	4	1	P0	SSSE3
PMADDWD	mm/x,r/m	1	4	1	P0	
PMADDUBSW	mm/x,r/m	1	4	1	P0	
PAVGB/W	mm/x,r/m	1	2	0.5	P23	
PMIN/MAX SB/SW/ SD	,,,,,,,	·	_		. =0	
UB/UW/UD	mm/x,r/m	1	2	0.5	P23	
PHMINPOSUW	x,r/m	2	4	1	P1 P23	SSE4.1
PABSB/W/D	mm/x,r/m	1	2	0.5	P23	SSSE3
PSIGNB/W/D	mm/x,r/m	1	2	0.5	P23	SSSE3
PSADBW	mm/x,r/m	2	4	1	P23	
MPSADBW	x,x,i	8	8	4	P1 P23	SSE4.1
5, 5511	,,,,,,			·	20	AMD XOP
VPCOMB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
	,,, -, -, -, -, -, -, -, -, -,	•	_			AMD XOP
VPCOMUB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
	,	•	,		•	

VDLIA DDDW/DD/DO/	1		[ı	l	I
VPHADDBW/BD/BQ/ WD/WQ/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPHADDUBW/BD/BQ/	X,X/111	'		0.5	F23	AIVID AOP
WD/WQ/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPHSUBBW/WD/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPMACSWW/WD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSDD	x,x,x/m,x	1	5	2	P0	AMD XOP
VPMACSDD VPMACSDQH/L	x,x,x/111,x x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSSWW/WD		1	4	1	P0	AMD XOP
VPMACSSWW/WD	x,x,x/m,x	1	5	2	P0	AMD XOP
	x,x,x/m,x	1 1	4	1	P0 P0	
VPMACSSDQH/L	x,x,x/m,x		4	1	P0 P0	AMD XOP
VPMADCSWD	x,x,x/m,x	1				AMD XOP
VPMADCSSWD	x,x,x/m,x	1	4	1	P0	AMD XOP
Logic						
PAND PANDN POR						
PXOR	mm/x,r/m	1	2	0.5	P23	
PSLL/RL W/D/Q		-				
PSRAW/D	mm/x,r/m	1	3	1	P1	
PSLL/RL W/D/Q						
PSRAW/D	mm/x,i	1	2	1	P1	
PSLLDQ, PSRLDQ	x,i	1	2	1	P1	
PTEST	x,r/m	2		1	P1 P3	SSE4.1
VPROTB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
VPROTB/W/D/Q	x,x,i	1	2	1	P1	AMD XOP
VPSHAB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
VPSHLB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
String instructions						
PCMPESTRI	x,x,i	27	16	10	P1 P2 P3	SSE4.2
PCMPESTRM	x,x,i x,x,i	27	10	10	P1 P2 P3	SSE4.2
PCMPISTRI	x,x,i x,x,i	7	13	3	P1 P2 P3	SSE4.2
PCMPISTRM	x,x,i x,x,i	7	7	4	P1 P2 P3	SSE4.2
	,·- , -					
Encryption						
PCLMULQDQ	x,x/m,i	5	12	7	P1	pclmul
VPCLMULQDQ	x,x,x,i	6	12	7	P1	pclmul
PCLMULQDQ	x,x,m,i	7	12	7	P1	pclmul
AESDEC	x,x	2	5	2	P01	aes
AESDECLAST	x,x	2	5	2	P01	aes
AESENC	x,x	2	5	2	P01	aes
AESENCLAST	x,x	2	5	2	P01	aes
AESIMC	x,x	1	5	1	P0	aes
AESKEYGENASSIST	x,x,i	1	5	1	P0	aes
	, ,				_	
Other						
EMMS		1		0.25		

Floating point XMM and YMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes
Move instructions						

MOVAPS/D MOVUPS/		4	0	0.05		in bouit done in
D VMOVAPS/D	X,X	1 2	0 2	0.25 0.5	none P23	inherit domain
MOVAPS/D MOVUPS/	у,у	2		0.5	P23	ivec
D	x,m128	1	6	0.5		
VMOVAPS/D	λ,111120			0.0		
VMOVUPS/D	y,m256	2	6	1		
MOVAPS/D MOVUPS/	,					
D	m128,x	1	5	1	P3	
VMOVAPS/D	m256,y	4	11	17	P3	
VMOVUPS/D	m256,y	8	15	20	P2 P3	
MOVSS/D	x,x	1	2	0.5	P01	fp
MOVSS/D	x,m32/64	1	6	0.5		
MOVSS/D	m32/64,x	1	5	1		
MOVHPS/D	x,m64	1	8	1	P1	
MOVLPS/D	x,m64	1	7	0.5	P01	
MOVHPS/D	m64,x	2	7	1	P1 P3	
MOVLPS/D	m64,x	1	6	1	P3	
MOVLHPS MOVHLPS	x,x	1	2	1	P1	ivec
MOVMSKPS/D	r32,x	2	10	1	P1 P3	
VMOVMSKPS/D	r32,y	2		1		
MOVNTPS/D	m128,x	1	5	2	P3	
VMOVNTPS/D	m256,y	4		18		
MOVNTSS/SD	m,x	1		4	P3	AMD SSE4A
SHUFPS/D	x,x/m,i	1	2	1	P1	ivec
VSHUFPS/D	y,y,y/m,i	2	2	2	P1	ivec
VPERMILPS/PD	x,x,x/m	1	3	1	P1	ivec
VPERMILPS/PD	y,y,y/m	2	3	2	P1	ivec
VPERMILPS/PD	x,x/m,i	1	2	1	P1	ivec
VPERMILPS/PD	y,y/m,i	2	2	2	P1	ivec
VPERM2F128	y,y,y,i	8	4	3	P23	ivec
VPERM2F128	y,y,m,i	10		4	P23	ivec
BLENDPS/PD	x,x/m,i	1	2	0.5	P23	ivec
VBLENDPS/PD	y,y,y/m,i	2	2	1	P23	ivec
BLENDVPS/PD	x,x/m,xmm0	1	2	1	P1	ivec
VBLENDVPS/PD	y,y,y/m,y	2	2	2	P1	ivec
MOVDDUP	X,X	1	2	1	P1	ivec
MOVDDUP	x,m64	1		0.5		
VMOVDDUP	y,y	2	2	2	P1	ivec
VMOVDDUP	y,m256	2	_	1		
VBROADCASTSS	x,m32	1	6	0.5		
VBROADCASTSS	y,m32	2	6	0.5	P23	
VBROADCASTSD	y,m64	2	6	0.5	P23	
VBROADCASTF128	y,m128	2	6	0.5	P23	
MOVSH/LDUP	X,X	1	2	1	P1	ivec
MOVSH/LDUP	x,m128	1	_	0.5		
VMOVSH/LDUP	у,у	2	2	2	P1	ivec
VMOVSH/LDUP	y,m256	2	_	1		
UNPCKH/LPS/D	x,x/m	1	2	1	P1	ivec
VUNPCKH/LPS/D	y,y,y/m	2	2	2	P1	ivec
EXTRACTPS	r32,x,i	2	_	1	P1 P3	
EXTRACTPS	m32,x,i	2	6	1	P1 P3	
VEXTRACTF128	x,y,i	1	2	0.5	P23	ivec
VEXTRACTF128	m128,y,i	2	6	1	P23	

INSERTPS	x,x,i	1	2	1	P1	
INSERTPS	x,m32,i	1	7	2	P1	
VINSERTF128	y,y,x,i	2	2	1	P23	ivec
VINSERTF128	y,y,m128,i	2	13	1	P23	
VMASKMOVPS/D	x,x,m128	1	7	0.5	P01	
VMASKMOVPS/D	y,y,m256	2	13	1	P01	
		18	~100	~90	P0 P1 P2 P3	
VMASKMOVPS/D	m128,x,x					
VMASKMOVPS/D	m256,y,y	34	~190	~180	P0 P1 P2 P3	
Conversion				_	D0.4	
CVTPD2PS	x,x	2	8	1	P01	ivec/fp
VCVTPD2PS	x,y	4	7	2	P01	ivec/fp
CVTPS2PD	X,X	2	8	1	P01	ivec/fp
VCVTPS2PD	y,x	4	8	2	P01	ivec/fp
CVTSD2SS	X,X	1	4	1	P0	fp
CVTSS2SD	x,x	1	4	1	P0	fp
CVTDQ2PS	X,X	1	4	1	P0	fp
VCVTDQ2PS	y,y	2	4	2	P0	fp
CVT(T) PS2DQ	x,x	1	4	1	P0	fp
VCVT(T) PS2DQ	y,y	2	4	2	P0	fp
CVTDQ2PD	x,x	2	8	_ 1	P01	ivec/fp
VCVTDQ2PD	y,x	4	8	2	P01	ivec/fp
CVT(T)PD2DQ	X,X	2	8	1	P01	fp/ivec
VCVT(T)PD2DQ		4	7	2	P01	fp/ivec
CVTPI2PS	x,y	2	8	1		•
	x,mm				P0 P23	ivec/fp
CVT(T)PS2PI	mm,x	1	4	1	P0	fp
CVTPI2PD	x,mm	2	7	1	P0 P1	ivec/fp
CVT(T) PD2PI	mm,x	2	7	1	P0 P1	fp/ivec
CVTSI2SS	x,r32	2	13	1	P0	fp
CVT(T)SS2SI	r32,x	2	12	1	P0 P3	fp
CVTSI2SD	x,r32/64	2	13	1	P0	fp
CVT(T)SD2SI	r32/64,x	2	12	1	P0 P3	fp
VCVTPS2PH	x/m,x,i	2	8	2	P0 P1	F16C
VCVTPS2PH	x/m,y,i	4	8	2	P0 P1	F16C
VCVTPH2PS	x,x/m	2	8	2	P0 P1	F16C
VCVTPH2PS	y,x/m	4	8	2	P0 P1	F16C
	-					
Arithmetic						
ADDSS/D SUBSS/D	x,x/m	1	5-6	0.5	P01	fma
ADDPS/D SUBPS/D	x,x/m	1	5-6	0.5	P01	fma
	ŕ					
VADDPS/D VSUBPS/D	y,y,y/m	2	5-6	1	P01	fma
ADDSUBPS/D	x,x/m	1	5-6	0.5	P01	fma
VADDSUBPS/D	y,y,y/m	2	5-6	1	P01	fma
	37373					
HADDPS/D HSUBPS/D	x,x	3	10	2	P01 P1	ivec/fma
		_				<u>.</u>
HADDPS/D HSUBPS/D	x,m	4		2	P01 P1	ivec/fma
VHADDPS/D	_	_		_		
VHSUBPS/D	y,y,y/m	8	10	4	P01 P1	ivec/fma
MULSS MULSD	x,x/m	1	5-6	0.5	P01	fma
MULPS MULPD	x,x/m	1	5-6	0.5	P01	fma
VMULPS VMULPD	y,y,y/m	2	5-6	1	P01	fma
DIVSS DIVPS	x,x/m	1	9-24	5-10	P01	fp

h	1 -	1 .	1			1 -
VDIVPS	y,y,y/m	2	9-24	9-20	P01	fp
DIVSD DIVPD	x,x/m	1	9-27	5-10	P01	fp
VDIVPD	y,y,y/m	2	9-27	9-18	P01	fp
RCPSS/PS	x,x/m	1	5	1	P01	fp
VRCPPS	y,y/m	2	5	2	P01	fp
CMPSS/D						
CMPPS/D	x,x/m	1	2	0.5	P01	fp
VCMPPS/D	y,y,y/m	2	2	1	P01	fp
COMISS/D UCOMISS/	3.3.3					'
D	x,x/m	2		1	P01 P3	fp
MAXSS/SD/PS/PD	,					'
MINSS/SD/PS/PD	x,x/m	1	2	0.5	P01	fp
	,					'
VMAXPS/D VMINPS/D	y,y,y/m	2	2	1	P01	fp
ROUNDSS/SD/PS/PD	x,x/m,i	1	4	1	P0	fp
VROUNDSS/SD/PS/		-		-		7
PD	y,y/m,i	2	4	2	P0	fp
DPPS	x,x,i	16	25	6	P01 P23	SSE4.1
DPPS	x,m,i	18		7	P01 P23	SSE4.1
VDPPS	y,y,y,i	25	27	13	P01 P3	SSE4.1
VDPPS		29	21	13	P01 P3	SSE4.1
DPPD	y,m,i	15	15		P01 P23	SSE4.1
	X,X,İ		15	5		
DPPD	x,m,i	17	5.0	6	P01 P23	SSE4.1
VFMADD132SS/SD	x,x,x/m	1	5-6	1	P01	FMA3
VFMADD132PS/PD	x,x,x/m	1	5-6	1	P01	FMA3
VFMADD132PS/PD	y,y,y/m	2	5-6	1	P01	FMA3
All other FMA3 instruction	ons: same as a	bove				FMA3
VFMADDSS/SD	x,x,x,x/m	1	5-6	0.5	P01	AMD FMA4
VFMADDPS/PD	x,x,x,x/m	1	5-6	0.5	P01	AMD FMA4
VFMADDPS/PD	y,y,y,y/m	2	5-6	1	P01	AMD FMA4
All other FMA4 instruction	ons: same as a	bove	·			AMD FMA4
Math						
SQRTSS/PS	x,x/m	1	13-15	5-12	P01	fp
VSQRTPS	y,y/m	2	14-15	9-24	P01	fp
SQRTSD/PD	x,x/m	1	24-26	5-15	P01	fp
VSQRTPD	y,y/m	2	24-26	9-29	P01	fp
RSQRTSS/PS	x,x/m	1	5	1	P01	fp
VRSQRTPS	y,y/m	2	5	2	P01	fp
VFRCZSS/SD/PS/PD	X,X	2	10	2	P01	AMD XOP
VFRCZSS/SD/PS/PD		3	10	2	P01	AMD XOP
VFRGZSS/SD/FS/FD	x,m	3	10	2	PUT	AIVID AOP
Lagia						
Logic						
AND/ANDN/OR/XORPS/	x,x/m	1	2	0.5	P23	ivec
VAND/ANDN/OR/	۸,۸/۱۱۱	l I		0.5	1 23	IVEC
XORPS/PD	y,y,y/m	2	2	1	P23	ivec
ACINI O/I D	y,y,y/111	_	_	'	1 20	1000
Other						
VZEROUPPER		9		4	P2 P3	32 bit mode
VZEROUPPER		16		5	P2 P3	64 bit mode
VZEROOFFER		17		6	P2 P3	32 bit mode
VZEROALL		32		10	P2 P3 P2 P3	64 bit mode
	m20	7				04 DIL IIIOUE
LDMXCSR	m32	/	1	34	P0 P3	

STMXCSR	m32	2		17	P0 P3
FXSAVE	m4096	67	136	136	P0 P1 P2 P3
FXRSTOR	m4096	116	176	176	P0 P1 P2 P3
XSAVE	m	122	196	196	P0 P1 P2 P3
XRSTOR	m	177	250	250	P0 P1 P2 P3

AMD Steamroller

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB, JNE,

etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, x = 128 bit xmm register, y = 256 bit ymm register, m = any memory operand including indirect operands, m64 means 64-bit memory operand, etc.

Ops: Number of macro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 macro-operations use microcode.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. The latency listed does not include the memory operand where the listing for register and memory operand are joined (r/

m).

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However, the throughput may be limited by other bottlenecks in the pipeline.

Execution pipe: Indicates which execution pipe or unit is used for the macro-operations:

Integer pipes:

EX0: integer ALU, division

EX1: integer ALU, multiplication, jump EX01: can use either EX0 or EX1 AG01: address generation unit 0 or 1 Floating point and vector pipes:

P0: floating point add, mul, div. Integer add, mul, bool P1: floating point add, mul, div. Shuffle, shift, pack

P2: Integer add. Bool, store P01: can use either P0 or P1 P02: can use either P0 or P2

Two macro-operations can execute simultaneously if they go to different

execution pipes

Domain: Tells which execution unit domain is used:

ivec: integer vector execution unit. fp: floating point execution unit. fma: floating point multiply/add subunit.

inherit: the output operand inherits the domain of the input operand.

ivec/fma means the input goes to the ivec domain and the output comes from the

fma domain.

There is an additional latency of 1 clock cycle if the output of an ivec instruction goes to the input of a fp or fma instruction, and when the output of a fp or fma instruction goes to the input of an ivec or store instruction. There is no latency between the fp and fma units. All other latencies after memory load and before mem-

ory store instructions are included in the latency counts.

An fma instruction has a latency of 5 if the output goes to another fma instruction, 6 if the output goes to an fp instruction, and 6+1 if the output goes to an ivec or

store instruction.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions						
MOV	r8,r8	1	1	0.5	EX01	

MOV	r16,r16	1	1	0.5	EX01	
MOV	r32,r32	1	1	0.25	EX01 or AG01	
MOV	r64,r64	1	1	0.25	EX01 or AG01	
MOV	r,i	1	1	0.5	EX01	
MOV		1	3	0.5	AG01	all addr. modes
	r,m		4			
MOV	m,r	1	4	1	EX01 AG01	all addr. modes
MOV	m,i	1		1		
MOVNTI	m,r	1	4	1		
MOVZX, MOVSX	r,r	1	1	0.5	EX01	
MOVSX	r,m	1	5	0.5	EX01	
MOVZX	r,m	1	4	0.5	EX01	
MOVSXD	r64,r32	1	1	0.5	EX01	
MOVSXD	r64,m32	1	5	0.5	EX01	
CMOVcc	r,r	1	1	0.5	EX01	
CMOVcc	r,m	1		0.5	EX01	
XCHG	r8,r8	2	1	1	EX01	
XCHG	r16,r16	2	1	1	EX01	
XCHG		2	-	0.5	EX01	
	r32,r32	2	1			
XCHG	r64,r64		1	0.5	EX01	.
XCHG	r,m	2	~38	~38	EX01	Timing depends on hw
XLAT	1,111	2	6	2	LXUI	IIVV
	_		O			
PUSH	r	1		1		
PUSH	i	1		1		
PUSH	m	2		1		
PUSHF(D/Q)		8		4		
PUSHA(D)		9		9		
POP	r	1		1		
POP	m	2		1		
POPF(D/Q)		34		19		
POPA(D)		14		8		
POP	sp	1	2			
LEA	r16,[m]	2	2-3		EX01	any addr. size
LEA	r32,[m]	1	2		EX01	16 bit addr. size
	102,[111]	'			LXOT	
LEA	r32/64,[m]	1	2	0.5	EX01	scale factor > 1 or 3 operands
LEA	r32/64,[m]	1	1	0.5	EX01	all other cases
LAHF	152/04,[11]	4	3	2	LXUI	all Other Cases
SAHF						
		2	2	1		
SALC		1	1	1		
BSWAP	r	1	1	0.5	EX01	
PREFETCHNTA	m	1		0.5		
PREFETCHT0/1/2	m	1		0.5		
PREFETCH/W	m	1		0.5		PREFETCHW
SFENCE		7		~80		
LFENCE		1		0.25		
MFENCE		7		~80		
A with we attend to a town att						
Arithmetic instructions		4	4	0.5	EV04	
ADD, SUB	r,r	1	1	0.5	EX01	
ADD, SUB	r,i	1	1	0.5	EX01	
ADD, SUB	r,m	1		0.5	EX01	
ADD, SUB	m,r	1	7	1	EX01	

		•	otournono	1		
ADD, SUB	m,i	1	7	1	EX01	
ADC, SBB	r,r	1	1		EX01	
ADC, SBB	r,i	1	1		EX01	
ADC, SBB	r,m	1	1	1	EX01	
ADC, SBB	m,r	1	9	1	EX01	
ADC, SBB	m,i	1	9	1	EX01	
CMP	r,r	1	1	0.5	EX01	
CMP	r,i	1	1	0.5	EX01	
CMP	r,m	1		0.5	EX01	
CMP	m,i	1		0.5	EX01	
INC, DEC, NEG	r	1	1	0.5	EX01	
INC, DEC, NEG	m	1	7	1	EX01	
AAA, AAS		10	6		LXOI	
DAA		16	8			
DAS		20	10			
AAD		4	6			
AAM		10	15	15		
MUL, IMUL	r8/m8	1	4	2	EX1	
MUL, IMUL	r16/m16	2	4	2	EX1	
MUL, IMUL	r32/m32	1	4	2	EX1	
MUL, IMUL	r64/m64	1	6	4	EX1	
IMUL	r16,r16/m16	1	4	2	EX1	
IMUL	r32,r32/m32	1	4	2	EX1	
IMUL	r64,r64/m64	1	6	4	EX1	
IMUL	r16,(r16),i	2	5	2	EX1	
IMUL	, ,	1	4	2	EX1	
IMUL	r32,(r32),i	1	6	4	EX1	
	r64,(r64),i		O			
IMUL	r16,m16,i	2		2	EX1	
IMUL	r32,m32,i	2 2		2 4	EX1	
IMUL DIV	r64,m64,i r8/m8	9	17-22	13-17	EX1 EX0	
		7	15-25			
DIV	r16/m16			15-25	EX0	
DIV	r32/m32	2 2	13-39	13-39	EX0	
DIV	r64/m64		13-70	13-70	EX0	
IDIV	r8/m8	9	17-22	13-17 14-24	EX0	
IDIV	r16/m16	7	14-25		EX0	
IDIV	r32/m32	2	13-39	13-39	EX0	
IDIV	r64/m64	2	13-70	13-70	EX0	
CBW, CWDE, CDQE		1	1	0.5	EX01	
CDQ, CQO		1 2	1	0.5	EX01	
CWD			1	1	EX01	
Logic instructions						
AND, OR, XOR	r,r	1	1	0.5	EX01	
AND, OR, XOR	r,i	1	1	0.5	EX01	
AND, OR, XOR	r,m	1		0.5	EX01	
AND, OR, XOR	m,r	1	7	1	EX01	
AND, OR, XOR	m,i	1	7		EX01	
TEST	r,r	1	1	0.5	EX01	
TEST	r,i	1	1	0.5	EX01	
TEST	m,r	1	'	0.5	EX01	
TEST	m,i	1		0.5	EX01	
NOT	r	1	1	0.5	EX01	
11401	, ·	'		0.5	LAUI	

NOT		4	7	4	EV04	
NOT	m	1	7	1	EX01	DMIA
ANDN	r,r,r	1	1	0.5	EX01	BMI1
SHL, SHR, SAR	r,i/CL	1	1	0.5	EX01	
ROL, ROR	r,i/CL	1	1	0.5	EX01	
RCL	r,1	1	1		EX01	
RCL	r,i	16	7		EX01	
RCL	r,cl	17	7		EX01	
RCR	r,1	1	1		EX01	
RCR	r,i	15	7		EX01	
RCR	r,cl	16	7		EX01	
SHLD, SHRD	r,r,i	6	3	3	EX01	
SHLD, SHRD	r,r,cl	7-8	4	4	EX01	
SHLD, SHRD	m,r,i/CL	8		4	EX01	
BT	r,r/i	1	1	0.5	EX01	
BT	m,i	1		0.5	EX01	
BT	m,r	7		3.5	EX01	
BTC, BTR, BTS	r,r/i	2	2	1	EX01	
BTC, BTR, BTS	m,i	4		2	EX01	
BTC, BTR, BTS	m,r	10		5	EX01	
BSF	r,r	6	3	3	EX01	
BSF	r,m	8	4	4	EX01	
BSR	r,r	7	4	4	EX01	
BSR	r,m	9		5	EX01	
SETcc	r	1	1	0.5	EX01	
SETcc	m	1	-	1	EX01	
CLC, STC		1		0.5	EX01	
CMC		1	1		EX01	
CLD		2		3	2701	
STD		2		4		
POPCNT	r16/32,r16/32	1	4	2		SSE4.2
POPCNT	r64,r64	1	4	4		SSE4.2
LZCNT	r,r	1	2	2	EX0	LZCNT
TZCNT	r,r	2	2	2	LXO	BMI1
BEXTR	r,r,r	2	2	1		BMI1
BEXTR	r,r,i	2	2	1		AMD TBM
BLSI	r,r	2	2			BMI1
BLSMSK	r,r	2	2	1		BMI1
BLSR	r,r	2	2	1		BMI1
BLCFILL	r,r	2	2	1		AMD TBM
BLCI	r,r	2	2	1		AMD TBM
BLCIC	r,r	2	2	1		AMD TBM
BLCMSK	r,r	2	2	1		AMD TBM
BLCS	r,r	2	2	1		AMD TBM
BLSFILL		2	2	1		AMD TBM
BLSI	r,r	2	2	1		AMD TBM
BLSIC	r,r	2	2	1		AMD TBM
T1MSKC	r,r	2	2	1		AMD TBM
TZMSK	r,r	2	2	1		AMD TBM
ILIVION	r,r	2		'		AIVID I DIVI
Control transfer instru	ctions					
JMP	short/near	1		2	EX1	
JMP	r	1		2	EX1	
JMP	m	1		2	EX1	

		_	riourin one	•		
Jcc	short/near	1 1		1-2	EX1	2 if jumping
fused CMP+Jcc	short/near	1		1-2	EX1	2 if jumping
J(E/R)CXZ	short	1		1-2	EX1	2 if jumping
LOOP	short	1		1-2	EX1	2 if jumping
LOOPE LOOPNE	short	1		1-2	EX1	2 if jumping
CALL	near	2		2	EX1	2 japg
CALL	r	2		2	EX1	
CALL	m	3		2	EX1	
RET	""	1		2	EX1	
RET	i	4		2	EX1	
BOUND	m	11		5		for no jump
INTO	111	4		2		
INTO		4				for no jump
String in atmostic as						
String instructions				2		
LODS	0/ 10	3		3		
REP LODS	m8/m16	6n		3n		
REP LODS	m32/m64	6n		2.5n		
STOS		3		3		
REP STOS		1n		~1n		small n
REP STOS		3 per 16B		2 per 16B		best case
MOVS		5		3		
REP MOVS		~1n		~1n		small n
REP MOVS		4-5 pr 16B		~2 per 16B		best case
SCAS		3		3		
REP SCAS		7n		3-4n		
CMPS		6		3		
REP CMPS		9n		4n		
Synchronization						
LOCK ADD	m,r	1	~39			
XADD	m,r	4	9-12			
LOCK XADD	m,r	4	~39			
CMPXCHG	m,r8	5	15			
CMPXCHG	m,r16	6	15			
CMPXCHG	m,r32/64	6	13			
LOCK CMPXCHG	m8,r8	5	~40			
LOCK CMPXCHG	m16,r16	6	~40			
LOCK CMPXCHG	m,r32/64	6	~40			
CMPXCHG8B	m64	18	~14			
LOCK CMPXCHG8B	m64	18	~42			
CMPXCHG16B	m128	24	~47			
LOCK CMPXCHG16B	m128	24	~80			
2001(011117(011010)	20		00			
Other						
NOP (90)		1		0.25	none	
Long NOP (0F 1F)		1		0.25	none	
PAUSE		8		4	HOHG	
ENTER	a,0	13		21		
ENTER	a,b	11+5b		20-30		
LEAVE	a,u	2		3		
CPUID		38-64		100-300		
XGETBV		4		30		
RDTSC		44		78		

RDTSCP		44		105	rdtscp	
RDPMC		22		360		
CRC32	r32,r8	3	3	2		
CRC32	r32,r16	5	5	5		
CRC32	r32,r32	7	6	6		

Floating point x87 instructions

Floating point x87	instructions	3				
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes
Move instructions	ll.				• •	
FLD	r	1	2	0.5	P01	fp
FLD	m32/64	1	7	1		fp
FLD	m80	8	11	4		fp
FBLD	m80	60	52	34	P0 P1 P2	fp
FST(P)	r	1	2	0.5	P01	fp
FST(P)	m32/64	2	7	1		fp
FSTP	m80	13	14	19		fp
FBSTP	m80	239	222	222	P0 P1 P2	fp
FXCH	r	1	0	0.5	P01	inherit
FILD	m	1	11	1	P01	fp
FIST(T)(P)	m	2	7	1	P0 P2	fp
FLDZ, FLD1		1		0.5	P01	fp
FCMOVcc	st0,r	8	3	3	P0 P1 P2	fp
FFREE	r	1		0.25	none	•
FINCSTP, FDECSTP		1	0	0.25	none	inherit
FNSTSW	AX	3	11	19	P0 P2	
FNSTSW	m16	2		17	P0 P2	
FLDCW	m16	1		3	-	
FNSTCW	m16	2		2		
Arithmetic instructions	 S					
FADD(P),FSUB(R)(P)	r/m	1	5	1	P01	fma
FIADD,FISUB(R)	m	2		2	P01	fma
FMUL(P)	r/m	1	5	1	P01	fma
FIMUL	m	2		2	P01	fma
FDIV(R)(P)	r	1	9-37	4-16	P01	fp
FDIV(R)	m	1			P01	fp
FIDIV(R)	m	2		4	P01	fp
FABS, FCHS		1	2	0.5	P01	fp
FCOM(P), FUCOM(P)	r/m	1		0.5	P01	fp
FCOMPP, FUCOMPP		1		0.5	P01	fp
FCOMI(P)	r	2	2	1	P01 P2	fp
FICOM(P)	m	2		1	P01	fp
FTST		1		0.5	P01	fp
FXAM		1	26	0.5	P01	fp
FRNDINT		1	4	1	P0	fp
FPREM FPREM1		1	17-60	12-53	P0	fp
Math						
FSQRT		1	10-50	5-20	P01	
FLDPI, etc.		1		0.5	P01	
FSIN		10-164	60-210	60-165	P0 P1 P2	

FCOS		18-166	76-158		P0 P1 P2	
FSINCOS		12-168		90-165	P0 P1 P2	
FPTAN		11-192	90-245	90-210	P0 P1 P2	
FPATAN		10-365	60-440	60-365	P0 P1 P2	
FSCALE		10	49	5	P0 P1 P2	
FXTRACT		12	8	5	P0 P1 P2	
F2XM1		10-18	60-74		P0 P1 P2	
FYL2X		9-183	60-280		P0 P1 P2	
FYL2XP1		206	~390		P0 P1 P2	
Other						
FNOP		1		0.25	none	
(F)WAIT		1		0.25	none	
FNCLEX		18		63	P0	
FNINIT		31		131	P0	
FNSAVE	m864	98	256	256	P0 P1 P2	
FRSTOR	m864	73	166	166	P0 P2	

Integer vector instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions						
MOVD	r32/64, mm/x	1	8	1	P2	
MOVD	mm/x, r32/64	2	9	1		
MOVD	mm/x,m32	1	2	0.5		
MOVD	m32,mm/x	1	3	1		
MOVQ	mm/x,mm/x	1	2	0.5	P02	
MOVQ	mm/x,m64	1	2	0.5		
MOVQ	m64,mm/x	1	3	1		
MOVDQA	X,X	1	0	0.25	none	inherit domain
MOVDQA	x,m	1	6	0.5		
MOVDQA	m,x	1	5	1	P2	
VMOVDQA	y,y	2	6	0.5	P02	
VMOVDQA	y,m256	2	5	1		
VMOVDQA	m256,y	2	4	1	P2	
MOVDQU	X,X	1	0	0.25	none	inherit domain
MOVDQU	x,m	1	6	0.5		
MOVDQU	m,x	1	5	1	P2	
LDDQU	x,m	1	6	0.5		
VMOVDQU	y,m256	2	7	1		
VMOVDQU	m256,y	2	6	1		
MOVDQ2Q	mm,x	1	1	0.5	P02	
MOVQ2DQ	x,mm	1	1	0.5	P02	
MOVNTQ	m,mm	1	5	1	P2	
MOVNTDQ	m,x	1	5	1	P2	
MOVNTDQA	x,m	1	2	0.5		
PACKSSWB/DW	mm/x,r/m	1	2	1	P1	
PACKUSWB	mm/x,r/m	1	2	1	P1	
PUNPCKH/LBW/WD/						
DQ	mm/x,r/m	1	2	1 1	P1	
PUNPCKHQDQ	x,r/m	1	2	1	P1	

		`	otourn ono	•		
PUNPCKLQDQ	x,r/m	1	2	1	P1	
PSHUFB	mm/x,r/m	1	3	1	P1	
PSHUFD	x,x,i	1	2	1	P1	
PSHUFW	mm,mm,i	1	2	1	P1	
PSHUFL/HW	x,x,i	1	2	1	P1	
PALIGNR	mm/x,r/m,i	1	2	1	P1	
PBLENDW	x,r/m	1	2	0.5	P02	SSE4.1
MASKMOVQ	mm,mm	31	32	16	P2	0021
MASKMOVDQU	X,X	65	45	31	P0 P1 P2	
PMOVMSKB	r32,mm/x	2	5	1	P1 P2	
PEXTRB/W/D/Q	r,x/mm,i	2	5	1	P1 P2	SSE4.1
PINSRB/W/D/Q	x/mm,r,i	2	6	1	P1	33L4.1
EXTRQ		1	3	1	P1	AMD SSE4A
EXTRQ	x,i,i		1	1	P1	AMD SSE4A
	X,X	1		·		AMD SSE4A
INSERTQ	x,x,i,i	1	1	1	P1	
INSERTQ	x,x	1	1	1	P1	AMD SSE4A
PMOVSXBW/BD/BQ/		4	2	4	D4	00544
WD/WQ/DQ	x,x	1	2	1	P1	SSE4.1
PMOVZXBW/BD/BQ/	V V	1	2	1	P1	SSE4.1
WD/WQ/DQ	X,X					
VPCMOV	x,x,x,x/m	1	2	1	P1	AMD XOP
VPCMOV	y,y,y,y/m	2	2	2	P1	AMD XOP
VPPERM	x,x,x,x/m	1	2	1	P1	AMD XOP
A with we atic impatument and						
Arithmetic instructions	5					
PADDB/W/D/Q/SB/SW/ USB/USW	mm/x,r/m	1	2	0.5	P02	
PSUBB/W/D/Q/SB/SW/	11111/7,1/111	I I		0.5	1 02	
USB/USW	mm/x,r/m	1	2	0.5	P02	
PHADD/SUB(S)W/D	X,X	3	5	2	P02 2P1	SSSE3
PCMPEQ/GT B/W/D	mm/x,r/m	1	2	0.5	P02	33320
PCMPEQQ	mm/x,r/m	1	2	0.5	P02	SSE4.1
PCMPGTQ	mm/x,r/m	1	2	0.5	P02	SSE4.2
PMULLW PMULHW	11111/77,17111		_	0.0	1 02	OOL4.2
PMULHUW PMULUDQ	mm/x,r/m	1	4	1	P0	
PMULLD	x,r/m	1	5	2	P0	SSE4.1
PMULDQ	x,r/m	1	4	_ 1	P0	SSE4.1
PMULHRSW	mm/x,r/m	1	4	1	P0	SSSE3
PMADDWD	mm/x,r/m	1	4	1	P0	33320
PMADDUBSW	mm/x,r/m	1	4	1	P0	
PAVGB/W	mm/x,r/m	1	2	0.5	P02	
PMIN/MAX SB/SW/ SD	11111/7,1/111	'		0.0	1 02	
UB/UW/UD	mm/x,r/m	1	2	0.5	P02	
PHMINPOSUW	x,r/m	2	4	1	P1 P02	SSE4.1
PABSB/W/D	mm/x,r/m	1	2	0.5	P02	SSSE3
PSIGNB/W/D	mm/x,r/m	1	2	0.5	P02	SSSE3
PSADBW	mm/x,r/m	2	4	1	P02	COOLS
MPSADBW	x,x,i	8	8	4	P1 P02	SSE4.1
IVII ONDOVV	^,^,1			7	1 11 02	AMD XOP
VPCOMB/W/D/Q	x,x,x/m,i	1	2	0.5	P02	latency 0 if i=6,7
	, ,,-					AMD XOP
VPCOMUB/W/D/Q VPHADDBW/BD/BQ/	x,x,x/m,i	1	2	0.5	P02	latency 0 if i=6,7
WD/WQ/DQ	x,x/m	1	2	0.5	P02	AMD XOP
		•		'	T. Control of the Con	, 1

VDLIA DDLIDVA/DD/DO/	l	I	I	I	1	I
VPHADDUBW/BD/BQ/ WD/WQ/DQ	x,x/m	1	2	0.5	P02	AMD XOP
VPHSUBBW/WD/DQ	x,x/m	1	2	0.5	P02	AMD XOP
VPMACSWW/WD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSDD	x,x,x/m,x	1	5	2	P0	AMD XOP
VPMACSDD VPMACSDQH/L		1	4	1	P0	AMD XOP
VPMACSDQH/L VPMACSSWW/WD	x,x,x/m,x	1		1	P0 P0	AMD XOP
VPMACSSWW/WD	x,x,x/m,x	1	4 5	2	P0 P0	AMD XOP
	x,x,x/m,x		4	1	P0 P0	
VPMACSSDQH/L	x,x,x/m,x	1	4			AMD XOP
VPMADCSWD	x,x,x/m,x	1		1	P0	AMD XOP
VPMADCSSWD	x,x,x/m,x	1	4	1	P0	AMD XOP
Logio						
Logic PAND PANDN POR						
PXOR	mm/x,r/m	1	2	0.5	P02	
PSLL/RL W/D/Q	11111/7,1/111	'		0.0	1 02	
PSRAW/D	mm/x,r/m	1	3	1	P1	
PSLL/RL W/D/Q	,,,,,,,					
PSRAW/D	mm/x,i	1	2	1	P1	
PSLLDQ, PSRLDQ	x,i	1	2	1	P1	
PTEST	x,r/m	2	14	1	P1 P2	SSE4.1
VPROTB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
VPROTB/W/D/Q	x,x,i	1	2	1	P1	AMD XOP
VPSHAB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
VPSHLB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
	, , ,					
String instructions						
PCMPESTRI	x,x,i	30	11	11	P0 P1 P2	SSE4.2
PCMPESTRM	x,x,i	30	10	10	P0 P1 P2	SSE4.2
PCMPISTRI	x,x,i	9	5	5	P0 P1 P2	SSE4.2
PCMPISTRM	x,x,i	8	6	6	P0 P1 P2	SSE4.2
Encryption						
PCLMULQDQ	x,x/m,i	7	11	7	P1	pclmul
VPCLMULQDQ	x,x,x,i	7	11	7	P1	pclmul
PCLMULQDQ	x,x,m,i	8		7	P1	pclmul
AESDEC	x,x	2	5	1	P01	aes
AESDECLAST	x,x	2	5	1	P01	aes
AESENC	x,x	2	5	1	P01	aes
AESENCLAST	x,x	2	5	1	P01	aes
AESIMC	x,x	1	5	1	P0	aes
AESKEYGENASSIST	x,x,i	1	5	1	P0	aes
Other						
EMMS		1		0.25		

Floating point vector instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes
Move instructions						
MOVAPS/D MOVUPS/						
D	X,X	1	0	0.25	none	inherit domain

VMOVAPS/D	y,y	2	2	0.5	P02	ivec
MOVAPS/D MOVUPS/	x,m128	1	6	0.5		
VMOVAPS/D	X,111120	'		0.0		
VMOVUPS/D	y,m256	2	6	1		
MOVAPS/D MOVUPS/						
D	m128,x	1	5	1	P2	
VMOVAPS/D	m256,y	2	5	2	P2	
VMOVUPS/D	m256,y	2	5	2	P2	
MOVSS/D	X,X	1	2	0.5	P01	fp
MOVSS/D	x,m32/64	1	6	0.5		
MOVSS/D	m32/64,x	1	5	1	P2	
MOVHPS/D	x,m64	1	6	1	P1	
MOVLPS/D	x,m64	1	6	0.5	P01	
MOVHPS/D	m64,x	2	6	1	P1 P2	
MOVLPS/D	m64,x	1	5	1	P2	
MOVLHPS MOVHLPS	X,X	1	2	1	P1	ivec
MOVMSKPS/D	r32,x	2	5	1	P1 P2	
VMOVMSKPS/D	r32,y	2	15	1	P1 P2	
MOVNTPS/D	m128,x	1	3	1	P2	
VMOVNTPS/D	m256,y	2	3	2-3	P2	
MOVNTSS/SD	m,x	1		3	P2	AMD SSE4A
SHUFPS/D	x,x/m,i	1	2	1	P2	ivec
VSHUFPS/D	y,y,y/m,i	2	2	2	P2	ivec
VPERMILPS/PD	x,x,x/m	1	3	1	P1	ivec
VPERMILPS/PD	y,y,y/m	2	3	2	P1	ivec
VPERMILPS/PD	x,x/m,i	1	2	1	P1	ivec
VPERMILPS/PD	y,y/m,i	2	2	2	P1	ivec
VPERM2F128	y,y,y,i	8	4	3.5	P0 P2	ivec
VPERM2F128	y,y,m,i	12		4	P0 P2	ivec
BLENDPS/PD	x,x/m,i	1	2	0.5	P01	fp
VBLENDPS/PD	y,y,y/m,i	2	2	1	P01	fp
BLENDVPS/PD	x,x/m,xmm0	1	2	0.5	P01	
VBLENDVPS/PD	y,y,y/m,y	2	2	1	P01	
MOVDDUP	X,X	1	2	1	P1	ivec
MOVDDUP	x,m64	1		0.5		
VMOVDDUP	y,y	2	2	2	P1	ivec
VMOVDDUP	y,m256	2		1		
VBROADCASTSS	x,m32	1	8	0.5		
VBROADCASTSS	y,m32	2	8	0.5	P02	
VBROADCASTSD	y,m64	2	8	0.5	P02	
VBROADCASTF128	y,m128	2	8	0.5	P02	
MOVSH/LDUP	X,X	1	2	1	P1	ivec
MOVSH/LDUP	x,m128	1		0.5		
VMOVSH/LDUP	y,y	2	2	2	P1	ivec
VMOVSH/LDUP	y,m256	2		1		
UNPCKH/LPS/D	x,x/m	1	2	1	P1	ivec
VUNPCKH/LPS/D	y,y,y/m	2	2	2	P1	ivec
EXTRACTPS	r32,x,i	2		1	P1 P2	
EXTRACTPS	m32,x,i	2	10	1	P1 P2	
VEXTRACTF128	x,y,i	1	2	0.5	P02	ivec
VEXTRACTF128	m128,y,i	2	10	1	P0 P2	
INSERTPS	x,x,i	1	2	1	P1	

INSERTPS	x,m32,i	1	9	2	P1	
VINSERTF128	y,y,x,i	2	2	1	P02	ivec
VINSERTF128	y,y,m128,i	2	10	1	P02	1700
VMASKMOVPS/D	x,x,m128	1	9	0.5	P01	
VMASKMOVPS/D	y,y,m256	2	9	1	P01	
VMASKMOVPS/D	m128,x,x	20	~35	8	P0 P1 P2	
VMASKMOVPS/D	m256,y,y	41	~35	16	P0 P1 P2	
VIVIAGRIVIO VP 3/D	111250,y,y	41	~33	10	PUP1P2	
Conversion						
CVTPD2PS	x,x	2	6	1	P01	ivec/fp
VCVTPD2PS	x,y	4	6	2	P01	ivec/fp
CVTPS2PD	x,x	2	6	1	P01	ivec/fp
VCVTPS2PD	y,x	4	6	2	P01	ivec/fp
CVTSD2SS	x,x	1	4	1	P0	fp
CVTSS2SD	x,x	1	4	1	P0	fp
CVTDQ2PS	x,x	1	4	1	P0	fp
VCVTDQ2PS	у,у	2	4	2	P0	fp
CVT(T) PS2DQ	x,x	1	4	_ 1	P0	fp
VCVT(T) PS2DQ	у,у	2	4	2	P0	fp
CVTDQ2PD	x,x	2	7	1	P01	ivec/fp
VCVTDQ2PD	y,x	4	7	2	P01	ivec/fp
CVT(T)PD2DQ	x,x	2	7	1	P01	fp/ivec
VCVT(T)PD2DQ	x,y	4	7	2	P01	fp/ivec
CVTPI2PS	x,mm	2	6	1	P0 P2	ivec/fp
CVT(T)PS2PI	mm,x	1	5	1	P0	fp
CVTPI2PD	x,mm	2	7	1	P0 P1	ivec/fp
CVT(T) PD2PI	mm,x	2	7	1	P0 P1	fp/ivec
CVTSI2SS	x,r32	2	13	1	P0	fp
CVT(T)SS2SI	r32,x	2	12	1	P0 P2	fp
CVTSI2SD	x,r32/64	2	12	1	P0	fp
CVT(T)SD2SI	r32/64,x	2	12	1	P0 P2	fp
VCVTPS2PH	x/m,x,i	2	7	2	P0 P1	F16C
VCVTPS2PH	x/m,x,i	4	7	2	P0 P1	F16C
VCVTPH2PS	x,x/m	2	7	2	P0 P1	F16C
VCVTPH2PS	y,x/m	4	7	2	P0 P1	F16C
V O V 11 1121 O	y,,,,,,,,,,	_	,	_	1011	1 100
Arithmetic						
ADDSS/D SUBSS/D	x,x/m	1	5-6	1	P01	fma
ADDPS/D SUBPS/D	x,x/m	1	5-6	1	P01	fma
VADDPS/D VSUBPS/D	y,y,y/m	2	5-6	2	P01	fma
ADDSUBPS/D	x,x/m	1	5-6	1	P01	fma
VADDSUBPS/D	y,y,y/m	2	5-6	1	P01	fma
VADDOODI O/D	y,y,y/111		3-0	ı	101	IIIIa
HADDPS/D HSUBPS/D	x,x	4	10	2	P0 P1	ivec/fma
VHADDPS/D			4.5	_	D0 / T /	
VHSUBPS/D	y,y,y/m	8	10	4	P01 P1	ivec/fma
MULSS MULSD	x,x/m	1	5-6	0.5	P01	fma
MULPS MULPD	x,x/m	1	5-6	0.5	P01	fma
VMULPS VMULPD	y,y,y/m	2	5-6	1	P01	fma
DIVSS DIVPS	x,x/m	1	9-17	4-6	P01	fp
VDIVPS	y,y,y/m	2	9-17	9-12	P01	fp
DIVSD DIVPD	x,x/m	1	9-32	4-13	P01	fp

VDI/PD			`	otourn one	•		
RCPSS/PS	VDIVPD	y,y,y/m	2	9-32	9-27	P01	fp
NRCPPS	RCPSS/PS			5	1	P01	
CMPS/ID		·			2		
CMPPS/ID		<i>y</i> , <i>y</i> ,	_		_		
VAMPPS/ID		x.x/m	1	2	0.5	P01	fp
COMISS/D UCOMISS/D		•					
D		y,y,y/111			•	101	ıρ
MAXSS/SD/PS/PD x,x/m 1 2 0.5 P01 fp VMAXPS/D VMINPS/D ROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD x,x/m,i 1 4 1 P0 fp DPPS VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD y,y/m,i 2 4 2 P0 fp DPPS VROUNDSS/SD/PS/PD VROUNDSS/SD/PS x,x,i 9 25 4 P0 P1 SSE4.1 DPPS VROUNDSS/SD/PS x,x,i 10 5 P0 P1 SSE4.1 DPPS VROUNDSS/SD/PS y,y,i 13 25 8 P0 P1 SSE4.1 DPPD VROUNDSS/SD/PS x,x,i 1 5-6 0.5 P0 P1 SSE4.1 DPPD x,x,i 1 5-6 0.5 P0 P1 SSE4.1 DPPD x,x,i/m 1 5-6 0.5 P0 P1 SSE4.1 DPPD x,x,x/m 1 5-6 0.5 P0 P1 FMA3 VFMADD132PS/PD VFMADDSS/SD x,x,x/m 1 5-6 0		y y/m	2		1	P01 P2	fn
MINSS/SD/PS/PD	-	λ,λ/111			•	10112	ıρ
VMAXPS/ID VMINPS/D VMAXPS/ID VMINPS/D ROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VROUNDSS/SD/PS/PD VRAIN 1 2 4 2 P0 fp PD S, x,xi, 9 25 4 P0 P1 SSE4.1 DPPS x,xi, 10 5 P0 P1 SSE4.1 VDPPS y,y,y, 13 25 8 P0 P1 SSE4.1 VDPPS y,m,i 15 8 P0 P1 SSE4.1 VDPPS y,m,i 15 8 P0 P1 SSE4.1 DPPD x,x,i 7 14 3 P0 P1 SSE4.1 DPPD x,x,i 7 14 3 P0 P1 SSE4.1 VFMADD132SS/SD x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADD132PS/PD x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADD132PS/PD x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADD132PS/PD x,x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADDSS/SD x,x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADDSS/SD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD y,y,y/m 2 5-6 1 P01 AMD FMA4 VFMADDPS/PD y,y,y/m 2 5-6 1 P01 AMD FMA4 VFMADDPS/PD y,y,y/m 2 5-6 1 P01 fp VSQRTPD y,y/m 2 12-13 9-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp VSQRTPD y,y/m 2 5 2 P01 fp VRSQRTPD y,y/m 2 5 2 P01 fp VRSQRTPS y,y/m 2 5 2 P01 fp VRSQRTPD y,y/m 2 5 2 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD y,y/m 2 5 2 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 5 5 2 P01 fp VRSQRTPD x,x/m 1 5 5 2 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 5 5 2 P01 fp VRSQRTPD x,x/m 1 5 5 2 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 5 5 1 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01 fp VRSQRTPD x,x/m 1 26-29 4-18 P01		y y/m	1	2	0.5	P ∩1	fn
ROUNDSS/SD/PS/PD	WIINGG/GD/I G/I D	۸,۸/۱۱۱	1		0.5	101	ıρ
ROUNDSS/SD/PS/PD	VMAXPS/D VMINPS/D	v v v/m	2	2	1	P01	fn
VROUNDSS/SD/PS/PD y,y/m,i 2 4 2 PO fp DPPS x,x,i 9 25 4 PO P1 SSE4.1 DPPS x,m,i 10 5 PO P1 SSE4.1 VDPPS y,y,y,i 13 25 8 PO P1 SSE4.1 VDPPS y,y,y,i 13 25 8 PO P1 SSE4.1 DPPD x,x,i 7 14 3 PO P1 SSE4.1 DPPD x,x,i 7 14 3 PO P1 SSE4.1 DPPD x,x,i 1 5-6 0.5 P01 FMA3 VFMADD132PS/PD x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADD2S/PD x,x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD x,x/x,x/m 1 5-6 0.5 P01							
PD		۸,۸/۱۱۱,۱	'	7	'	10	ip ip
DPPS		v v/m i	2	4	2	PΛ	fn
DPPS							
VDPPS				25			
VDPPS				0.5			
DPPD				25			
DPPD		=					
VFMADD132SS/SD x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADD132PS/PD x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADD132PS/PD y,y,y/m 2 5-6 1 P01 FMA3 All other FMA3 instructions: same as above VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD x,x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD x,x,x/m 1 12-13 4-9 P01 fp Math SQRTSS/PS x,x/m 1 12-13 4-9 P01 fp VSQRTPB y,y/m 2 12-13 9-18 P01 fp VSQRTSD/PD x,x/m 1 26-29 4-18 P01 fp				14			
VFMADD132PS/PD x,x,x/m 1 5-6 0.5 P01 FMA3 VFMADD132PS/PD y,y,y/m 2 5-6 1 P01 FMA3 All other FMA3 instructions: same as above VFMADDSS/SD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD y,y,y/m 2 5-6 1 P01 AMD FMA4 VFMADDPS/PD y,y,y/m 2 5-6 1 P01 AMD FMA4 VFMADDPS/PD y,y,y/m 2 5-6 1 P01 AMD FMA4 VFMADDPS/PD y,y,y/m 2 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp VSQRTPD y,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp VFRCZSS/SD/PS/PD		x,m,i	8				
VFMADD132PS/PD y,y,y/m 2 5-6 1 P01 FMA3 All other FMA3 instructions: same as above VFMADDSS/SD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD y,y,y,y/m 2 5-6 1 P01 AMD FMA4 VFMADDPS/PD y,y,y,y/m 2 5-6 1 P01 AMD FMA4 VFMADDPS/PD y,y,y,y/m 2 5-6 1 P01 AMD FMA4 VFMADDPS/PD y,y,y/m 2 5-6 1 P01 AMD FMA4 Math SQRTSS/PS x,x/m 1 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp VSQRTPD y,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp <tr< td=""><td>VFMADD132SS/SD</td><td>x,x,x/m</td><td>1</td><td>5-6</td><td>0.5</td><td>P01</td><td>FMA3</td></tr<>	VFMADD132SS/SD	x,x,x/m	1	5-6	0.5	P01	FMA3
All other FMA3 instructions: same as above VFMADDSS/SD	VFMADD132PS/PD	x,x,x/m	1	5-6	0.5	P01	FMA3
VFMADDSS/SD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD y,y,y,y/m 2 5-6 1 P01 AMD FMA4 All other FMA4 instructions: same as above AMD FMA4 AMD FMA4 Math SQRTSS/PS x,x/m 1 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp VSQRTPD y,y/m 2 12-13 9-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp VSQRTSS/PS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP Logic AND/ANDN/OR/XORPS/PD y,y,y/m 2 2	VFMADD132PS/PD	y,y,y/m	2	5-6	1	P01	FMA3
VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD y,y,y,y/m 2 5-6 1 P01 AMD FMA4 All other FMA4 instructions: same as above AMD FMA4 AMD FMA4 Math SQRTSS/PS x,x/m 1 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp VSQRTSD/PD x,x/m 1 26-29 4-18 P01 fp VSQRTSS/PD y,y/m 2 27-28 9-37 P01 fp VRSQRTSS/PS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP Logic AND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 <th< td=""><td>All other FMA3 instruction</td><td>ns: same as a</td><td>bove</td><td>1</td><td></td><td></td><td>FMA3</td></th<>	All other FMA3 instruction	ns: same as a	bove	1			FMA3
VFMADDPS/PD x,x,x,x/m 1 5-6 0.5 P01 AMD FMA4 VFMADDPS/PD y,y,y,y/m 2 5-6 1 P01 AMD FMA4 All other FMA4 instructions: same as above Math P01 AMD FMA4 Math SQRTSS/PS x,x/m 1 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp SQRTSD/PD x,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp VRSQRTSS/PS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP Logic AND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 <td>VFMADDSS/SD</td> <td>x,x,x,x/m</td> <td>1</td> <td>5-6</td> <td>0.5</td> <td>P01</td> <td>AMD FMA4</td>	VFMADDSS/SD	x,x,x,x/m	1	5-6	0.5	P01	AMD FMA4
VFMADDPS/PD y,y,y,y/m 2 5-6 1 P01 AMD FMA4 All other FMA4 instructions: same as above Math X,x/m 1 12-13 4-9 P01 fp SQRTSS/PS x,x/m 1 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp SQRTSD/PD x,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp VRSQRTPS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 fp VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m			1	5-6		P01	
All other FMA4 instructions: same as above Math SQRTSS/PS X,x/m			2				
Math x,x/m 1 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp SQRTSD/PD x,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp VSQRTSS/PS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 fp VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP Logic X,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec <td></td> <td></td> <td></td> <td></td> <td>•</td> <td></td> <td></td>					•		
SQRTSS/PS x,x/m 1 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp SQRTSD/PD x,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp VRSQRTPS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 fp VRSQRTPS y,y/m 2 5 2 P01 fp VRSQRTPS y,y/m 2 5 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP Logic x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode 64 bit mode	, an outer 1 white instruction			1			7 WID I WIX CT
SQRTSS/PS x,x/m 1 12-13 4-9 P01 fp VSQRTPS y,y/m 2 12-13 9-18 P01 fp SQRTSD/PD x,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp VRSQRTPS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 fp VRSQRTPS y,y/m 2 5 2 P01 fp VRSQRTPS y,y/m 2 5 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP Logic x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode 64 bit mode	Math						
VSQRTPS y,y/m 2 12-13 9-18 P01 fp SQRTSD/PD x,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp RSQRTSS/PS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x/m 4 2 P01 AMD XOP Logic x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode 64 bit mode VZEROUPPER 16 5 64 bit mode 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32		x x/m	1	12-13	4-9	P01	fn
SQRTSD/PD x,x/m 1 26-29 4-18 P01 fp VSQRTPD y,y/m 2 27-28 9-37 P01 fp RSQRTSS/PS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP Logic X,x/m 4 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPER 9 4 32 bit mode 64 bit mode VZEROUPPER 16 5 64 bit mode 64 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2							
VSQRTPD y,y/m 2 27-28 9-37 P01 fp RSQRTSS/PS x,x/m 1 5 1 P01 fp VRSQRTPS y,y/m 2 5 2 P01 fp VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,m 4 2 P01 AMD XOP Logic X,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode 64 bit mode VZEROUPPER 16 5 P02 32 bit mode 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode STMXCSR m32 2							
RSQRTSS/PS	· ·	·					
VRSQRTPS y,y/m 2 5 2 P01 fp VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,m 4 2 P01 AMD XOP Logic AND/ANDN/OR/XORPS/PD x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode 64 bit mode VZEROUPPER 16 5 64 bit mode 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2							•
VFRCZSS/SD/PS/PD x,x 2 10 2 P01 AMD XOP VFRCZSS/SD/PS/PD x,m 4 2 P01 AMD XOP Logic AND/ANDN/OR/XORPS/PD x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2							
VFRCZSS/SD/PS/PD x,m 4 2 P01 AMD XOP Logic AND/ANDN/OR/XORPS/PD x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 17 P0 P2							· ·
Logic AND/ANDN/OR/XORPS/PD x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode 64 bit mode VZEROUPPER 16 5 64 bit mode 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 10 P02 64 bit mode VZEROALL 32 17 P0 P2 7		•		10			
AND/ANDN/OR/XORPS/PD x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2	VFRCZSS/SD/PS/PD	x,m	4		2	P01	AMD XOP
AND/ANDN/OR/XORPS/PD x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2							
PD x,x/m 1 2 0.5 P02 ivec VAND/ANDN/OR/ XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2							
VAND/ANDN/OR/XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2			_	0	0.5	DOO	•
XORPS/PD y,y,y/m 2 2 1 P02 ivec Other VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2	-	x,x/m	1	2	0.5	P02	ivec
Other 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2				0	4	DOO	•
VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2	XORPS/PD	y,y,y/m	2	2	1	P02	ivec
VZEROUPPER 9 4 32 bit mode VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2	0.11						
VZEROUPPER 16 5 64 bit mode VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2							0013
VZEROALL 17 6 P02 32 bit mode VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2							
VZEROALL 32 10 P02 64 bit mode LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2							
LDMXCSR m32 9 36 P0 P2 STMXCSR m32 2 17 P0 P2							
STMXCSR m32 2 17 P0 P2							64 bit mode
	LDMXCSR				36	P0 P2	
FXSAVE m4096 59-67 78 P0 P1 P2	STMXCSR	m32	2		17	P0 P2	
	FXSAVE	m4096	59-67		78	P0 P1 P2	

Steamroller

FXRSTOR	m4096	104-112	160	P0 P1 P2	
XSAVE	m	121-137	147-166	P0 P1 P2	
XRSTOR	m	191-209	291-297	P0 P1 P2	

AMD Excavator

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB, JNE,

etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, x = 128 bit xmm register, y = 256 bit ymm register, m = 256 memory operand including indirect operands, m = 256 memory operand, etc.

Ops: Number of macro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 macro-operations use microcode.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. The latency listed does not include the memory operand where the listing for register and memory operand are joined (r/

m). The numbers are approximate.

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However, the throughput may be limited by other bottlenecks in the pipeline.

Execution pipe: Indicates which execution pipe or unit is used for the macro-operations:

Integer pipes:

EX0: integer ALU, division

EX1: integer ALU, multiplication, jump EX01: can use either EX0 or EX1 AG01: address generation unit 0 or 1 Floating point and vector pipes:

P0: floating point add, mul, div. Integer add, mul, bool P1: floating point add, mul, div. Shuffle, shift, pack

P2: Integer add. Bool, store P01: can use either P0 or P1 P02: can use either P0 or P2

Two macro-operations can execute simultaneously if they go to different

execution pipes

Domain: Tells which execution unit domain is used:

ivec: integer vector execution unit. fp: floating point execution unit. fma: floating point multiply/add subunit.

inherit: the output operand inherits the domain of the input operand.

ivec/fma means the input goes to the ivec domain and the output comes from the

fma domain.

There is an additional latency of 1 clock cycle if the output of an ivec instruction goes to the input of a fp or fma instruction, and when the output of a fp or fma instruction goes to the input of an ivec or store instruction. There is no latency between the fp and fma units. All other latencies after memory load and before mem-

ory store instructions are included in the latency counts.

An fma instruction has a latency of 5 if the output goes to another fma instruction, 6 if the output goes to an fp instruction, and 6+1 if the output goes to an ivec or

store instruction.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions						
MOV	r8,r8	1	1	0.5	EX01	

MOV	r16,r16	1	1	0.5	EX01	
MOV	r32,r32	1	1	0.25	EX01 or AG01	
MOV	r64,r64	1	1	0.25	EX01 or AG01	
MOV	r,i	1	1	0.25	EX01 or AG01	
MOV		1	3	0.5	AG01	all addr. modes
	r,m		4			
MOV	m,r	1	4	1	EX01 AG01	all addr. modes
MOV	m,i	1		1		
MOVNTI	m,r	1	4	1		
MOVZX, MOVSX	r,r	1	1	0.25		
MOVSX	r,m	1	5	0.5	EX01	
MOVZX	r,m	1	4	0.5	EX01	
MOVSXD	r64,r32	1	1	0.25	EX01	
MOVSXD	r64,m32	1	5	0.5	EX01	
CMOVcc	r,r	1	2	0.5	EX01	
CMOVcc	r,m	1	_	0.5	EX01	
XCHG	r8,r8	2	1	1	EX01	
1		2				
XCHG	r16,r16		1	1	EX01	
XCHG	r32,r32	2	1	0.5	EX01	
XCHG	r64,r64	2	1	0.5	EX01	
VOLIO		_	50	50	EV04	Timing depends on
XCHG	r,m	2	~50	~50	EX01	hardware
XLAT		2	6	2		
PUSH	r	1		1		
PUSH	i	1		1		
PUSH	m	2		1		
PUSHF(D/Q)		8		4		
PUSHA(D)		9		9		
POP	r	1		0.5		
POP	m	2		1		
POPF(D/Q)		34		12		
POPA(D)		14		8		
, ,			4	2		
POP	sp	1	1	2	EV04	
LEA	r16,[m]	2	2-3		EX01	
LEA	r32/64,[m]	1	1	0.5	EX01	all other cases
LAHF		4	2	2		
SAHF		2	2	1		
SALC		1	1	1		
BSWAP	r	1	1	0.5	EX01	
PREFETCHNTA	m	1		0.5		
PREFETCHT0/1/2	m	1		0.5		
PREFETCH/W	m	1		0.5		PREFETCHW
SFENCE		7		~100		111212101111
LFENCE		1		0.25		
		7				
MFENCE		/		~100		
A widh was a tip i was two a tip was						
Arithmetic instructions		4	4	0.05		
ADD, SUB	r,r	1	1	0.25		
ADD, SUB	r,i	1	1	0.25		
ADD, SUB	r,m	1		0.5	EX01	
ADD, SUB	m,r	1	7	1	EX01	
ADD, SUB	m,i	1	7	1	EX01	
ADC, SBB	r,r	1	1		EX01	
ADC, SBB	r,i	1	1		EX01	
,				l		I

			LXOGVATO			
ADC, SBB	r,m	1	1	1 1	EX01	
ADC, SBB	m,r	1	9	1	EX01	
1		1	9	1	EX01	
ADC, SBB	m,i					
CMP	r,r	1	1	0.25	EX01	
CMP	r,i	1	1	0.25	EX01	
CMP	r,m	1		0.5	EX01	
CMP	m,i	1		0.5	EX01	
INC, DEC, NEG	r	1	1	0.25	EX01	
INC, DEC, NEG	m	1	7	1	EX01	
AAA, AAS		10	5			
DAA		16	8			
DAS		20	9			
AAD		4	5			
AAM		10	15	15		
MUL, IMUL	r8/m8	1	3	2	EX1	
MUL, IMUL	r16/m16	2	3	2	EX1	
MUL, IMUL	r32/m32	1	3	2	EX1	
MUL, IMUL	r64/m64	1	5	4	EX1	
IMUL	r16,r16/m16	1	3	2	EX1	
IMUL	r32,r32/m32	1	3	2	EX1	
IMUL	· ·		5	4	EX1	
	r64,r64/m64	1				
IMUL	r16,(r16),i	2	5	2	EX1	
IMUL	r32,(r32),i	1	4	2	EX1	
IMUL	r64,(r64),i	1	6	4	EX1	
IMUL	r16,m16,i	2		2	EX1	
IMUL	r32,m32,i	2	4	2	EX1	
IMUL	r64,m64,i	2	7	4	EX1	
DIV	r8/m8	9	17-19		EX0	
DIV	r16/m16	7	16-26		EX0	approximate
DIV	r32/m32	2	14-42		EX0	timings
DIV	r64/m64	2	14-74		EX0	
IDIV	r8/m8	9	15-19		EX0	
IDIV	r16/m16	7	16-26		EX0	
IDIV		2			EX0	
	r32/m32		14-42			
IDIV	r64/m64	2	15-74		EX0	
CBW, CWDE, CDQE		1	1	_	EX01	
CDQ, CQO		1	1	0.5	EX01	
CWD		2	1	1	EX01	
Logic instructions	_					
AND, OR, XOR	r,r	1	1	0.33		
AND, OR, XOR	r,i	1	1	0.33		
AND, OR, XOR	r,m	1		0.5	EX01	
AND, OR, XOR	m,r	1	7	1	EX01	
AND, OR, XOR	m,i	1	7	1	EX01	
TEST	r,r	1	1	0.25	-	
TEST	r,i	1	1	0.25		
TEST	m,r	1		0.23	EX01	
TEST		1		0.5	EX01	
	m,i	1 4	4		ĽΛUΙ	
NOT	r	 	1	0.33	EV04	
NOT	m	1	7	1	EX01	5.47
ANDN	r,r,r	1	1	0.33	- 1/4 :	BMI1
SHL, SHR, SAR	r,i/CL	1	1	0.5	EX01	

			LX04 Vator			
ROL, ROR	r,i/CL	1	1	0.5	EX01	
RCL	r,1	1	1	0.0	EX01	
RCL	r,i	16	4		EX01	
RCL	r,cl	17	5		EX01	
RCR					EX01	
	r,1	1	1			
RCR	r,i	15	5		EX01	
RCR	r,cl	16	5		EX01	
SHLD, SHRD	r,r,i	6	3	3	EX01	
SHLD, SHRD	r,r,cl	7	3	3	EX01	
SHLD, SHRD	m,r,i/CL	8		4	EX01	
BT	r,r/i	1	1	0.5	EX01	
ВТ	m,i	1		0.5	EX01	
ВТ	m,r	7		4	EX01	
BTC, BTR, BTS	r,r/i	2	2	1	EX01	
BTC, BTR, BTS	m,i	4		2	EX01	
BTC, BTR, BTS	m,r	10		4	EX01	
BSF	r,r	6	3	3	EX01	
BSF		8	4	4	EX01	
BSR	r,m	7	4	4	EX01	
BSR	r,r	9	4	5		
	r,m		4		EX01	
SETcc	r	1	1	0.5	EX01	
SETcc	m	1		1	EX01	
CLC, STC		1		0.25		
CMC		1	1		EX01	
CLD		2		3		
STD		2		4		
POPCNT	r,r	1	4	1		SSE4.2
LZCNT	r,r	2	2	2	EX0	LZCNT
TZCNT	r,r	2	2	2		BMI1
BEXTR	r,r,r	1	1	0.5		BMI1
BEXTR	r,r,i	1	1	0.5		AMD TBM
BLSI	r,r	2	2	0.5		BMI1
BLSMSK	r,r	2	2	0.5		BMI1
BLSR	r,r	2	2	0.5		BMI1
BLCFILL	r,r	2	2	0.5		AMD TBM
BLCI	r,r	2	2	0.5		AMD TBM
BLCIC		2	2	0.5		AMD TBM
BLCMSK	r,r	2	2	0.5		AMD TBM
	r,r	2	2			
BLCS	r,r			0.5		AMD TBM
BLSFILL	r,r	2	2	0.5		AMD TBM
BLSI	r,r	2	2	0.5		AMD TBM
BLSIC	r,r	2	2	0.5		AMD TBM
T1MSKC	r,r	2	2	0.5		AMD TBM
TZMSK	r,r	2	2	0.5		AMD TBM
Control transfer instru						
JMP	short/near	1		2	EX1	
JMP	r	1		2	EX1	
JMP	m	1		2	EX1	
Jcc	short/near	1		1-2	EX1	2 if jumping
fused CMP+Jcc	short/near	1		1-2	EX1	2 if jumping
J(E/R)CXZ	short	1		1-2	EX1	2 if jumping
LOOP	short	1		1-2	EX1	2 if jumping
1	ı	i -	I	. =		Gat

LOOPE LOOPNE CALL CALL CALL RET RET BOUND INTO	short near r m i m	1 2 2 3 1 4 11 4		1-2 2 2 2 2 2 2 4 2	EX1 EX1 EX1 EX1 EX1 EX1	2 if jumping for no jump for no jump
String instructions LODS REP LODS STOS REP STOS REP STOS MOVS REP MOVS REP MOVS REP MOVS SCAS REP SCAS CMPS REP CMPS	m8/m16	3 6n 3 1n 3 per 16B 5 ~1n 4-5 pr 16B 3 7n 6 9n		3 2n – 3n 3 ~1n 2 per 16B 3 ~1n ~2 per 16B 3 3-4n 3 4n		small n best case small n best case
Synchronization LOCK ADD XADD LOCK XADD CMPXCHG CMPXCHG CMPXCHG LOCK CMPXCHG LOCK CMPXCHG LOCK CMPXCHG LOCK CMPXCHG CMPXCHGB LOCK CMPXCHG CMPXCHGBB LOCK CMPXCHGBB CMPXCHG16B LOCK CMPXCHG16B	m,r m,r m,r8 m,r16 m,r32/64 m8,r8 m16,r16 m,r32/64 m64 m64 m128 m128	1 4 4 5 6 6 5 6 18 18 24 24	~35 ~10 ~35 11 11 12 ~40 ~40 ~40 ~14 ~42 ~40 ~80			
Other NOP (90) Long NOP (0F 1F) PAUSE ENTER ENTER LEAVE CPUID XGETBV RDTSC RDTSCP RDPMC CRC32 CRC32 CRC32	a,0 a,b r32,r8 r32,r16 r32,r32	1 8 13 11+5b 2 38-64 4 44 44 22 3 5	3 5 6	0.25 0.25 4 21 20-30 3 100-300 30 78 105 380 2 5 6	none none	rdtscp

Floating point x87 instructions								
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes		
Move instructions								
FLD	r	1	2	0.5	P01	fp		
FLD	m32/64	1	6	1		fp		
FLD	m80	9	15	3		fp		
FBLD	m80	61	52	34	P0 P1 P2	fp		
FST(P)	r	1	2	0.5	P01	fp		
FST(P)	m32/64	2	6	1		fp		
FSTP	m80	13	14	19		fp		
FBSTP	m80	240	222	222	P0 P1 P2	fp		
FXCH	r	1	0	0.5	P01	inherit		
FILD	m	1	11	1	P01	fp		
FIST(T)(P)	m	2	7	1	P0 P2	fp		
FLDZ, FLD1	111	1	'	0.5	P01	fp		
FCMOVcc	ot0 r	8	3	3	P0 P1 P2	-		
FFREE	st0,r		3			fp		
	r	1	0	0.25	none	: In:-		
FINCSTP, FDECSTP	A > /	1	0	0.25	none	inherit		
FNSTSW	AX	3	11	19	P0 P2			
FNSTSW	m16	2		15	P0 P2			
FLDCW	m16	1		3				
FNSTCW	m16	2		2				
Arithmetic instructions								
FADD(P),FSUB(R)(P)	r/m	1	5	1	P01	fma		
FIADD,FISUB(R)	m	2		2	P01	fma		
FMUL(P)	r/m	1	5	1	P01	fma		
FIMUL	m	2		2	P01	fma		
FDIV(R)(P)	r	1	9-37	4-16	P01	fp		
FDIV(R)	m	1			P01	fp		
FIDIV(R)	m	2		4	P01	fp		
FABS, FCHS		1	2	0.5	P01	fp		
FCOM(P), FUCOM(P)	r/m	1		0.5	P01	fp		
FCOMPP, FUCOMPP		1		0.5	P01	fp		
FCOMI(P)	r	2	2	1	P01 P2	fp		
FICOM(P)	m	2	_	1	P01	fp		
FTST		1		0.5	P01	fp		
FXAM		1	26	0.5	P01	fp		
FRNDINT		1	4	1	P0	fp		
FPREM FPREM1		1	17-60	12-53	P0	fp		
B# a4la								
Math		4	10.00	F 00	D04			
FSQRT		1	10-20	5-20	P01			
FLDPI, etc.		1 1	00.010	0.5	P01			
FSIN		10-164	60-210	60-165	P0 P1 P2			
FCOS		18-166	76-158		P0 P1 P2			
FSINCOS		12-168		90-165	P0 P1 P2			
FPTAN		11-192	90-245	90-210	P0 P1 P2			
FPATAN		10-365	60-440	60-365	P0 P1 P2			
FSCALE		8	15	4	P0 P1 P2			

FXTRACT		13	8	5	P0 P1 P2	
F2XM1		10-18	60-74		P0 P1 P2	
FYL2X		9-183	60-280		P0 P1 P2	
FYL2XP1		206	~390		P0 P1 P2	
Other						
FNOP		1		0.25	none	
(F)WAIT		1		0.25	none	
FNCLEX		20		63	P0	
FNINIT		34		150	P0	
FNSAVE	m864	100	~250	~250	P0 P1 P2	
FRSTOR	m864	78	~150	~150	P0 P2	

Integer vector instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
				throughput	pipes	
Move instructions						
MOVD	r32, mm	1	7	1		
MOVD	mm, r32	2	8	1		
MOVD/Q	r32/64, x	1	7	1		
MOVD/Q	x, r32/64	2	9	1		
MOVD	mm/x,m32	1	5	0.5		
MOVD	m32,mm/x	1	4	1	P2	
MOVQ	mm/x,mm/x	1	2	0.5		
MOVQ	mm/x,m64	1	5	0.5		
MOVQ	m64,mm/x	1	4	1	P2	
MOVDQA	X,X	1	0	0.25		renaming
MOVDQA	x,m	1	5	0.5		
MOVDQA	m,x	1	4	1	P2	
VMOVDQA	y,y	2	2	0.5		lower half renamed
VMOVDQA	y,m	2	5	1		
VMOVDQA	m,y	2	4	2	P2	
MOVDQU	x,x	1	0	0.25		renaming
MOVDQU	x,m	1	5	0.5		
MOVDQU	m,x	1	4	1	P2	
LDDQU	x,m	1	5	0.5		
VMOVDQU	y,m	2	5	1		
VMOVDQU	m,y	2	4	2	P2	
MOVDQ2Q	mm,x	1	2	0.5		
MOVQ2DQ	x,mm	1	2	0.5		
MOVNTQ	m,mm	1	4	1	P2	
MOVNTDQ	m,x	1	4	1	P2	
VMOVNTDQA	m,y	2	5	2		
MOVNTDQA	x,m	1	4	0.5		
VMOVNTDQA	y,m	2	5	1		
PACKSSWB/DW	mm/x,r/m	1	2	1	P1	
PACKUSWB	mm/x,r/m	1	2	1	P1	
VPACKSSWB/DW	y,r/m	2	2	2	P1	
VPACKUSWB	y,r/m	2	2	2	P1	
PUNPCKH/LBW/WD/						
DQ	mm/x,r/m	1	2	1	P1	

PUNPCKL/HQDQ	x,r/m	1	2	1	P1	
VPUNPCKL/HQDQ	y,r/m	2	2	2	P1	
PSHUFB	mm/x,r/m	1	2	1	P1	
VPSHUFB	y,r/m	2	2	2	P1	
PSHUFD	x,x,i	1	2	1	P1	
VPSHUFD	y,y,i	2	2	2	P1	
PSHUFW	mm,mm,i	1	2	1	P1	
PSHUFL/HW	x,x,i	1	2	1	P1	
VPSHUFL/HW	y,y,i	2	2	2	P1	
PALIGNR	mm/x,r/m,i	1	2	1	P1	
VPALIGNR	y,r/m,i	2	2	2	P1	
PBLENDW	x,r/m,i	1	2	0.5	P02	
PBLENDW	y,r/m,i	2	2	1	P02	
MASKMOVQ	mm,mm	31	~50	~50		
MASKMOVDQU	x,x	65	~50	~50		
VPMASKMOVD/Q	x,x,m	1	6	0.5		
VPMASKMOVD/Q	y,y,m	2	6	1		
VPMASKMOVD/Q	m,x,x	2	10-40	30		
VPMASKMOVD/Q	m,y,y	4	10-40	60		
PMOVMSKB	r32,mm/x	2	8	1	P12	
VPMOVMSKB	r32,y	2	9	2	P12	
PEXTRB/W/D/Q	r,x/mm,i	2	9	1	P1 P2	
PINSRB/W/D/Q	x/mm,r,i	2	10	1 1	P1	
EXTRQ	x,i,i	1	2	1 1	P1	AMD SSE4A
EXTRQ	X,1,1 X,X	1	2	1	P1	AMD SSE4A
INSERTQ	x,x,i,i	1	1	1	P1	AMD SSE4A
INSERTQ	X,X,1,1 X,X	1	2	1	P1	AMD SSE4A
PMOVSXBW/BD/BQ/	Α,Α	'	_	•		7 WID COL-17 (
WD/WQ/DQ	x,x	1	2	1	P1	
PMOVSXBW/BD/BQ/	,	-		-		
WD/WQ/DQ	y,x	3	3	3	P1	
PMOVZXBW/BD/BQ/						
WD/WQ/DQ	X,X	1	2	1	P1	
PMOVZXBW/BD/BQ/						
WD/WQ/DQ	y,x	3	3	3	P1	
VINSERTI128	y,y,x,i	2	2	1	P0 P2	AVX2
VINSERTI128	y,y,m,i	2	7	1	P0 P2	AVX2
VPBROADCAST		_	0	4	D4	A) ()(O
B/W/D/Q	X,X	1	2	1	P1	AVX2
VPBROADCAST B/W/D/Q	x,m					
VPBROADCAST	۸,۱۱۱					
B/W/D/Q	y,x	2	3	2	P1	AVX2
VPBROADCAST	,					
B/W/D/Q	y,m					
VBROADCASTI128	y,m128	2	5	0.5	P02	AVX2
VPGATHERDD	x,[r+s*x],x	34		15		AVX2
VPGATHERDD	y,[r+s*y],y	51		21		AVX2
VPGATHERQD	x,[r+s*x],x	25		13		AVX2
VPGATHERQD	x,[r+s*y],x	32		14		AVX2
VPGATHERDQ	x,[r+s*x],x	24		13		AVX2
VPGATHERDQ	y,[r+s*x],y	32		15		AVX2
VPGATHERQQ	x,[r+s*x],x	24		49		AVX2
VPGATHERQQ	y,[r+s*y],y	31		15		AVX2

1		I	1	I	1
Arithmetic instructions	5				
PADDB/W/D/Q/SB/SW/				0.5	500
USB/USW	mm/x,r/m	1	2	0.5	P02
VPADD	y,y,r/m	2	2	1	P02
PSUBB/W/D/Q/SB/SW/	, ,			0.5	Doo
USB/USW	mm/x,r/m	1	2	0.5	P02
VPSUB	y,y,r/m	2	2	1	P02
PHADD/SUB(S)W/D	mm,mm	3	4	2	P0 P1 P2
PHADD/SUB(S)W/D	X,X	4	4	2	P0 P1 P2
VPHADD/SUB(S)W/D	y,y,y	8	4	4	P0 P1 P2
PCMPEQ B/W/D/Q	mm/x,r/m	1	2	0.5	P02
VPCMPEQ B/W/D/Q	y,y,r/m	2	2	1	P02
PCMPGT B/W/D/Q	mm/x,r/m	1	2	0.5	P02
VPCMPGT B/W/D/Q	y,y,r/m	2	2	1	P02
PMULLW PMULHW					
PMULHUW PMULLD					
PMULDQ PMULUDQ	mm/x,r/m	1	4	1	P0
VPMULLW VPMULHW					
PMULLD VPMULHUW					
VPMULDQ				_	
VPMULUDQ	y,y,r/m	2	4	2	P0
PMULHRSW	mm/x,r/m	1	4	1	P0
VPMULHRSW	y,y,r/m	2	4	2	P0
PMADDWD	mm/x,r/m	1	4	1	P0
VPMADDWD	y,y,r/m	2	4	2	P0
PMADDUBSW	mm/x,r/m	1	4	1	P0
VPMADDUBSW	y,y,r/m	2	4	2	P0
PAVGB/W	mm/x,r/m	1	2	0.5	P02
VPAVGB/W	mm/x,r/m	2	2	1	P02
PMIN/MAX SB/SW/ SD					
UB/UW/UD	mm/x,r/m	1	2	0.5	P02
VPMIN/MAX SB/SW/					
SD UB/UW/UD	y,y,r/m	2	2	1	P02
PHMINPOSUW	x,r/m	2	3	1	P02 P1
PABSB/W/D	mm/x,r/m	1	2	0.5	P02
VPABSB/W/D	y,r/m	2	2	1	P02
PSIGNB/W/D	mm/x,r/m	1	2	0.5	P02
VPSIGNB/W/D	y,r/m	2	2	1	P02
PSADBW	mm/x,r/m	2	4	1	P02
VPSADBW	y,y,r/m	4	4	2	P02
MPSADBW	x,x,i	12	7	4	P0 P1 P2
VMPSADBW	y,y,y,i	8	7	8	P0 P1 P2
57.2211	<i>J</i> , <i>J</i> , <i>J</i> ,.				
Logic					
PAND PANDN POR					
PXOR	mm/x,r/m	1	2	0.5	P02
VPAND VPANDN	,,	-	_		
VPOR VPXOR	y,y,r/m	2	2	1	P02
PSLL/RL W/D/Q	3737				
PSRAW/D	mm/x,r/m	1	2	1	P1
VPSLL/RL W/D/Q VP-	,				
SRAW/D	y,y,x/m	2	2	2	P1
PSLL/RL W/D/Q					
PSRAW/D	mm/x,i	1	2	1	P1
SRAW/D PSLL/RL W/D/Q					

VPSLL/RL W/D/Q VP- SRAW/D PSLLDQ, PSRLDQ VPSLLDQ VPSRLDQ	y,y,i x,i y,y,i	2 1 2	2 2 2	2 1 2	P1 P1 P1	
VPSLLVD/Q VPSRAVD VPSRLVD/Q VPSLLVD/Q	x,x,x	1	2	1	P1	AVX2
VPSRAVD VPSRLVD/Q	у,у,у	2	2	2	P1	AVX2
PTEST VPTEST	x,r/m y,y/m	2 4		1 2	P1 P2 P1 P2	
String instructions						
PCMPESTRI PCMPESTRI	x,x,i x,m,i	31 31	42	9		SSE4.2 SSE4.2
PCMPESTRM	x,x,i	30	37	8		SSE4.2
PCMPESTRM PCMPISTRI	x,m,i x,x,i	31 10	21	6		SSE4.2 SSE4.2
PCMPISTRI PCMPISTRM	x,m,i	9	26	5		SSE4.2 SSE4.2
PCMPISTRM	x,x,i x,m,i	10	20	5		SSE4.2 SSE4.2
Encryption						
PCLMULQDQ AESDEC	x,x/m,i x,x	4 2	5 4	5 1	P01	pclmul aes
AESDECLAST	X,X X,X	2	4	1	P01	aes
AESENC	X,X	2	5	1	P01	aes
AESENCLAST AESIMC	X,X	2	4	1 1	P01 P0	aes
AESKEYGENASSIST	x,x x,x,i	1	4	1	P0 P0	aes aes
Other						
EMMS		1		0.25		

Floating point vector instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Domain, notes
				throughput	pipes	
Move instructions						
MOVAPS/D MOVUPS/						
D	X,X	1	0	0.25	none	inherit domain
VMOVAPS/D	y,y	2	2	0.5	P02	ivec
MOVAPS/D MOVUPS/						
D	x,m128	1	5	0.5		
VMOVAPS/D						
VMOVUPS/D	y,m256	2	5	1		
MOVAPS/D MOVUPS/						
D	m128,x	1	4	1	P2	
VMOVAPS/D	m256,y	2	4	2	P2	
VMOVUPS/D	m256,y	2	4	2	P2	
MOVSS/D	x,x	1	2	0.5	P01	
MOVSS/D	x,m32/64	1	5	0.5		
MOVSS/D	m32/64,x	1	4	1	P2	

			_xcavato.			
MOVHPS/D	x,m64	1	6	1	P2	
MOVLPS/D	x,m64	1	6	0.5	P01	
MOVHPS/D	m64,x	2	6	2	P2	
MOVLPS/D	m64,x	1	6	1	P2	
MOVLHPS MOVHLPS	x,x	1	2	1	P2	
MOVMSKPS/D	r32,x	2	8	1	P1 P2	
VMOVMSKPS/D	r32,y	3	8	2	P1 P2	
MOVNTPS/D	m128,x	1	4	1	P2	
VMOVNTPS/D	m256,y	2	4	3	P2	
MOVNTSS/SD	m,x	2				AMD SSE4A
SHUFPS/D	x,x/m,i	1	2	1	P2	ivec
VSHUFPS/D	y,y,y/m,i	2	2	2	P2	ivec
VPERMILPS/PD	x,x,x/m	1	3	1	P1	ivec
VPERMILPS/PD	y,y,y/m	2	3	2	P1	ivec
VPERMILPS/PD	x,x/m,i	1	2	1	P1	ivec
VPERMILPS/PD	y,y/m,i	2	2	2	P1	ivec
VPERM2F128	y,y,y,i	8	4	3	P0 P2	ivec
VPERM2F128	y,y,m,i	12		4	P0 P2	ivec
BLENDPS/PD	x,x/m,i	1	2	0.5	P01	fp
VBLENDPS/PD	y,y,y/m,i	2	2	1	P01	fp
BLENDVPS/PD	x,x/m,xmm0	1	2	0.5	P01	ıρ
VBLENDVPS/PD	y,y,y/m,y	2	2	0.5	P01	
MOVDDUP	x,X	1	2	1	P1	ivec
MOVDDUP	x,m64	1		0.5	ГІ	IVEC
VMOVDDUP	· ·	2	2	2	P1	ivec
VMOVDDUP	y,y y,m256	2		1	ГІ	IVEC
VBROADCASTSS		1	2	1	P2	
	X,X			-	FZ	
VBROADCASTSS	x,m32	1 2	5	0.5	DO	
VBROADCASTSS/SD	y,x	2	3	2	P2	
VBROADCASTSS/SD	y,m32		5	0.5	P02	
VBROADCASTF128	y,m128	2	5	0.5	P02	
MOVSH/LDUP	X,X	1	2	1	P1	ivec
MOVSH/LDUP	x,m128	1	0	0.5	D4	•
VMOVSH/LDUP	у,у	2	3	2	P1	ivec
VMOVSH/LDUP	y,m256	2	•	1	D 0	
UNPCKH/LPS/D	x,x/m	1	2	1	P2	
VUNPCKH/LPS/D	y,y,y/m	2	2	2	P2	
EXTRACTPS	r32,x,i	2	9	2	P2	
EXTRACTPS	m32,x,i	2	6	2	P2	
VEXTRACTF128	x,y,i	1	2	0.5	P02	ivec
VEXTRACTF128	m128,y,i	2	7	1	P0 P2	
INSERTPS	x,x,i	1	2	1	P2	
INSERTPS	x,m32,i	1	7	1	P2	
VINSERTF128	y,y,x,i	2	2	1	P02	ivec
VINSERTF128	y,y,m128,i	2	8	1	P02	
VMASKMOVPS/D	x,x,m128	1	8	0.5	P01	
VMASKMOVPS/D	y,y,m256	2	8	1	P01	
VMASKMOVPS/D	m128,x,x	20	~40		P0 P1 P2	
VMASKMOVPS/D	m256,y,y	41	~40		P0 P1 P2	
Conversion					_	
CVTPD2PS	x,x	2	6	1	P0 P2	
VCVTPD2PS	x,y	4	6	2	P0 P2	

CVTPS2PD				LXOGVATO			
VCVTPS2PD	CVTPS2PD	X,X	2	6	1	P0 P2	
CVTSD2SS	VCVTPS2PD		4	6	2	P0 P2	
CVTS2SSD	CVTSD2SS		1	4	1	P0	
CVTDQ2PS	CVTSS2SD		1	4	1	P0	
VCVTTQ2PS			1	4	1	P0	
CVT(T) PS2DQ			2	4			
VCVT[T] PS2DQ							
CVTDQ2PD							
VCVTTQ2PD							
CVT(T)PD2DQ							
VCVTT(T)PD2DQ							
CVTPI2PS	, ,						
CVT(T)PS2PI	` '						
CVTPi2PD					•		
CVT(T) PD2PI mm,x 2 6 1 P0 P1 CVTSI2SS x,r32 2 12 1 P0 int CVT(T)SS2SI r32,x 2 12 1 P0 P2 CVT(T)SD2SI x,r32/64,x 2 12 1 P0 P1 CVT(T)SD2SI r32/64,x 2 12 1 P0 P2 VCVTPS2PH x/m,x,i 2 6 2 P0 P1 F16C VCVTPB2PH x/m,y,i 4 6 2 P0 P1 F16C VCVTPH2PS x,x/m 2 6 2 P0 P1 F16C VCVTPH2PS x,x/m 2 6 2 P0 P1 F16C Arithmetic A A 1 5 1 P01 fma ADDSS/D SUBPS/D x,x/m 1 5 1 P01 fma VADDPS/D VSUBPS/D x,x/m 1 5 1 P01 fma VADDSUBPS/D x,x/m </td <td></td> <td></td> <td></td> <td></td> <td>•</td> <td></td> <td></td>					•		
CVTSi2SS					1		
CVT(T)SS2SI r32,x 2 12 1 P0 P2 CVTSI2SD x,r32/64,x 2 12 1 P0 int CVT(T)SD2SI r32/64,x 2 12 1 P0 P1 F16C VCVTPS2PH x/m,x,i 2 6 2 P0 P1 F16C VCVTPB2PH x/m,y,i 4 6 2 P0 P1 F16C VCVTPH2PS x,x/m 2 6 2 P0 P1 F16C VCVTPH2PS x,x/m 4 6 2 P0 P1 F16C VCVTPH2PS x,x/m 4 6 2 P0 P1 F16C ADDSS/D SUBSS/D x,x/m 1 5 1 P01 fma ADDPS/D VSUBPS/D x,x/m 1 5 1 P01 fma VADDSUBPS/D x,x/m 1 5 1 P01 fma VADDSUBPS/D x,x/m 1 5 1 P01 fma					1		
CVTSI2SD x,r32/64 2 12 1 P0 int CVT(T)SD2SI r32/64,x 2 12 1 P0 P2 VCVTPS2PH x/m,x,i 2 6 2 P0 P1 F16C VCVTPS2PH x/m,x,i 4 6 2 P0 P1 F16C VCVTPH2PS x,x/m 2 6 2 P0 P1 F16C VCVTPH2PS x,x/m 4 6 2 P0 P1 F16C VCVTPH2PS x,x/m 4 6 2 P0 P1 F16C VCVTPH2PS x,x/m 4 6 2 P0 P1 F16C Arithmetic Antition of the point					1		
CVT(T)SD2SI	CVT(T)SS2SI	r32,x			1	P0 P2	
VCVTPS2PH x/m,x,i 2 6 2 P0 P1 F16C VCVTPS2PH x/m,y,i 4 6 2 P0 P1 F16C VCVTPH2PS x,x/m 2 6 2 P0 P1 F16C VCVTPH2PS y,x/m 4 6 2 P0 P1 F16C VCVTPH2PS y,x/m 4 6 2 P0 P1 F16C ACTPH2PS y,x/m 4 5 1 P01 fma ADDSUBPS/D y,x,y/m 2 5 2 P01 fma ADDSUBPS/D y,y,y/m 2 5 1 P01 fma </td <td>CVTSI2SD</td> <td>x,r32/64</td> <td></td> <td>12</td> <td>1</td> <td>P0 int</td> <td></td>	CVTSI2SD	x,r32/64		12	1	P0 int	
VCVTPS2PH VCVTPH2PS x/m,y,i 4 6 2 P0 P1 F16C F16C VCVTPH2PS x,x/m 2 6 2 P0 P1 F16C VCVTPH2PS x,x/m 4 6 2 P0 P1 F16C Arithmetic ADDSS/D SUBSS/D X,x/m 1 5 1 P01 fma ADDPS/D VSUBPS/D ADDS/D VSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VAX 4 9 2 P01 fma HADDPS/D HSUBPS/D VHADDPS/D VHADDPS/D VHADDPS/D VHADDPS/D VHADDPS/D VAX/m x,x/m 1 5 1 P01 fma MULPS MULSD MULSD MULPD VY,y,y/m VHULPD VY,y,y/m VAULPD VY,y,y/m VAULPD VY,y,y/m VAULPD VY,y,y/m VAULPD VY,y,y/m VA 2 5 1 P01 fma MULPS VMULPD VY,y,y/m VAIVE V	CVT(T)SD2SI	r32/64,x	2	12	1	P0 P2	
VCVTPH2PS x,x/m 2 6 2 P0 P1 F16C VCVTPH2PS y,x/m 4 6 2 P0 P1 F16C Arithmetic ADDSS/D SUBSS/D ADDSS/D SUBPS/D ADDSS/D SUBPS/D VX,x/m 1 5 1 P01 fma ADDSS/D VSUBPS/D ADDSUBPS/D VADDSUBPS/D VADDSUB	VCVTPS2PH	x/m,x,i	2	6	2	P0 P1	F16C
VCVTPH2PS x,x/m 2 6 2 P0 P1 F16C VCVTPH2PS y,x/m 4 6 2 P0 P1 F16C Arithmetic ADDSS/D SUBSS/D ADDSS/D SUBPS/D	VCVTPS2PH	x/m,y,i	4	6	2	P0 P1	F16C
Arithmetic X, x/m 1 5 1 P01 fma ADDPS/D SUBPS/D X, x/m 1 5 1 P01 fma VADDPS/D VSUBPS/D X, x/m 1 5 1 P01 fma VADDSUBPS/D Y, y, y/m 2 5 2 P01 fma ADDSUBPS/D X, x/m 1 5 1 P01 fma VADDSUBPS/D X, x/m 1 5 1 P01 fma VADDSUBPS/D X, x/m 1 5 1 P01 fma VADDPS/D HSUBPS/D X, x/m 1 5 1 P01 fma VHADDPS/D Y, y, y/m 2 5 1 P01 fma VHADDPS/D Y, y, y/m 8 9 3 P01 P1 ivec/fma VHADDPS/D Y, y, y/m 8 9 3 P01 P1 ivec/fma MULPS WULPD Y, y, y/m 1 5 0.5	VCVTPH2PS		2	6	2	P0 P1	F16C
Arithmetic x,x/m 1 5 1 P01 fma ADDPS/D SUBPS/D x,x/m 1 5 1 P01 fma VADDPS/D VSUBPS/D x,x/m 1 5 1 P01 fma VADDSUBPS/D y,y,y/m 2 5 2 P01 fma ADDSUBPS/D x,x/m 1 5 1 P01 fma HADDPS/D HSUBPS/D x,x/m 1 5 1 P01 fma HADDPS/D HSUBPS/D x,x 4 9 2 P0 P1 ivec/fma HADDPS/D HSUBPS/D x,x 4 9 2 P0 P1 ivec/fma HADDPS/D HSUBPS/D x,x 4 9 2 P0 P1 ivec/fma HADDPS/D HSUBPS/D x,x/m 4 9 2 P0 P1 ivec/fma HADDPS/D HSUBPS/D x,x/m 1 5 0.5 P01 fma WHADDPS/D y,y,y/m 8 9	VCVTPH2PS			6		P0 P1	F16C
ADDSS/D SUBSS/D ADDPS/D SUBPS/D x,x/m 1 5 1 P01 fma VADDPS/D SUBPS/D ADDPS/D VADDPS/D VADDPS/D VADDPS/D VADDSUBPS/D ADDSUBPS/D ADDSUBPS/D ADDSUBPS/D VADDSUBPS/D ADDSUBPS/D		3 /					
ADDPS/D SUBPS/D x,x/m 1 5 1 P01 fma VADDPS/D VSUBPS/D ADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VASUBPS/D VASUBP	Arithmetic						
ADDPS/D SUBPS/D x,x/m 1 5 1 P01 fma VADDPS/D VSUBPS/D ADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VADDSUBPS/D VASUBPS/D VASUBP		x,x/m	1	5	1	P01	fma
VADDPS/D VSUBPS/D ADDSUBPS/D ADDSUBPS/D y,y,y/m 2 5 2 P01 fma fma fma fma fma fma fma fma fma fma			1			P01	
ADDSUBPS/D x,x/m 1 5 1 P01 fma VADDSUBPS/D y,y,y/m 2 5 1 P01 fma HADDPS/D HSUBPS/D VHADDPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS VHSULSD x,x/m 1 5 0.5 P01 fma MULSS MULSD x,x/m 1 5 0.5 P01 fma MULPS WULPD x,x/m 1 5 0.5 P01 fma VMULPS VMULPD VMULPD y,y,y/m 2 5 1 P01 fma VDIVPS VDIVPS x,x/m 1 12 4 P01 fp VDIVPD VDIVPD V,y,y/m 2 12 8 P01 fp VDIVPD V,y,y/m 2 8-22 8-16 P01 fp VCPPS y,x/m 1 5 2 P01 fp VRCPPS y,y/m 2 5 2 P01 fp VRCPPS/D x,x/m 1 2		71,74					2
ADDSUBPS/D x,x/m 1 5 1 P01 fma VADDSUBPS/D y,y,y/m 2 5 1 P01 fma HADDPS/D HSUBPS/D VHADDPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS /D VHSUBPS/D VH	VADDPS/D VSUBPS/D	y,y,y/m	2	5	2	P01	fma
VADDSUBPS/D y,y,y/m 2 5 1 P01 fma HADDPS/D HSUBPS/D VHADDPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPS/D VHSUBPD X,x/m y,y,y/m 8 9 3 P01 P1 ivec/fma MULSS MULSD X,x/m MULPD VHS WULPD VHS VMULPD	ADDSUBPS/D		1		1	P01	fma
HADDPS/D HSUBPS/D VHADDPS/D VHADDPS/D VHSUBPS/D SUBPD VHSUBPD VHSUBPD VHSUBPD VHSUBPD VHSUBPD VHSUBPS VHSUBPD VHSUBPD VHSUBPD VHSUBPS VHSUBPD VHSUBPS VHSUBPS VHSUBPS VHSUBPS VHSUBPS VHSUBPS VHSUBPS/VHSUBPS			2		1		
VHADDPS/D VHSUBPS/D y,y,y/m 8 9 3 P01 P1 ivec/fma MULSS MULSD x,x/m 1 5 0.5 P01 fma MULPS MULPD x,x/m 1 5 0.5 P01 fma VMULPS VMULPD y,y,y/m 2 5 1 P01 fma VMUSS DIVPS x,x/m 1 12 4 P01 fp VDIVPS y,y,y/m 2 12 8 P01 fp DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp VRCPPS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPPS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp		J , J , J ,	_		-		
VHADDPS/D VHSUBPS/D VHSUBPS/D y,y,y/m 8 9 3 P01 P1 ivec/fma MULSS MULSD x,x/m 1 5 0.5 P01 fma MULPS MULPD x,x/m 1 5 0.5 P01 fma VMULPS VMULPD y,y,y/m 2 5 1 P01 fma DIVSS DIVPS x,x/m 1 12 4 P01 fp VDIVPS y,y,y/m 2 12 8 P01 fp DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp VCOPS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPPS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 <t< td=""><td>HADDPS/D HSUBPS/D</td><td>x,x</td><td>4</td><td>9</td><td>2</td><td>P0 P1</td><td>ivec/fma</td></t<>	HADDPS/D HSUBPS/D	x,x	4	9	2	P0 P1	ivec/fma
VHSUBPS/D y,y,y/m 8 9 3 P01 P1 ivec/fma MULSS MULSD x,x/m 1 5 0.5 P01 fma MULPS MULPD x,x/m 1 5 0.5 P01 fma VMULPS VMULPD y,y,y/m 2 5 1 P01 fma DIVSS DIVPS x,x/m 1 12 4 P01 fp VDIVPS y,y,y/m 2 12 8 P01 fp DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp VCOPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp	VHADDPS/D						
MULPS MULPD x,x/m 1 5 0.5 P01 fma VMULPS VMULPD y,y,y/m 2 5 1 P01 fma DIVSS DIVPS x,x/m 1 12 4 P01 fp VDIVPS y,y,y/m 2 12 8 P01 fp DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp RCPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp		y,y,y/m	8	9	3	P01 P1	ivec/fma
VMULPS VMULPD y,y,y/m 2 5 1 P01 fma DIVSS DIVPS x,x/m 1 12 4 P01 fp VDIVPS y,y,y/m 2 12 8 P01 fp DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp RCPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp	MULSS MULSD	x,x/m	1	5	0.5	P01	fma
VMULPS VMULPD y,y,y/m 2 5 1 P01 fma DIVSS DIVPS x,x/m 1 12 4 P01 fp VDIVPS y,y,y/m 2 12 8 P01 fp DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp RCPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp	MULPS MULPD	x,x/m	1	5	0.5	P01	fma
DIVSS DIVPS x,x/m 1 12 4 P01 fp VDIVPS y,y,y/m 2 12 8 P01 fp DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp RCPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp	VMULPS VMULPD		2	5		P01	fma
VDIVPS y,y,y/m 2 12 8 P01 fp DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp RCPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp	DIVSS DIVPS						
DIVSD DIVPD x,x/m 1 8-22 4-8 P01 fp VDIVPD y,y,y/m 2 8-22 8-16 P01 fp RCPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp MAXSS/SD/PS/PD x,x/m 2 8 1 P01 P2 fp			2				-
VDIVPD y,y,y/m 2 8-22 8-16 P01 fp RCPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp MAXSS/SD/PS/PD x,x/m 2 8 1 P01 P2 fp							-
RCPSS/PS x,x/m 1 5 1 P01 fp VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp MAXSS/SD/PS/PD x,x/m 2 8 1 P01 P2 fp			I -				-
VRCPPS y,y/m 2 5 2 P01 fp CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp MAXSS/SD/PS/PD x,x/m 2 8 1 P01 P2 fp							-
CMPSS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp MAXSS/SD/PS/PD x,x/m 2 8 1 P01 P2 fp							
CMPPS/D x,x/m 1 2 0.5 P01 fp VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp MAXSS/SD/PS/PD x,x/m 2 8 1 P01 P2 fp	_	у,у/пп		3		FUI	ıρ
VCMPPS/D y,y,y/m 2 2 1 P01 fp COMISS/D UCOMISS/D x,x/m 2 8 1 P01 P2 fp MAXSS/SD/PS/PD fp fp fp fp		v v/m	1	2	0.5	P ∩1	fn
COMISS/D UCOMISS/ D							1
D		y,y,y/111			!	FUI	ıp
MAXSS/SD/PS/PD		v v/m	2	Ω	1	D01 D2	fn
	-	۸,۸/۱۱۱		0	'	FULFZ	ıp
1 2 0.5 101 lp		y y/m	1	2	0.5	P01	fn
	IVIII VOO/OD/II O/FD	A,A/111	'	_	0.5	101	l ib
VMAXPS/D VMINPS/D y,y,y/m 2 2 1 P01 fp	VMAXPS/D VMINPS/D	y,y,y/m	2	2	1	P01	fp

ROUNDSS/SD/PS/PD	x,x/m,i	1	4	1	P0	fp
VROUNDSS/SD/PS/				_		
PD	y,y/m,i	2	4	2	P0	fp
DPPS	x,x/m,i	9	20	4	P0 P1	
VDPPS	y,y,y,i	13	22	5	P0 P1	
VDPPS	y,m,i	15		6	P0 P1	
DPPD	x,x,i	7	14	4	P0 P1	
DPPD	x,m,i	8		4	P0 P1	
VFMADD132SS/SD	x,x,x/m	1	5	0.5	P01	FMA3
VFMADD132PS/PD	x,x,x/m	1	5	0.5	P01	FMA3
VFMADD132PS/PD	y,y,y/m	2	5	1	P01	FMA3
All other FMA3 instruction		bove				FMA3
VFMADDSS/SD	x,x,x,x/m	1	5	0.5	P01	AMD FMA4
VFMADDPS/PD	x,x,x,x/m	1	5	0.5	P01	AMD FMA4
VFMADDPS/PD	y,y,y,y/m	2	5	1	P01	AMD FMA4
All other FMA4 instruction			Ü	•	101	AMD FMA4
All other r way instruction	ns. same as a	DOVC				AIVIDTIVIA
Math						
SQRTSS/PS	x,x/m	1	13	4	P01	fp
VSQRTPS	y,y/m	2	13	9	P01	fp
SQRTSD/PD	x,x/m	1	16	4-9	P01	fp
VSQRTPD	y,y/m	2	16	8-19	P01	fp
RSQRTSS/PS	x,x/m	1	4	1	P01	fp
VRSQRTPS	y,y/m	2	4	2	P01	fp
VFRCZSS/SD/PS/PD	у, у/111 Х,Х	2	38	4	P01	AMD XOP
VFRCZSS/SD/PS/PD	x,n	3	30	7	P01	AMD XOP
VI NC233/3D/F3/FD	۸,۱۱۱	3			FUI	AIVID AOF
Logic						
AND/ANDN/OR/XORPS/						
PD	x,x/m	1	2	0.5	P02	ivec
VAND/ANDN/OR/	71,74		_	0.0	. •=	
XORPS/PD	y,y,y/m	2	2	1	P02	ivec
in the second se	<i>J</i> , <i>J</i> , <i>J</i> ,	_	_	•	. •=	
Other						
VZEROUPPER		10		4		32 bit mode
VZEROUPPER		17		5		64 bit mode
VZEROALL		18		6	P02	32 bit mode
VZEROALL		33		9	P02	64 bit mode
LDMXCSR	m32	9		~50	ΓUZ	04 bit iiioue
STMXCSR						
	m32	2		~50		20 hit -
FXSAVE	m4096	76		100		32 bit mode
FXSAVE	m4096	84		180		64 bit mode
FXRSTOR	m4096	124		250		32 bit mode
FXRSTOR	m4096	129		150		64 bit mode
XSAVE	m	139		215		32 bit mode
XSAVE	m	155		170		64 bit mode
XRSTOR	m	204		350		32 bit mode
XRSTOR	m					64 bit mode

AMD Zen 1

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB, JNE,

etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, x = 128 bit xmm register, y = 256 bit ymm register, m = 256 any memory operand including indirect operands, m64 means m64 memory operand, etc.

Ops: Number of macro-operations issued from instruction decoder to schedulers.

Latency: This is the delay that the instruction generates in a dependency chain. The numbers are minimum values. Cache misses, misalignment, and exceptions may in-

crease the clock counts considerably. The latency listed does not include the memory operand where the listing for register and memory operand are joined (r/

m).

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However,

the throughput may be limited by other bottlenecks in the pipeline.

Execution pipe: Indicates which execution pipe or unit is used for the macro-operations:

P0: Floating point and vector pipe 0 P1: Floating point and vector pipe 1 P2: Floating point and vector pipe 2 P3: Floating point and vector pipe 3 P0 P1: Uses both P0 and P1

P01: Uses either P0 and P1

Where no unit is specified, it uses one or more integer pipe or address generation

units.

Two micro-operations can execute simultaneously if they go to different

execution pipes

Domain: Tells which execution unit domain is used:

ivec: integer vector execution unit. fp: floating point execution unit.

inherit: the output operand inherits the domain of the input operand.

There is an additional latency of 1 clock cycle if the output of an ivec instruction goes to the input of a fp instruction, and when the output of a fp instruction goes to the input of an ivec instruction. All other latencies after memory load and before

memory store instructions are included in the latency counts.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions						
MOV	r8,r8	1	1	0.25		
MOV	r16,r16	1	1	0.25		
MOV	r32,r32	1	0	0.3		renaming
MOV	r64,r64	1	0	0.2		renaming
MOV	r,i	1		0.25		
MOV	r,m	1	4	0.5		
MOV	m,r	1	3	1		
MOV	m,i	1		1		
MOVNTI	m,r	1	high	1		
MOVZX, MOVSX	r,r	1	1	0.25		

MOV/ZV MOV/OV	l			0.5	I
MOVZX, MOVSX	r,m	1	4	0.5	
MOVSXD	r64,r32	1	1	0.25	
MOVSXD	r64,m32	1	3	0.5	
CMOVcc	r,r	1	1	0.25	
CMOVcc	r,m	1		0.5	
XCHG	r8,r8	2	1	1	
XCHG	r16,r16	2	1	1	
XCHG	r32,r32	2	0	0.33	renaming
XCHG	r64,r64	2	0	0.33	renaming
XCHG	r,m	2	~30	~30	depends on hw
XLAT		2	5	2	
PUSH	r	1		1	
PUSH	i	1		1	
PUSH	m	2		1	
PUSHF(D/Q)		9		4	
PUSH	sp	2		1	
PUSHA(D)		9		8	
POP	r	1		0.5	
POP	m	2		1	
POPF(D/Q)	""	35		13	
POPA(D)		9		4	
POPA(D)	90	1	2		
	sp		2 2	2	
LEA	r16,[m]	2		1	241.4
LEA	r32,[m]	1	1	0.5	64 bit mode
LEA	r32/64,[m]		1	0.5	rip relative
LEA	r32/64,[m]	1	1	0.25	all other cases
LAHF		4	3	2	
SAHF		2	2	0.5	
SALC		1	1	1	
BSWAP	r	1	1	0.25	
MOVBE	r,[m]	1		0.5	MOVBE
MOVBE	[m],r	1		1	MOVBE
PREFETCHNTA	m	1		0.5	
PREFETCHT0/1/2	m	1		0.5	
PREFETCH/W	m	1		0.5	PREFETCHW
SFENCE		4		~20	
LFENCE		1		0.25	
MFENCE		7		~70	
2.102		•			
Arithmetic instructions	 				
ADD, SUB	r,r	1	1	0.25	
ADD, SUB	r,i	1	1	0.25	
ADD, SUB	r,m	1		0.5	
ADD, SUB	m,r	1	6	1	
ADD, SUB	m,i	1	6	1	
ADC, SBB		1	1	'	
1	r,r	· ·			
ADC, SBB	r,i	1	1	4	
ADC, SBB	r,m	1	1	1	
ADC, SBB	m,r	1	6	1	
ADC, SBB	m,i	1	6	1	
ADCX ADOX	r,r	1	1	1	ADX
CMP	r,r	1	1	0.25	
CMP	r,i	1	1	0.25	

OMP	l	1 4	l	0.5	
CMP	r,m	1		0.5	
CMP	m,i	1		0.5	
INC, DEC, NEG	r	1	1	0.25	
INC, DEC, NEG	m	1	6	1	
AAA, AAS		10	6		
DAA		16	8		
DAS		20	9		
AAD		4	5		
AAM		4	13		
MUL, IMUL	r8/m8	1	3	1	
MUL, IMUL	r16/m16	3	3	2	
MUL, IMUL	r32/m32	2	3	2	
MUL, IMUL	r64/m64	2	3	2	
IMUL	r,r	1	3	1	
IMUL	r,m	1		1	
IMUL	r16,r16,i	2	4	1	
IMUL	r32,r32,i	1	3	1	
IMUL	r64,r64,i	1	3	1	
IMUL	r16,m16,i	2		1	
IMUL	r32,m32,i	1		1	
IMUL	r64,m64,i	1	4	1	DAMO
MULX	r,r,r	2	4	2	BMI2
DIV	r8/m8	1	13-16	13-16	
DIV	r16/m16	2	14-21	14-21	depends on
DIV	r32/m32	2	14-30	14-30	operand values
DIV	r64/m64	2	14-46	14-45	
IDIV	r8/m8	1	13-16	13-16	
IDIV	r16/m16	2	13-21	14-22	
IDIV	r32/m32	2	14-30	14-30	
IDIV	r64/m64	2	14-47	14-45	
CBW		1	1	1	
CWDE, CDQE		1	1	0.5	
CDQ, CQO		1	1	0.25	
CWD		2	1	1	
Logic instructions					
AND, OR, XOR	r,r	1	1	0.25	
AND, OR, XOR	r,i	1	1	0.25	
AND, OR, XOR	r,m	1	-	0.5	
AND, OR, XOR	m,r	1	6	1	
AND, OR, XOR	m,i	1	6	1	
TEST	r,r	1	1	0.25	
TEST	r,i	1	1	0.25	
TEST	m,r	1		0.5	
TEST	m,i	1		0.5	
NOT	r	1	1	0.25	
NOT	m	1	6	1	
ANDN	r,r,r	1	1	0.25	BMI1
SHL, SHR, SAR	r,i/CL	1	1	0.25	
ROL, ROR	r,i/CL	1	1	0.25	
RCL	r,1	1	1	1	
RCL	r,i	9	4	4	
RCL	r,cl	9	4	4	

	1		ı		i
RCR	r,1	1	1	1	
RCR	r,i	7	3	3	
RCR	r,cl	7	3	3	
SHLD, SHRD	r,r,i	6	3	3	
SHLD, SHRD		7	3	3	
I	r,r,cl		3		
SHLD, SHRD	m,r,i/CL	8		3	
SARX	r,r,r	1	1	0.25	BMI2
SHLX	r,r,r	1	1	0.25	BMI2
SHRX	r,r,r	1	1	0.25	BMI2
RORX	r,r,i	1	1	0.25	BMI2
ВТ	r,r/i	1	1	0.25	
BT	m,i	1	•	0.5	
BT		5		3	
1	m,r		_		
BTC, BTR, BTS	r,r/i	2	2	0.5	
BTC, BTR, BTS	m,i	4		2	
BTC, BTR, BTS	m,r	8		3	
BSF	r,r	6	3	3	
BSF	r,m	8	4	4	
BSR	r,r	6	4	4	
BSR	r,m	8	4	4	
SETcc	r	1	1	0.5	
SETcc		1		2	
I	m	-			
CLC, STC		1		0.25	
CMC		1	1		
CLD		2		3	
STD		2		4	
POPCNT	r,r	1	1	0.25	SSE4.2
LZCNT	r,r	1	1	0.25	LZCNT
TZCNT	r,r	2	2	0.5	BMI1
BEXTR	r,r,r	1	1	0.25	BMI1
BLSI	r,r	2	2	0.5	BMI1
BLSMSK	r,r	2	2	0.5	BMI1
BLSR		2	2	0.5	BMI1
	r,r				
PDEP	r,r,r	6	18	18	BMI2
PEXT	r,r,r	7	18	18	BMI2
BZHI	r,r,r	1	1	0.25	BMI2
Control transfer instru					
JMP	short/near	1		2	
JMP	r	1		2	
JMP	m	1		2	
Jcc	short/near	1		0.5-2	2 if jumping
fused CMP+Jcc	short/near	1		0.5-2	2 if jumping
J(E/R)CXZ	short	1		0.5-2	2 if jumping
LOOP	short	1		2	2 if jumping
LOOPE LOOPNE	short	1		2	2 if jumping
CALL	near	2		2	Z ii juilipilig
		2		2	
CALL	r				
CALL	m	6		2	
RET		1		2	
RET	i	2		2	
BOUND	m	11		3	for no jump
INTO		4		2	for no jump
				'	

String instructions					
LODS		3		3	
REP LODS	m	6n		2n	
STOS		3		3	
REP STOS		1n		~1n	small n
REP STOS		3 per 16B		1 per 16B	best case
MOVS		5		3	
REP MOVS		~1n		~1n	small n
REP MOVS		4 pr 16B		1 per 16B	best case
SCAS		3		3	2001 00.00
REP SCAS		7n		2n	
CMPS		6		3	
REP CMPS		9n		3n	
Synchronization					
LOCK ADD	m,r	1	~17		
XADD	m,r	4	7		
LOCK XADD	m,r	4	~23		
CMPXCHG	m,r8	5	8		
CMPXCHG	m,r16	6	8		
CMPXCHG	m,r32/64	6	8		
LOCK CMPXCHG	m8,r8	5	~22		
LOCK CMPXCHG	m16,r16	6	~22		
LOCK CMPXCHG	m,r32/64	6	~22		
CMPXCHG8B	m64	18	8		
LOCK CMPXCHG8B	m64	18	~22		
CMPXCHG16B	m128	27	13		
LOCK CMPXCHG16B	m128	27	~21		
Other					
NOP (90)		1		0.2	
Long NOP (0F 1F)		1 1		0.2	
PAUSE		8		3	
ENTER	a,0	12		16	
ENTER	a,b	11+3b		~18+b	
LEAVE	,	2		3	
CPUID		37-50		125-133	
XGETBV				42	
RDTSC		37		36	
RDTSCP		64		64	rdtscp
RDPMC		20		20	. 3.50p
CRC32	r32,r8	3	3	3	
CRC32	r32,r16	3	3	3	
CRC32	r32,r32	3	3	3	
RDRAND RDSEED	r16/32	13	3	~1200	
RDRAND RDSEED	r64	19		~2500	
TOTALIA TOOLLO	107	10		2000	

Floating point x87 instructions

· · · · · · · · · · · · · · · · · · ·							
Instruction	Operands	Ops	_	Reciprocal throughput	Execution pipes	Domain, notes	
Move instructions							

FLD	r	1	1	1	P3
FLD	m32/64		6	1	P1
FLD	m80	8	7	4	
FBLD	m80	24	,	24	P2 P3
FST(P)	r	1	1	1	P3
FST(P)	m32/64	2	5	1	P2 P3
FSTP	m80	15	8	'	FZ F3
FBSTP	m80	274	0	- 115	P2 P3
FXCH		1	0	~145 0.25	P2 P3
	r	-			P3
FILD	m	1	8	1	
FIST(T)(P)	m	2	5		P2 P3
FLDZ, FLD1	a±0 =		2	1	P3
FCMOVcc	st0,r	7	3	3	P0 P1 P2 P3
FFREE	r	1	0	0.25	
FINCSTP, FDECSTP	A 3./	1	0	0.25	D0 D0
FNSTSW	AX	3		16	P2 P3
FNSTSW	m16	2		14	P2 P3
FLDCW	m16	1		2	P3
FNSTCW	m16	2		2	P2 P3
Arithmetic instructions	S				
FADD(P),FSUB(R)(P)	r/m	1	5	1	P0
FIADD, FISUB(R)	m	2		1	P0 P3
FMUL(P)	r/m	1	5	1	P0
FIMUL	m	2		1	P0 P3
FDIV(R)(P)	r	1	8-15	4-6	P3
FDIV(R)	m	1			P3
FIDIV(R)	m	2			P3
FABS, FCHS		1	1	1	P3
FCOM(P), FUCOM(P)	r/m	1		1	P0
FCOMPP, FUCOMPP		1		1	P0
FCOMI(P)	r	2		1	P2
FICOM(P)	m	2		1	P0 P3
FTST		1		1	P0
FXAM		1		1	P3
FRNDINT		1	4	3	
FPREM FPREM1		2		12-50	
Math					
FSQRT		1	8-21	4-10	P3
FLDPI, etc.		1	0-21	4-10	P3
FSIN		11-60	50-170	'	P0 P3
FCOS		55	50-170		FUFU
FSINCOS		80-140	60-120		P0 P3
FPTAN		11-52	~90	50-80	1013
FPATAN		11-32	50-160	45-150	
FSCALE		8	9	43-130	P0 P2 P3
FXTRACT		13	10	7	P0 P2 P3
F2XM1		10	~50	'	P2 P3
FYL2X		10-25	~50 ~50	~50	P2 P3 P0 P2 P3
FYL2XP1		69	~135	~30 ~135	P0 P2 P3
Other					

FNOP		1	0.25	
(F)WAIT		1	0.25	
FNCLEX		20	45	
FNINIT		34	85	
FNSAVE	m864	99	~160	
FRSTOR	m864	77	~130	

Integer vector instructions								
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes		
Move instructions								
MOVD	r32, mm	1	3	1				
MOVD	mm, r32	2	3	1				
MOVD	r32/64, x	1	3	1				
MOVD	x, r32/64	1	3	1				
MOVD	mm/x,m32	1	4	0.5				
MOVD	m32,mm/x	1	4	1	P2			
MOVQ	mm/x,mm/x	1	1	0.25	P0123			
MOVQ	mm/x,m64	1	4	0.5				
MOVQ	m64,mm/x	1	4	1	P2			
MOVDQA	x,x	1	0	0.25		renaming		
MOVDQA	x,m	1	3	0.5				
MOVDQA	m,x	1	4	1	P2			
VMOVDQA	y,y	2	1	0.5		lower half renamed		
VMOVDQA	y,m256	2	3	1				
VMOVDQA	m256,y	2	4	2	P2			
MOVDQU	x,x	1	0	0.25		renaming		
MOVDQU	x,m	1	3	0.5				
MOVDQU	m,x	1	4	1	P2			
LDDQU	x,m	1	3	0.5				
VMOVDQU	ymm,m256	2	3	1				
VMOVDQU	m256,ymm	2	4	2	P2			
MOVDQ2Q	mm,xmm	1	1	0.25	P0123			
MOVQ2DQ	xmm,mm	1	1	0.25	P0123			
MOVNTQ	m,mm	1	~900	1	P2			
MOVNTDQ	m,xmm	1	~900	1	P2			
MOVNTDQA	xmm,m	1	3	0.5				
PACKSSWB/DW	mm/x,r/m	1	1	0.5	P12			
PACKUSWB	mm/x,r/m	1	1	0.5	P12			
VPACKSSWB/DW	y,r/m	2	1	1	P12			
VPACKUSWB	y,r/m	2	1	1	P12			
PUNPCKH/LBW/WD/								
DQ	mm/x,r/m	1	1	0.5	P12	ivec		
PUNPCKL/HQDQ	xmm,r/m	1	1	0.5	P12			
VPUNPCKL/HQDQ	y,r/m	2	1	1	P12			
PSHUFB	mm/x,r/m	1	1	0.5	P12			
VPSHUFB	y,r/m	2	1	1	P12			
PSHUFD	x,x,i	1	1	0.5	P12			
VPSHUFD	y,y,i	2	1	1	P12			
PSHUFW	mm,mm,i	1	1	0.5	P12			
PSHUFL/HW	x,x,i	1	1	0.5	P12			

VPSHUFL/HW	y,y,i	2	1	1	P12	
PALIGNR	mm/x,r/m,i	1	1	0.5	P12	
VPALIGNR	y,r/m,i	2	1	1	P12	
PBLENDW	x,r/m,i	1	1	0.33	P013	SSE4.1
PBLENDW	y,r/m,i	2	1	0.67	P013	332
MASKMOVQ	mm,mm	30	~3000	~9	1 010	
MASKMOVDQU	-	60	~3000	~18		
	X,X				DO	
PMOVMSKB	r32,mm/x	1	3	1	P2	
VPMOVMSKB	r32,y	2	3	2	P2	
PEXTRB/W/D/Q	r,x/mm,i	2	3	1	P1 P2	SSE4.1
PINSRB/W/D/Q	x/mm,r,i	2	3	1	P12	
EXTRQ	x,i,i	2	6	4		AMD SSE4A
EXTRQ	x,x	1	4	4	P1	AMD SSE4A
INSERTQ	x,x,i,i	2	4	4	P0 P1	AMD SSE4A
INSERTQ	x,x	1	4	4	P1	AMD SSE4A
PMOVSXBW/BD/BQ/	7.,7.		-	·		72 3 3 2 1
WD/WQ/DQ	x,x	1	1	0.5	P12	SSE4.1
PMOVZXBW/BD/BQ/	Λ,Λ	•		0.0	1 12	0021.1
WD/WQ/DQ	x,x	1	1	0.5	P12	SSE4.1
VINSERTI128	y,y,x,i	2	1	.67	P013	AVX2
VINSERTI128		2	4	1	P013	AVX2 AVX2
1	y,y,m,i		4	'	P013	AVAZ
VPBROADCAST B/W/D/Q	x,x	1	1	0.5	P12	AVX2
VPBROADCAST	^,^	I I	'	0.5	F 12	AVA2
B/W/D/Q	x,m	1	4	1		AVX2
VPBROADCAST	χ,	•		•		7.77.2
B/W/D/Q	y,x	2	1	1	P12	AVX2
VPBROADCAST	, ,,,,	_				
B/W/D/Q	y,m	2		0.5		AVX2
VBROADCASTI128	y,m128	2	3	0.5		AVX2
VPGATHERDD	x,[r+s*x],x	38		13	P0 P1 P2	AVX2
VPGATHERDD	y,[r+s*y],y	66		20	P0 P1 P2	AVX2
VPGATHERQD	x,[r+s*x],x	24		9	P0 P1 P2	AVX2
VPGATHERQD	x,[r+s*y],x	36		12	P0 P1 P2	AVX2
VPGATHERQD						
,	x,[r+s*x],x	23		9	P0 P1 P2	AVX2
VPGATHERDQ	y,[r+s*x],y	35		12	P0 P1 P2	AVX2
VPGATHERQQ	x,[r+s*x],x	23		9	P0 P1 P2	AVX2
VPGATHERQQ	y,[r+s*y],y	35		12	P0 P1 P2	AVX2
Arithmetic instructions	3					
PADDB/W/D/Q/SB/SW/						
USB/USW	mm/x,r/m	1	1	0.33		ivec
VPADD	y,y,r/m	2	1	0.67		
PSUBB/W/D/Q/SB/SW/						
USB/USW	mm/x,r/m	1	1	0.33		
VPSUB	y,y,r/m	2	1	0.67		
PHADD/SUB(S)W/D	x,x	4	2	2	P0 P1 P2 P3	SSSE3
VPHADD/SUB(S)W/D	y,y,y	8	3	3	P0 P1 P2 P3	
PCMPEQ B/W/D	mm/x,r/m	1	1	0.33	P013	
VPCMPEQ B/W/D		2	2	0.53	P013	
	y,y,r/m					
PCMPEQQ	mm/x,r/m	1	1	0.5	P03	
VPCMPEQQ	y,y,r/m	2	2	1	P03	
PCMPGT B/W/D	mm/x,r/m	1	1	0.33	P013	
VPCMPGT B/W/D	y,y,r/m	2	2	0.67	P013	
PCMPGTQ	mm/x,r/m	1	1	1	P0	

VPCMPGTQ	y,y,r/m	2		2	P0	
PMULLW PMULHW						
PMULHUW PMULDQ						
PMULUDQ	mm/x,r/m	1	3	1	P0	
VPMULLW VPMULHW						
VPMULHUW						
VPMULDQ			2		DO	
VPMULUDQ	y,y,r/m	2	3	2	P0	0054.4
PMULLD	x,r/m	1	4	2	P0	SSE4.1
VPMULLD	y,y,r/m	2	4	4	P0	00050
PMULHRSW	mm/x,r/m	1	4	1	P0	SSSE3
VPMULHRSW	y,y,r/m	2	4	2	P0	
PMADDWD	mm/x,r/m	1	3	1	P0	
VPMADDWD	y,y,r/m	2	3	2	P0	
PMADDUBSW	mm/x,r/m	1	4	1	P0	
VPMADDUBSW	y,y,r/m	2	4	2	P0	
PAVGB/W	mm/x,r/m	1	1	0.5	P03	
VPAVGB/W	mm/x,r/m	2	1	1	P03	
PMIN/MAX SB/SW/ SD						
UB/UW/UD	mm/x,r/m	1	1	0.33	P013	
VPMIN/MAX SB/SW/						
SD UB/UW/UD	y,y,r/m	2	1	0.67	P013	
PHMINPOSUW	x,r/m	1	3	2		SSE4.1
PABSB/W/D	mm/x,r/m	1	1	0.5	P03	SSSE3
VPABSB/W/D	y,r/m	2	1	1	P03	
PSIGNB/W/D	mm/x,r/m	1	1	0.5	P03	SSSE3
VPSIGNB/W/D	y,r/m	2	1	1	P03	SSSE3
PSADBW	mm/x,r/m	1	3	1	P0	
VPSADBW	y,y,r/m	1	3	2	P0	
MPSADBW	x,x,i	4	4	2	P0 P1 P2	SSE4.1
VMPSADBW	y,y,y,i	8	4	3		
Logic						
PAND PANDN POR						
PXOR	mm/x,r/m	1	1	0.25	P0123	ivec
VPAND VPANDN						
VPOR VPXOR	y,y,r/m	2	1	0.5	P0123	
PSLL/RL W/D/Q						
PSRAW/D	mm/x,r/m	1	1	1	P2	
VPSLL/RL W/D/Q VP-						
SRAW/D	y,y,x/m	2	1	2	P2	
PSLL/RL W/D/Q						
PSRAW/D	mm/x,i	1	1	1	P2	
VPSLL/RL W/D/Q VP-					50	
SRAW/D	y,y,i	2	1	2	P2	
PSLLDQ, PSRLDQ	x,i	1	1	0.5	P12	
VPSLLDQ VPSRLDQ	y,y,i	2	1	1	P12	
VPSLLVD/Q						
VPSRAVD		_			D4	A) ()/O
VPSRLVD/Q	x,x,x	1	3	2	P1	AVX2
VPSLLVD/Q						
VPSRAVD VPSRLVD/Q	V V V	2	3	4	P1	AVX2
PTEST	y,y,y v.r/m				P1	SSE4.1
VPTEST	x,r/m	1 3	2 4	1 2	P1 P2	33E4.1
VFIESI	y,y/m	3	4	2	PIP2	

String instructions						
PCMPESTRI	x,x,i	6	8	3	P1 P2	SSE4.2
PCMPESTRI	x,m,i	12		4		SSE4.2
PCMPESTRM	x,x,i	7	8	3	P0 P1 P2	SSE4.2
PCMPESTRM	x,m,i	12		4		SSE4.2
PCMPISTRI	x,x,i	2	11	2	P1 P2	SSE4.2
PCMPISTRI	x,m,i	3		2		SSE4.2
PCMPISTRM	x,x,i	3	7	2	P1 P2	SSE4.2
PCMPISTRM	x,m,i	4		2		SSE4.2
Encryption						
PCLMULQDQ	x,x/m,i	4	4	2		pclmul
AESDEC	x,x	1	4	0.5	P01	aes
AESDECLAST	x,x	1	4	0.5	P01	aes
AESENC	X,X	1	4	0.5	P01	aes
AESENCLAST	X,X	1	4	0.5	P01	aes
AESIMC	X,X	1	4	0.5	P01	aes
AESKEYGENASSIST	x,x,i	1	4	0.5	P01	aes
SHA1RNDS4	x,x,i	1	6	4	P1	sha
SHA1NEXTE	x,x	1	1	1	P1	sha
SHA1MSG1	x,x	2	2	1	multi	sha
SHA1MSG2	x,x	1	1	0.5	P12	sha
SHA256RNDS2	x,x	1	4	2	P1	sha
SHA256MSG1	x,x	2	2	0.5	P0123	sha
SHA256MSG2	X,X	4	3	2	P0123	sha
Other						
EMMS		1		0.25		

Floating point XMM and YMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes
Move instructions						
MOVAPS/D MOVUPS/						
D	X,X	1	0	0.25	none	inherit
VMOVAPS/D	y,y	2	1	0.5		inherit
MOVAPS/D MOVUPS/						
D	x,m128	1	3	0.5		
VMOVAPS/D						
VMOVUPS/D	y,m256	2	5	1		
MOVAPS/D MOVUPS/						
D	m128,x	1	4	1	P2	
VMOVAPS/D						
VMOVUPS/D	m256,y	2	3	2	P2	
MOVSS/D	X,X	1	1	0.25	P0123	
MOVSS/D	x,m32/64	1	4	0.5		
MOVSS/D	m32/64,x	1	4	1	P2	
MOVHPS/D	x,m64	1	4	0.5	P12	
MOVLPS/D	x,m64	1	4	0.5	P12	
MOVHPS/D	m64,x	2	5	1	P1 P2	
MOVLPS/D	m64,x	1	4	1	P1 P2	

MOVLHPS MOVHLPS	x,x	1	1	0.5	P12	ivec
MOVMSKPS/D	r32,x	1	3	1	P2	1000
VMOVMSKPS/D	r32,y	1	3		P2	
MOVNTPS/D	m128,x	1	~950	1	P2	
VMOVNTPS/D	m256,y	2	~950	2	P2	
MOVNTSS/SD	m,x	1	330	4	P2	AMD SSE4A
SHUFPS/D	x,x/m,i	1	1	0.5	P12	ivec
VSHUFPS/D		2	1		P2	
VPERMILPS/PD	y,y,y/m,i	1	3	1 2	P2 P1	ivec
VPERMILPS/PD	x,x,x/m	2	4	4	P1	
	y,y,y/m	1	=		P12	
VPERMILPS/PD	x,x/m,i		1	0.5	P12	
VPERMILPS/PD	y,y/m,i	2	1	1		
VPERM2F128	y,y,y,i	8	3	3	P0 P1 P3	
VPERM2F128	y,y,m,i	12	_	4	P0 P1 P3	A) ///O
VPERMPS	у,у,у	3	5	4	P0 P1 P2	AVX2
VPERMPS	y,y,m	4	0	4	D0 D4 D0 D0	AVX2
VPERMPD	y,y,i	3	2	2	P0 P1 P2 P3	AVX2
VPERMPD	y,m,i	4		2	P0 P1 P2 P3	AVX2
BLENDPS/PD	x,x/m,i	1	1	0.5	P01	fp
VBLENDPS/PD	y,y,y/m,i	2	1	1	P01	
BLENDVPS/PD	x,x/m,xmm0	1	1	0.5	P01	
VBLENDVPS/PD	y,y,y/m,y	2	1	1	P01	
MOVDDUP	X,X	1	1	0.5	P12	ivec
MOVDDUP	x,m64	1		0.5		
VMOVDDUP	y,y	2	1	1	P12	
VMOVDDUP	y,m256	2		1		
VBROADCASTSS/D	X,X	1	1	0.5	P12	
VBROADCASTSS/D	y,x	2	1	1	P12	
VBROADCASTSS/D	x,m	2	3	0.5		
VBROADCASTSS	y,m32	2	3	1		
VBROADCASTF128	y,m128	2	4	5		
MOVSH/LDUP	x,x/m	1	1	0.5	P12	ivec
VMOVSH/LDUP	y,y/m	2		1	P12	
UNPCKH/LPS/D	x,x/m	1	1	0.5	P12	
VUNPCKH/LPS/D	y,y,y/m	2	1	1	P12	
EXTRACTPS	r32,x,i	2	3	1	P1 P2	
EXTRACTPS	m32,x,i	2	6	1	P1 P2	
VEXTRACTF128	x,y,i	1	1	0.33	P013	
VEXTRACTF128	m128,y,i	2	7	1	P01 P2	
INSERTPS	x,x,i	1	1	0.5	P12	ivec
INSERTPS	x,m32,i	1	4	1	P12	
VINSERTF128	y,y,x,i	2	1	0.5	P013	
VINSERTF128	y,y,m128,i	2	5	1	P013	
VMASKMOVPS/D	x,x,m128	1	4	1	P01	
VMASKMOVPS/D	y,y,m256	2	4	1	P01	
VMASKMOVPS/D	m128,x,x	19	~50	5	P1 P2	
VMASKMOVPS/D	m256,y,y	42	~50	11	P1 P2	
VGATHERDPS	x,[r+s*x],x	38		13	P0 P1 P2	AVX2
VGATHERDPS	y,[r+s*y],y	66		20	P0 P1 P2	AVX2
VGATHERQPS	x,[r+s*x],x	24		9	P0 P1 P2	AVX2
VGATHERQPS	x,[r+s*y],x	36		12	P0 P1 P2	AVX2
VGATHERDPD	x,[r+s*x],x	23		9	P0 P1 P2	AVX2
VGATHERDPD	y,[r+s*x],y	35		12	P0 P1 P2	AVX2

VGATHERQPD	x,[r+s*x],x	23		9	P0 P1 P2	AVX2
VGATHERQPD	y,[r+s*y],y	35		12	P0 P1 P2	AVX2
	J / L J J / J					
Conversion						
CVTPD2PS	X,X	1	3	1	P3	fp
VCVTPD2PS	x,y	2	5	2	P3	
CVTPS2PD	x,x	1	3	1	P3	
VCVTPS2PD	y,x	2	5	2	P3	
CVTSD2SS	x,x	1	3	1	P3	
CVTSS2SD	x,x	1	3	1	P3	
CVTDQ2PS	x,x	1	4	1	P3	
VCVTDQ2PS	y,y	2	4	2	P3	
CVT(T) PS2DQ	x,x	1	4	1	P3	mixed domain
VCVT(T) PS2DQ	у,у	2	4	2	P3	
CVTDQ2PD	X,X	2	6	1	P12 P3	
VCVTDQ2PD	y,x	4	6	2	P12 P3	
CVT(T)PD2DQ	X,X	2	6	1	P12 P3	
VCVT(T)PD2DQ	x,x x,y	4	6	2	P12 P3	
CVTPI2PS	x,y x,mm	2	6	1	P12 P3	
CVT(T)PS2PI		2	6	1	P12 P3	
CVT(1)P32P1 CVTPI2PD	mm,x	2	6		P12 P3	
	x,mm	2	6	1		
CVT(T) PD2PI	mm,x	1		1	P12 P3	
CVTSI2SS	x,r32	2	8	1	D0 D0	
CVT(T)SS2SI	r32,x	2	7	1	P2 P3	
CVTSI2SD	x,r32/64	2	8	1		
CVT(T)SD2SI	r32/64,x	2	7	1	P2 P3	
VCVTPS2PH	x/m,x,i	2	6	2	P1 P3	
VCVTPS2PH	x/m,y,i	4	6	2	P12 P3	
VCVTPH2PS	x,x/m	2	6	2	P1 P3	
VCVTPH2PS	y,x/m	4	6	2	P12 P3	
A with we atio						
Arithmetic ADDSS/D SUBSS/D		4	2	0.5	DOO	f
ADDPS/D SUBPS/D	x,x/m	1	3	0.5	P23	fp
ADDP5/D SUBP5/D	x,x/m	1	3	0.5	P23	fp
VADDPS/D VSUBPS/D	y,y,y/m	2	3	1	P23	fp
ADDSUBPS/D	x,x/m	1	3	0.5	P23	fp
VADDSUBPS/D	y,y,y/m	2	3	1	P23	
		_	7		D4 D0 D0	
HADDPS/D HSUBPS/D	X,X	4	7	2	P1 P2 P3	
VHADDPS/D VHSUBPS/D	y,y,y/m	8	7	3	P1 P2 P3	mixed domain
MULSS MULPS	x,x/m	1	3	0.5	P01	
MULSD MULPD	•	1	4	0.5	P01	fp
VMULPS	x,x/m	2	3		P01	fp
	y,y,y/m			1		fp fra
VMULPD	y,y,y/m	2	4	1	P01	fp
DIVSS DIVPS	x,x/m	1	10	3	P3	
VDIVPS	y,y,y/m	2	10	6	P3	
DIVSD DIVPD	x,x/m	1	8-13	4-5	P3	
VDIVPD	y,y,y/m	2	8-13	8-9	P3	
RCPSS/PS	x,x/m	1	5	1	P01	
VRCPPS	y,y/m	2	5	2	P01	
CMPSS/D						
CMPPS/D	x,x/m	1	1	0.5	P01	

VCMPPS/D	y,y,y/m	2	1	1	P01	
COMISS/D UCOMISS/D	x,x/m	2	4	1	P012	
MAXSS/SD/PS/PD MINSS/SD/PS/PD	x,x/m	1	1	0.5	P01	fp
VMAXPS/D VMINPS/D	y,y,y/m	2	1	1	P01	
ROUNDSS/SD/PS/PD	x,x/m,i	1	4	1	P3	fp
VROUNDSS/SD/PS/ PD	y,y/m,i	2	4	2	P3	fp
DPPS	x,x,i	8	15	4	P0 P1 P2 P3	SSE4.1
DPPS	x,m,i	10		5	P0 P1 P2 P3	SSE4.1
VDPPS	y,y,y,i	13	16	5	P0 P1 P2 P3	SSE4.1
VDPPS	y,m,i	14		5	P0 P1 P2 P3	SSE4.1
DPPD	x,x,i	3	10	3	P0 P1 P2 P3	SSE4.1
DPPD	x,m,i	5		4	P0 P1 P2 P3	SSE4.1
VFMADD132SS/SD	x,x,x/m	1	5	0.5	P01	FMA3
VFMADD132PS/PD	x,x,x/m	1	5	0.5	P01	FMA3
VFMADD132PS/PD	y,y,y/m	2	5	1	P01	FMA3
All other FMA3 instruction	ns: same as a	bove	1		P01	FMA3
VFMADDSS/SD	x,x,x,x/m	1	5	0.5	P01	Not officially
VFMADDPS/PD	x,x,x,x/m	1	5	0.5	P01	supported.
VFMADDPS/PD	y,y,y,y/m	2	5	1	P01	Don't use!
All other FMA4 instruction	ons: same as a	bove	1			
Math						
SQRTSS/PS	x,x/m	1	9-10	4-5	P3	fp
VSQRTPS	y,y/m	2	9-10	8-10		
SQRTSD/PD	x,x/m	1	14-15	4-8		
VSQRTPD	y,y/m	2	14-15	8-16		
RSQRTSS/PS	x,x/m	1	5	1	P01	
VRSQRTPS	y,y/m	2	5	2	P01	
Logic						
AND/ANDN/OR/XORPS/	x,x/m	1	1	0.25	P0123	fp
VAND/ANDN/OR/	λ,λ/111	'	'	0.20	1 0120	īΡ
XORPS/PD	y,y,y/m	2	1	0.5	P0123	
Other						
VZEROUPPER		10		4		32 bit mode
VZEROUPPER		17		6		64 bit mode
VZEROALL		18		6		32 bit mode
VZEROALL		33		11		64 bit mode
LDMXCSR	m32	1		16		
STMXCSR	m32	2		14		
FXSAVE	m4096	87		90		
FXRSTOR	m4096	121		140		
XSAVE	m	160		166		
XSAVEOPT	m	97		130		
XRSTOR	m	213		340		
XSAVEC	m	111		150		

AMD Zen 2

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the same

data. Similar instructions with or without V name prefix and with two or three oper-

ands behave the same unless otherwise noted.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, x = 128 bit xmm register, y = 256 bit ymm register, v = 4 any vector register, v = 4 memory operand including indirect operands, m64 means 64-bit

memory operand, etc.

Ops: Number of macro-operations issued from instruction decoder to schedulers.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. The latency listed does not include the memory operand where the listing for register and memory operand are joined (r/

m).

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/3 indicates that the execution units can handle 3 instructions per clock cycle in one thread. However, the throughput may be limited by other bottlenecks in the pipeline.

Execution pipe: Indicates which execution pipe or unit is used for the macro-operations:

P0: Floating point and vector pipe 0
P1: Floating point and vector pipe 1

P2: Floating point and vector pipe 2
P3: Floating point and vector pipe 3

P0 P1: Uses both P0 and P1 P01: Uses either P0 and P1

Where no unit is specified, it uses one or more integer pipe or address generation

units.

Two micro-operations can execute simultaneously if they go to different

execution pipes

Domain: Tells which execution unit domain is used:

ivec: integer vector execution unit. fp: floating point execution unit.

inherit: the output operand inherits the domain of the input operand.

There is an additional latency of 1 clock cycle if the output of an ivec instruction goes to the input of a fp instruction, and when the output of a fp instruction goes to the input of an ivec instruction. All other latencies after memory load and before

memory store instructions are included in the latency counts.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions						
MOV	r8,r8	1	1	0.3		
MOV	r16,r16	1	1	0.3		
MOV	r32,r32	1	0	0.25		renaming
MOV	r64,r64	1	0	0.25		renaming
MOV	r,i	1		0.25		
MOV	r,m8/16	1	4	0.5		
MOV	r,m32/64	1	0-4	0.5		may mirror
MOV	m,r8/16	1	3	1		
MOV	m,r32/64	1	0-3	1		may mirror

MOV	m:	4	1	4	
1	m,i	1	000	1	
MOVNTI	m,r	1	~900	1	
MOVZX, MOVSX	r,r	1	1	0.25	
MOVZX, MOVSX	r,m	1	3-4	0.5	
MOVSXD	r64,r32	1	1	0.25	
MOVSXD	r64,m32	1	3	0.5	
CMOVcc	r,r	1	1	0.3	
CMOVcc	r,m	1		0.5	
XCHG	r8,r8	2	1	1	
XCHG	r16,r16	2	1	1	
XCHG	r32,r32	2	0	0.33	renaming
XCHG	r64,r64	2	0	0.33	renaming
XCHG	r,m	2	~18	~18	depends on hw
XLAT	,,,,,	2	5	2	
PUSH	r	1		1	
PUSH	i	1		1	
PUSH	m	2		1	
	111	9		4	
PUSHF(D/Q)					
PUSH	sp	2		1	
PUSHA(D)		9		8	
POP	r	1		0.5	
POP	m	2		1	
POPF(D/Q)		35		13	
POPA(D)		9		4	
POP	sp	1	2	2	
LEA	r16,[m]	2	2	1	
LEA	r32/64,[m]	1	1	0.5	rip relative
LEA	r32/64,[m]	1	1-2	0.25	all other cases
LAHF		4	3	2	
SAHF		2	2	0.5	
SALC		1	1	1	
BSWAP	r	1	1	0.3	
MOVBE	r,[m]	1	'	0.5	MOVBE
MOVBE	[m],r	1		1	MOVBE
PREFETCHNTA		1		0.5	MOVBE
	m	•			
PREFETCHT0/1/2	m	1		0.5	DDEEETOLIM
PREFETCH/W	m	1		0.5	PREFETCHW
SFENCE		1		1	
LFENCE		1		11	
MFENCE		7		~78	
Arithmetic instructions	s				
ADD, SUB	r,r	1	1	0.3	
ADD, SUB	r,i	1	1	0.3	
ADD, SUB	r,m	1		0.5	
ADD, SUB	m8/16,r	1	7	1	
ADD, SUB	m32/64,r	1	1	1	mirrored in cpu
ADC, SBB	r,r	1	1	-	, in spa
ADC, SBB	r,i	1	1		
ADC, SBB	r,m	1	1	1	
ADC, SBB	m8/16,r/i	1	7	1	
ADC, SBB	m32/64,r/i	1	1	1	mirrored in cpu
		1	1		ADX
ADCX ADOX	r,r	I	1	1	ADX

CMP CMP CMP CMP INC, DEC, NEG INC, DEC, NEG INC, DEC, NEG AAA, AAS DAA DAS AAD AAM MUL, IMUL MUL, IMUL	r,r r,i r,m m,i r m8/16 m32/64 r8/m8 r16/m16 r32/m32	1 1 1 1 1 1 10 16 20 4 4 1 3	1 1 7 1 6 9 9 5 13 3 3	0.25 0.25 0.5 0.5 0.3 1 1	mirrored in cpu
MUL, IMUL MUL, IMUL IMUL IMUL IMUL IMUL IMUL IMUL IMUL	r64/m64 r,r r,m r16,r16,i r32,r32,i r64,r64,i r16,m16,i r32,m32,i r64,m64,i r,r,r r8/m8 r16/m16 r32/m32 r64/m64 r8/m8 r16/m16 r32/m32 r64/m64	2 1 1 2 1 1 2 1 2 1 2 2 2 1 2 2 2 1 1 2 2 1 2 2	3 3 3 4 3 3 3 4 12-15 13-20 13-28 13-44 1 1 1 1	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2-15 13-20 13-28 13-44 12-15 13-28 13-44 0.5 0.5 0.25 1	BMI2 depends on operand values
AND, OR, XOR AND, OR, XOR AND, OR, XOR AND, OR, XOR AND, OR, XOR AND, OR, XOR TEST TEST TEST TEST NOT NOT	r,r r,i r,m m8/16,r/i m32/64,r/i r,r r,i m,r m,i r m8/16 m32/64	1 1 1 1 1 1 1 1 1 1	1 1 7 1 1 1 7	0.3 0.3 0.5 1 1 0.25 0.25 0.5 0.5 0.3 1	mirrored in cou
NOT ANDN SHL, SHR, SAR	m32/64 r,r,r r,i/CL	1 1 1	1 1 1	1 0.25 0.3	mirrored in cpu BMI1

ROL, ROR	r,i/CL	1	1	0.3	
RCL	r,1	1	1	1	
RCL	r,i	9	4	4	
RCL	r,cl	9	4	4	
RCR		1	1	1	
	r,1				
RCR	r,i	7	3	3	
RCR	r,cl	7	3	3	
SHLD, SHRD	r,r,i	6	3	3	
SHLD, SHRD	r,r,cl	7	3	3	
SHLD, SHRD	m,r,i/CL	8		3	
SARX	r,r,r	1	1	0.25	BMI2
SHLX	r,r,r	1	1	0.25	BMI2
SHRX	r,r,r	1	1	0.25	BMI2
RORX	r,r,i	1	1	0.25	BMI2
BT	r,r/i	1	1	0.25	52
BT	m,i	1	'	0.5	
BT		5		3	
	m,r		_		
BTC, BTR, BTS	r,r/i	2	2	0.67	
BTC, BTR, BTS	m,i	4	9	2	
BTC, BTR, BTS	m,r	8	9	3	
BSF	r,r	6	3	3	
BSF	r,m	8	4	4	
BSR	r,r	6	4	4	
BSR	r,m	8	5	4	
SETcc	r	1	1	0.5	
SETcc	m	1		2	
CLC, STC		1		0.25	
CMC		1	1	1	
CLD		1	-	3	
STD		1		4	
POPCNT	rr	1	1	0.25	SSE4.2
LZCNT	r,r		1	0.25	LZCNT
1	r,r	1			
TZCNT	r,r	2	2	0.5	BMI1
BEXTR	r,r,r	1	1	0.25	BMI1
BLSI	r,r	2	2	0.5	BMI1
BLSMSK	r,r	2	2	0.5	BMI1
BLSR	r,r	2	2	0.5	BMI1
PDEP	r,r,r	6	19	19	BMI2
PEXT	r,r,r	7	19	19	BMI2
BZHI	r,r,r	1	1	0.25	BMI2
Control transfer instru	ctions				
JMP	short/near	1		2	
JMP	r	1		2	
JMP	m	1		2	
Jcc	short/near	1		0.5-2	
fused CMP+Jcc	short/near	1		0.5-2	0.5 predicted not
J(E/R)CXZ	short	1		0.5-2	taken. 2
LOOP	short	1		0.5-2	predicted taken.
LOOPE LOOPNE	short			0.5-2	Higher if mispredicted
		2			mispredicted
CALL	near			2	
CALL	r	2		2	
CALL	m	6		2	

RET		1		2	
RET	i	2		2	
BOUND	m	11		4	for no jump
INTO		4		2	for no jump
		7			ioi iio jailip
String instructions					
LODS		3		3	
REP LODS	m	6n		2n	
STOS		3		3	
REP STOS		3n		~1n	small n
REP STOS		2 per 16B		0.5 per 16B	best case
MOVS		5 per 10B		3	Dest case
REP MOVS		~1n		~1n	small n
REP MOVS		2 pr 16B		0.5 per 16B	best case
SCAS		3		3	Desi Case
REP SCAS		7n		2n	
		6		3	
CMPS					
REP CMPS		9n		3n	
Synchronization					
LOCK ADD	m,r	1	~17		
XADD	m,r	4	7		
LOCK XADD	m,r	4	~17		
CMPXCHG	m,r8	5	9		
CMPXCHG	m,r16/32/64	6	9		
LOCK CMPXCHG	m,r	5	~17		
CMPXCHG8B	m64	18	9		
LOCK CMPXCHG8B	m64	18	~17		
CMPXCHG16B	m128	27	14		
LOCK CMPXCHG16B	m128	27	~17		
LOCK CIVIPACHG 10B	111120	21	~17		
Other					
NOP (90)		1		0.2	
Long NOP (0F 1F)		1		0.2	
PAUSE		8		~65	
ENTER	a,0	12		18	
ENTER	a,b	11+3b		~20+b	
LEAVE	,	2		3	
CPUID		37-68		130-140	
XGETBV		38		48	
RDTSC		37		37	
RDTSCP		37		67	rdtscp
RDPMC		19		21	ı-
CRC32	r32,r8	1	3	3	
CRC32	r32,r16	1	3	3	
CRC32	r32,r32	1	3	3	
RDRAND RDSEED	r16/32	11	•	~1900	
RDRAND RDSEED	r64	17		~3700	
	1.5-			1 07 00	

Floating point x87 instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Domain, notes
	_	_		throughput	pipes	

Move instructions						
FLD	r	1	1	1	P3	
FLD	m32/64	1	6	1	P1	
FLD	m80	8	7	4		
FBLD	m80	24	'	24	P2 P3	
		1	4		P2 P3	
FST(P)	r		1 5	1		
FST(P)	m32/64	2	5	1	P2 P3	
FSTP	m80	15	8	455	B0 B0	
FBSTP	m80	274	_	~155	P2 P3	
FXCH	r	1	0	0.5		
FILD	m	1	8	1	P3	
FIST(T)(P)	m	2	5	1	P2 P3	
FLDZ, FLD1		1		1	P3	
FCMOVcc	st0,r	7	3	3	P0 P1 P2 P3	
FFREE	r	1	0	0.25		
FINCSTP, FDECSTP		1	0	0.25		
FNSTSW	AX	3		16	P2 P3	
FNSTSW	m16	2		14	P2 P3	
FLDCW	m16	1		2	P3	
FNSTCW	m16	2		2	P2 P3	
Arithmetic instructions	;					
FADD(P),FSUB(R)(P)	r/m	1	5	1	P0	
FIADD,FISUB(R)	m	2		1	P0 P3	
FMUL(P)	r/m	1	5	1	P0	
FIMUL	m	2		1	P0 P3	
FDIV(R)(P)	r	1	15	6	P3	
FDIV(R)	m	1			P3	
FIDIV(R)	m	2			P3	
FABS, FCHS		1	1	1	P3	
FCOM(P), FUCOM(P)	r/m	1	•	1	P0	
FCOMPP, FUCOMPP	17111	1		1	P0	
FCOMI(P)	r	2		1	P0 P2	
FICOM(P)	m	2		1	P0 P3	
FTST	111	1		1	P0	
FXAM		1		1	P3	
FRNDINT		1	3	3	13	
FPREM FPREM1		2	J	12-60		
				12-00		
Math						
FSQRT		1	22	8	P3	
FLDPI, etc.		1		1	P3	
FSIN		11-60	50-170	1	P0 P3	
FCOS		16-60	50-170		TUFU	
FSINCOS		11-85	75-140		P0 P3	
FPTAN		11-85	75-140 75-95		7073	
FPATAN		11-82	50-160	A	D0 D0 D0	
FSCALE		8	8	4	P0 P2 P3	
FXTRACT		13	8	8	P0 P2 P3	
F2XM1		10	55		P2 P3	
FYL2X		10-25	~50	~50	P0 P2 P3	
FYL2XP1		69	~145	~145	P0 P2 P3	

Other					
FNOP		1	0.25		
(F)WAIT		1	0.25		
FNCLEX		20	49		
FNINIT		34	92		
FNSAVE	m864	99	~186		
FRSTOR	m864	77	~140		

Integer vector inst	ructions					
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions						
MOVD	r32, mm	1	3	1		
MOVD	mm, r32	2	4	1		
MOVD/Q	r32/64, x	1	3	1	P2	
MOVD/Q	x, r32/64	1	3	1		
MOVD	mm/x,m32	1	4	0.5		
MOVD	m32,mm/x	1	4	1	P2	
MOVQ	mm/x,mm/x	1	1	0.25	P0123	
MOVQ	mm/x,m64	1	4	0.5		
MOVQ	m64,mm/x	1	4	1	P2	
MOVDQA	x,x	1	0	0.25		renaming
MOVDQA	x,m	1	4	0.5		, and the second
MOVDQA	m,x	1	4	1	P2	
VMOVDQA	y,y	1	0	0.25		renaming
VMOVDQA	y,m256	1	4	0.5		J
VMOVDQA	m256,y	1	4	1	P2	
MOVDQU	x,x	1	0	0.25		renaming
MOVDQU	x,m	1	4	0.5		J
MOVDQU	m,x	1	4	1	P2	
LDDQU	x,m	1	4	0.5		
VMOVDQU	y,m	1	4	0.5		
VMOVDQU	m,y	1	4	1	P2	
MOVDQ2Q	mm,x	1	1	0.25	P0123	
MOVQ2DQ	x,mm	1	1	0.25	P0123	
MOVNTQ	m,mm	1	~930	1	P2	
MOVNTDQ	m,x	1	~940	1	P2	
MOVNTDQA	x,m	1	5	0.5		
PACKSSWB/DW	mm/x,r/m	1	1	0.5	P12	
PACKUSWB	mm/x,r/m	1	1	0.5	P12	
VPACKSSWB/DW	v,r/m	1	1	0.5	P12	
VPACKUSWB	v,r/m	1	1	0.5	P12	
PUNPCKH/LBW/WD/	,					
DQ	v,r/m	1	1	0.5	P12	
PUNPCKL/HQDQ	v,r/m	1	1	0.5	P12	
PSHUFB	v,r/m	1	1	0.5	P12	
PSHUFD	v,v,i	1	1	0.5	P12	
PSHUFW	mm,mm,i	1	1	0.5	P12	
PSHUFL/HW	v,v,i	1	1	0.5	P12	
PALIGNR	v,r/m,i	1	1	0.5	P12	
PBLENDW	v,r/m,i	1	1	0.33	P013	

MASKMOVDQU	MASKMOVQ	mm mm	30	~2000	~9		
PMOVMSKB		mm,mm					
PMOVMSKB		· · · · · · · · · · · · · · · · · · ·				DO	
VPMOVMSKB							
PEXTRB/W/D/Q		•					
PINSRB.W/D/Q		_					
EXTRQ	· ·						
EXTRQ	· ·						AAAD 0054A
INSERTQ	· ·						
NSERTQ	· ·						
PMOVSXBW/BD/BQ/ WD/WQ/DQ	· ·						
WDMWQ/DQ		X,X	1	2	2	P1	AMD SSE4A
VPMOVSXBW/BD/BQ/ V, X	· ·		_	4	0.5	D40	
WDMQ/JQ		X,X	1	1	0.5	P12	
PMOVZXBW/BD/BQ/ WD/WQ/DQ			_	4	4	D40	
WD/WQ/DQ x,x 1 1 0.5 P12 VPMOVZXBW/BD/BQ/ WD/WQ/DQ y,x 2 4 1 P12 VINSERTI128 y,y,x,i 1 1 1 P2 AVX2 VPBROADCAST BW/D/Q x,x 1 1 0.5 P12 AVX2 VPBROADCAST BW/D/Q x,m 0.5 P12 AVX2 VPBROADCAST BW/D/Q y,x 2 4 1 P12 AVX2 VPBROADCAST BW/D/Q y,x 1 1 1 P2 AVX2 VPBROADCASTB BW/D/Q y,x 1 1 1 P2 AVX2 VPBROADCASTB BW/D/Q y,m 0.5 AVX2 VPBROADCASTB BW/D/Q y,m 0.5 AVX2 VPBROADCASTB WB/D/Q y,m 0.5 AVX2 VPBROADCASTB WB/D/Q y,m 0.5 AVX2 VPBGATHERDD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDD x,[r+s*x],x <t< td=""><td></td><td>y,x</td><td></td><td>4</td><td>1</td><td>P12</td><td></td></t<>		y,x		4	1	P12	
VPMOVZXBW/BD/BQ/ WD/WQ/DQ y,x 2 4 1 P12 P12 VINSERTI128 y,y,x,i 1 1 1 P2 AVX2 VINSERTI128 y,y,m,i 1 5 1 P2 AVX2 VPBROADCAST BW/D/Q x,x 1 1 0.5 P12 AVX2 VPBROADCAST BW/D/Q x,m 0.5 P12 AVX2 AVX2 VPBROADCAST BW/D/Q y,x 2 4 1 P12 AVX2 VPBROADCAST BW/D/Q y,x 1 1 1 P2 AVX2 VPBROADCAST BW/D/Q y,m128 1 4 0.5 AVX2 VPBROADCAST BW/D/Q y,m128 1 4 0.5 AVX2 VPBATHERDD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDD x,[r+s*x],x 20 6 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ <td></td> <td>V V</td> <td>4</td> <td>4</td> <td>0.5</td> <td>D12</td> <td></td>		V V	4	4	0.5	D12	
WDIWQ/DQ		X,X	'	'	0.5	FIZ	
VINSERTI128		V V	2	1	1	D12	
VINSERTI128 V,y,m,i VPBROADCAST NW/DQ VPBROADCAST NW/DQ VPBROADCAST NW/DQ VPBROADCAST NW/D VPBROADCAST N/T N N N N N N N N N N N N N N N N N		-					۸۱/۲۵
VPBROADCAST BW/ID/Q x,x 1 1 0.5 P12 AVX2 VPBROADCAST BW/ID/Q x,m 0.5 AVX2 VPBROADCAST BW/ID/Q y,x 2 4 1 P12 AVX2 VPBROADCASTQ BW/ID/Q y,x 1 1 1 P2 AVX2 VPBROADCAST BW/ID/Q y,m128 1 4 0.5 AVX2 VPBROADCASTI128 y,m128 1 4 0.5 AVX2 VPGATHERDD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDD x,[r+s*y],x 20 6 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*y],x 32 9 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 <				-			
B/W/D/Q		y,y,111,1		3	ı	FZ	AVAZ
VPBROADCAST BW/ID/Q x,m 0.5 AVX2 VPBROADCAST BW/ID y,x 2 4 1 P12 AVX2 VPBROADCASTQ VPBROADCAST BW/ID/Q y,x 1 1 1 P2 AVX2 VPBROADCAST BW/ID/Q y,m 0.5 AVX2 AVX2 AVX2 VBROADCASTI128 y,m128 1 4 0.5 AVX2 VPGATHERDD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDD y,[r+s*x],x 20 6 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ y,r 1 1		X.X	1	1	0.5	P12	AVX2
B/W/D/Q		,	-	-			
B/W/D		x,m			0.5		AVX2
VPBROADCASTQ y,x 1 1 1 P2 AVX2 VPBROADCAST BW/D/Q y,m 0.5 AVX2 AVX2 VBROADCASTI128 y,m128 1 4 0.5 AVX2 VPGATHERDD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDD y,[r+s*y],x 20 6 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*y],x 32 9 P0 P1 P2 AVX2 VPGATHERDQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 9 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 9 P0 P1 P2 AVX2 AVX2 VPGATHERQQ x,[r+s*x],x 19	VPBROADCAST						
VPBROADCAST BW/VD/Q y,m 0.5 AVX2 VBROADCASTI128 y,m128 1 4 0.5 AVX2 VPGATHERDD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDD y,[r+s*y],y 60 16 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*y],x 20 6 P0 P1 P2 AVX2 VPGATHERDQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERDQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 9 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 9 P0 P1 P2 AVX2 AVX2 VPGATHERQQ y,r 1 1 0.33 <td>B/W/D</td> <td>y,x</td> <td></td> <td></td> <td></td> <td></td> <td></td>	B/W/D	y,x					
BM/D/Q	VPBROADCASTQ	y,x	1	1	1	P2	AVX2
VBROADCASTI128 y,m128 1 4 0.5 AVX2 VPGATHERDD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDD y,[r+s*y],y 60 16 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*x],x 20 6 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*y],x 32 9 P0 P1 P2 AVX2 VPGATHERDQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 9 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 9 P0 P1 P2 AVX2 AVX2 VPGATHERQQ y,r 1<					0.5		A) ()/O
VPGATHERDD x,[r+s*x],x 32 9 P0 P1 P2 AVX2 VPGATHERDD y,[r+s*y],y 60 16 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*x],x 20 6 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*y],x 32 9 P0 P1 P2 AVX2 VPGATHERDQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 9 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 9 P0 P1 P2 AVX2 Arithmetic instructions N,r 1 1 0.5 P03 PSUBB/W/D/Q v,r 1 <td< td=""><td></td><td>-</td><td></td><td>4</td><td></td><td></td><td></td></td<>		-		4			
VPGATHERDD y, [r+s*y], y 60 16 P0 P1 P2 AVX2 VPGATHERQD x, [r+s*x], x 20 6 P0 P1 P2 AVX2 VPGATHERQD x, [r+s*y], x 32 9 P0 P1 P2 AVX2 VPGATHERDQ x, [r+s*x], x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y, [r+s*x], x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x, [r+s*x], x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x, [r+s*x], x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x, [r+s*x], x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x, [r+s*x], x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x, [r+s*x], x 19 9 P0 P1 P2 AVX2 VPGATHERQQ x, [r+s*x], x 19 9 P0 P1 P2 AVX2 Arithmetic instructions 1 1 0.5 P03 PADDB/W/D/Q v,r 1 <t< td=""><td></td><td> </td><td>· ·</td><td>4</td><td></td><td>D0 D4 D0</td><td></td></t<>			· ·	4		D0 D4 D0	
VPGATHERQD x,[r+s*x],x 20 6 P0 P1 P2 AVX2 VPGATHERQD x,[r+s*y],x 32 9 P0 P1 P2 AVX2 VPGATHERDQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*y],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*x],x 19 9 P0 P1 P2 AVX2 AVX2 VPGATHERQQ y,r 1 1 0.5 P0 P1 P2 AVX2 AVX2 VPGATHERQQ y,r 1 1 0.5 P03 PADDB/W/D/Q <							
VPGATHERQD x,[r+s*y],x 32 9 P0 P1 P2 AVX2 VPGATHERDQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERDQ y,[r+s*x],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*y],y 32 9 P0 P1 P2 AVX2 Arithmetic instructions PADDB/W/D/Q y,r 1 1 0.33 P013 PADDB/W/D/Q v,r 1 1 0.5 P03 PSUBB/W/D/Q v,r/m 1 1 0.5 P03 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 <t< td=""><td>_</td><td></td><td></td><td></td><td></td><td></td><td></td></t<>	_						
VPGATHERDQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERDQ y,[r+s*x],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*y],y 32 9 P0 P1 P2 AVX2 AVX2 VPGATHERQQ y,[r+s*x],x 19 6 P0 P1 P2 AVX2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 AVX2 VPGATHERQQ y,[r+s*y],y 32 9 P0 P1 P2 AVX2 AVX2 VPGATHERQQ y,r 1 1 0.5 P03 PADDB/W/D/Q V,r 1 1 0.5 P03 P0 P1 P2 PBADD/SUB(S)W/D V,r/m 1 1 0.5 P0 P1 P2							
VPGATHERDQ y,[r+s*x],y 32 9 P0 P1 P2 AVX2 VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*y],y 32 9 P0 P1 P2 AVX2 Arithmetic instructions AVX2 9 P0 P1 P2 AVX2 Arithmetic instructions AVX2 9 P0 P1 P2 AVX2 Arithmetic instructions AVX2 9 P0 P1 P2 AVX2 AVX2 AVX2 9 P0 P1 P2 AVX2 AVX2 AVX2 9 P0 P1 P2 AVX2 AVX2 AVX2 AVX2							
VPGATHERQQ x,[r+s*x],x 19 6 P0 P1 P2 AVX2 VPGATHERQQ y,[r+s*y],y 32 9 P0 P1 P2 AVX2 Arithmetic instructions PADDB/W/D/Q v,r 1 1 0.33 P013 PADDB/W/D/Q v,m 1 1 0.5 P03 PADD(U)SB/W v,r/m 1 1 0.33 P013 PSUBB/W/D/Q v,r 1 1 0.5 P03 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D y,y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPGT B/W/D v,v 1 1 0.33 P013	VPGATHERDQ		19		6	P0 P1 P2	AVX2
VPGATHERQQ y,[r+s*y],y 32 9 P0 P1 P2 AVX2 Arithmetic instructions PADDB/W/D/Q v,r 1 1 0.33 P013 PADDB/W/D/Q v,m 1 1 0.5 P03 PADD(U)SB/W v,r/m 1 1 0.5 P03 PSUBB/W/D/Q v,r/m 1 1 0.5 P03 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	VPGATHERDQ	y,[r+s*x],y	32		9	P0 P1 P2	AVX2
Arithmetic instructions PADDB/W/D/Q v,r 1 1 0.33 P013 PADDB/W/D/Q v,m 1 1 0.5 P03 PADD(U)SB/W v,r/m 1 1 0.5 P03 PSUBB/W/D/Q v,r 1 1 0.5 P03 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	VPGATHERQQ	x,[r+s*x],x	19		6	P0 P1 P2	AVX2
PADDB/W/D/Q v,r 1 1 0.33 P013 PADDB/W/D/Q v,m 1 1 0.5 P03 PADD(U)SB/W v,r/m 1 1 0.5 P03 PSUBB/W/D/Q v,r 1 1 0.33 P013 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	VPGATHERQQ	y,[r+s*y],y	32		9	P0 P1 P2	AVX2
PADDB/W/D/Q v,r 1 1 0.33 P013 PADDB/W/D/Q v,m 1 1 0.5 P03 PADD(U)SB/W v,r/m 1 1 0.5 P03 PSUBB/W/D/Q v,r 1 1 0.33 P013 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013							
PADDB/W/D/Q v,m 1 1 0.5 P03 PADD(U)SB/W v,r/m 1 1 0.5 P03 PSUBB/W/D/Q v,r 1 1 0.33 P013 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	Arithmetic instructions	5					
PADD(U)SB/W v,r/m 1 1 0.5 P03 PSUBB/W/D/Q v,r 1 1 0.33 P013 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	PADDB/W/D/Q	v,r	1	1	0.33	P013	
PSUBB/W/D/Q v,r 1 1 0.33 P013 PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	PADDB/W/D/Q	v,m	1	1	0.5		
PSUB(U)SB/W v,r/m 1 1 0.5 P03 PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	PADD(U)SB/W	v,r/m	1	1	0.5	P03	
PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	PSUBB/W/D/Q	v,r	1	1	0.33	P013	
PHADD/SUB(S)W/D mm,mm 3 2 2 P0 P1 P2 P3 PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	PSUB(U)SB/W		1	1	0.5	P03	
PHADD/SUB(S)W/D x,x 4 2 2 P0 P1 P2 P3 VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	` '		3	2		P0 P1 P2 P3	
VPHADD/SUB(S)W/D y,y,y 3 2 2 P0 P1 P2 P3 PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	` '		4	2	2	P0 P1 P2 P3	
PCMPEQ B/W/D v,v 1 1 0.33 P013 PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	` '						
PCMPEQQ v,v 1 1 0.5 P03 PCMPGT B/W/D v,v 1 1 0.33 P013	` '						
PCMPGT B/W/D v,v 1 1 0.33 P013			1				

	ı	ı	i	I.	I	1
PMULLW PMULHW						
PMULHUW PMULDQ				_		
PMULUDQ	V,V	1	3	1	P0	
PMULLD	V,V	1	4	2	P0	
PMULHRSW	V,V	1	4	1	P0	
PMADDWD	V,V	1	3	1	P0	
PMADDUBSW	V,V	1	4	1	P0	
PAVGB/W	V,V	1	1	0.5	P03	
PMIN/MAX SB/SW/ SD	,					
UB/UW/UD	V,V	1	1	0.33	P013	
PHMINPOSUW	x,x	1	3	2		
PABSB/W/D	V , V	1	1	0.5	P03	
PSIGNB/W/D	V,V V,V	1	1	0.5	P03	
PSADBW		1	3	1	P0	
	V,V			· ·		
MPSADBW	x,x,i	4	4	2	P0 P1 P2	
Logic						
PAND PANDN POR			4	0.05	D0400	
PXOR	V,V	1	1	0.25	P0123	
PSLL/RL W/D/Q			4		D 0	
PSRAW/D	V,V	1	1	1	P2	
VPSLL/RL W/D/Q VP-				_		
SRAW/D	v,v,i	1	1	1	P2	
PSLLDQ, PSRLDQ	v,i	1	1	0.5	P12	
VPSLLVD/Q						
VPSRAVD			_	_		
VPSRLVD/Q	V,V,V	1	3	2	P1	AVX2
PTEST	V,V	1	3	1	P2	
String instructions						
PCMPESTRI	x,x,i	6	11	3	P1 P2	SSE4.2
PCMPESTRI	x,m,i	12		4		SSE4.2
PCMPESTRM	x,x,i	7	7	3	P0 P1 P2	SSE4.2
PCMPESTRM	x,m,i	12		4		SSE4.2
PCMPISTRI	x,x,i	2	8	2	P1 P2	SSE4.2
PCMPISTRI	x,m,i	3		2		SSE4.2
PCMPISTRM		3	7	2	P1 P2	SSE4.2
PCMPISTRM	x,x,i v m i	4	1	2	FIFZ	SSE4.2
POMPISTRIM	x,m,i	4		2		33E4.2
Coomenties						
Encryption		_				
PCLMULQDQ	x,x/m,i	4	4	2	504	pclmul
AESDEC	X,X	1	4	0.5	P01	aes
AESDECLAST	X,X	1	4	0.5	P01	aes
AESENC	X,X	1	4	0.5	P01	aes
AESENCLAST	X,X	1	4	0.5	P01	aes
AESIMC	X,X	1	4	0.5	P01	aes
AESKEYGENASSIST	x,x,i	1	4	0.5	P01	aes
SHA1RNDS4	x,x,i	1	6	4	P1	sha
SHA1NEXTE	x,x	1	1	1	P1	sha
SHA1MSG1	X,X X,X	2	2	1	multi	sha
SHA1MSG2		1	1	0.5	P12	sha
	X,X	1		0.5		
SHA256RNDS2	X,X	1	4		P1	sha
SHA256MSG1	x,x	2	2	0.5	P0123	sha
SHA256MSG2	X,X	4	3	2	P0123	sha

Zen2

Other				
EMMS	1	0.25		

Floating point XMM and YMM instructions								
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes		
Move instructions					• •			
MOVAPS/D	V,V	1	0	0.25	none (rename)	inherit		
MOVUPS/D	V,V	1	0	0.25	none (rename)	inherit		
MOVAPS/D MOVUPS/D	x,m128	1	4	0.5				
VMOVAPS/D VMOVUPS/D	y,m256	1	5	0.5				
MOVAPS/D MOVUPS/D	m128,x	1	4	1	P2			
VMOVAPS/D VMOVUPS/D	m256,y	1	4	1	P2			
MOVSS/D	X,X	1	1	0.25	P0123			
MOVSS/D	x,m32/64	1	4	0.5				
MOVSS/D	m32/64,x	1	4	1	P2			
MOVHPS/D	x,m64	1	5	0.5	P12			
MOVLPS/D	x,m64	1	5	0.5	P12			
MOVHPS/D	m64,x	2	5	1	P1 P2			
MOVLPS/D	m64,x	1	4	1	P1 P2			
MOVLHPS MOVHLPS	x,x	1	1	0.5	P12	ivec		
MOVMSKPS/D	r32,x	1	3	1	P2			
VMOVMSKPS/D	r32,y	1	5	1	P2			
MOVNTPS/D	m,v	1	~950	1	P2			
MOVNTSS/SD	m,x	1		3	P2	AMD SSE4A		
SHUFPS/D	v,v/m,i	1	1	0.5	P12	ivec		
VPERMILPS/PD	v,v,v/m	1	3	2	P1			
VPERMILPS/PD	v,v/m,i	1	1	0.5	P12			
VPERM2F128	y,y,y/m,i	1	3	1	P2			
VPERMPS	y,y,y	2	8	2	P12	AVX2		
VPERMPS	y,y,m	3		2		AVX2		
VPERMPD	y,y,i	2	6	1	P12	AVX2		
VPERMPD	y,m,i	3		1		AVX2		
BLENDPS/PD	y,y,i	1	1	0.33	P013			
BLENDVPS/PD	x,x/m,xmm0	1	1	0.5	P01			
VBLENDVPS/PD	v,v,v/m,v	1	1	0.5	P01			
MOVDDUP	V,V	1	1	0.5	P12	ivec		
MOVDDUP	x,m64	1	9	0.5				
VMOVDDUP	y,m256	1		0.5				
VBROADCASTSS/D	X,X	1	1	0.5	P12			
VBROADCASTSS/D	y,x	2	4	1	P12			
VBROADCASTSS/D	v,m	1	4	0.5				
VBROADCASTF128	y,m128	1	4	0.5				
MOVSH/LDUP	V,V	1	1	0.5	P12	ivec		
UNPCKH/LPS/D	v,v/m	1	1	0.5	P12			
EXTRACTPS	r32,x,i	2	4	1	P1 P2			
EXTRACTPS	m32,x,i	2	5	1	P1 P2			
VEXTRACTF128	x,y,i	1	3	1	P2			
VEXTRACTF128	m128,y,i	2	6	1	P2			
INSERTPS	x,x,i	1	1	0.5	P12	ivec		

Zen2

INSERTPS VINSERTF128 VINSERTF128 VMASKMOVPS/D VMASKMOVPS/D VMASKMOVPS/D VGATHERDPS VGATHERDPS VGATHERQPS VGATHERQPS VGATHERDPD VGATHERDPD VGATHERDPD	x,m32,i y,y,x,i y,y,m128,i v,v,m m128,x,x m256,y,y x,[r+s*x],x y,[r+s*y],y x,[r+s*x],x x,[r+s*y],x x,[r+s*x],x y,[r+s*x],x	1 1 1 1 19 44 32 60 20 32 19 32 19	5 1 5 5-6 ~50 ~23	1 1 0.5 6 12 9 16 6 9	P12 P2 P2 P01	AVX2 AVX2 AVX2 AVX2 AVX2 AVX2 AVX2
VGATHERQPD	y,[r+s*y],y	32		9		AVX2 AVX2
	7,1, - 71,7					
Conversion						
CVTPD2PS	X,X	1	3	1	P3	fp
VCVTPD2PS	x,y	2	5	2	P2 P3	
CVTPS2PD	X,X	1	3	1	P3	
VCVTPS2PD	y,x	2	5 3	2	P2 P3	
CVTSD2SS CVTSS2SD	X,X	1 1	3	1 1	P3 P3	
CVT3323D CVTDQ2PS	X,X V,V	1	3	1	P3	
CVTDQ2F3 CVT(T) PS2DQ	V,V V,V	1	3		P3	mixed domain
CVTDQ2PD	X,X	1	3		P3	IIIIXCU UOIIIAIII
VCVTDQ2PD	y,x	2	5	1 1	P2 P3	
CVT(T)PD2DQ	x,x	1	3	1	P3	
VCVT(T)PD2DQ	x,y	2	5	1	P2	P3
CVTPI2PS	x,mm	3	6	1	P1 P2 P3	
CVT(T)PS2PI	mm,x	1	3	1	P3	
CVTPI2PD	x,mm	3	6	1	P1 P2 P3	
CVT(T) PD2PI	mm,x	2	4	1	P12 P3	
CVTSI2SS	x,r32	2	7	1		
CVT(T)SS2SI	r32,x	2	6	1	P2 P3	
CVTSI2SD	x,r32/64	2	7	1		
CVT(T)SD2SI	r32/64,x	2	6	1	P2 P3	
VCVTPS2PH	x/m,x,i	1	3	1	P3	
VCVTPU2PS	x/m,y,i	2	5	1	P2 P3	
VCVTPH2PS VCVTPH2PS	x,x/m	1 2	3 5	1 1	P3 P2 P3	
VCVIPHZPS	y,x/m		5	l '	P2 P3	
Arithmetic						
ADDSS/D SUBSS/D	x,x/m	1	3	0.5	P23	fp
ADDPS/D SUBPS/D	v,v/m	1	3	0.5	P23	fp
ADDSUBPS/D	v,v/m	1	3	0.5	P23	fp
HADD/SUBPS/D	x,x	4	7	2	P1 P2 P3	
VHADD/SUBPS/D	y,y,y/m	3	7	2	P1 P2 P3	mixed domain
MULSS MULSD	x,x/m	1	3	0.5	P01	fp
MULPS MULPD	v,v/m	1	3	0.5	P01	fp
DIVSS DIVPS	v,v/m	1	10	3	P3	
DIVSD DIVPD	v,v/m	1	13	5	P3	
RCPSS/PS	v,v/m	1	5 1	1	P01	
CMPSS/SD/PS/PD	v,v/m	1	I	0.5	P01	

Zen2

(U)COMISS/SD	x,x/m	2	5	1	P012	
MAXSS/SD/PS/PD MINSS/SD/PS/PD	v,v/m	1	1	0.5	P01	fp
ROUNDSS/SD/PS/PD	v,v/m,i	1	3	1	P3	fp
DPPS	x,x,i	8	15	4	P0 P1 P2 P3	SSE4.1
VDPPS	y,y,y,i	7	15	4	P0 P1 P2 P3	0021.1
DPPD	x,x,i	3	9	3	P0 P1 P2 P3	SSE4.1
VFMADD132PS/PD	v,v,v/m	1	5	0.5	P01	FMA3
All other FMA3 instruction		bove			P01	FMA3
						-
Math						
SQRTSS/PS	v,v/m	1	14	6	P3	
SQRTSD/PD	v,v/m	1	20	9	P3	
RSQRTSS/PS	x,x/m	1	5	1	P01	
VRSQRTPS	y,y/m	1	3-4	1	P01	
Logic						
AND(N)/OR/XORPS/PD	v,v/m	1	1	0.25	P0123	fp
Other						
VZEROUPPER		1		0.25		
VZEROALL		10		4		32 bit mode
VZEROALL		18		6		64 bit mode
LDMXCSR	m32	1		17		
STMXCSR	m32	2		16		2011
FXSAVE	m4096	85		77		32 bit mode
FXSAVE	m4096	93		85		64 bit mode
FXRSTOR	m4096	137		182		32 bit mode
FXRSTOR	m4096	142		182		64 bit mode
XSAVE, XSAVEOPT	m	110		103		32 bit mode
XSAVE, XSAVEOPT	m	112		104		64 bit mode
XRSTOR	m	182		216		32 bit mode
XRSTOR	m	162		184		64 bit mode
XSAVEC	m	295		320		32 bit mode
XSAVEC	m	277		290		64 bit mode

AMD Bobcat

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB,

JNE, etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, xmm = 128 bit xmm register, m = any memory operand including

indirect operands, m64 means 64-bit memory operand, etc.

Ops: Number of micro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 micro-operations are micro-coded.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latencies listed do not include memory operands where the oper-

and is listed as register or memory (r/m).

The clock frequency varies dynamically, which makes it difficult to measure latencies. The values listed are measured after the execution of millions of similar instructions, assuming that this will make the processor boost the clock frequency

to the highest possible value.

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/2 indicates that the execution units can handle 2 instructions per clock cycle in one thread. However, the throughput may be limited by other bottlenecks in the pipe-

line.

Execution pipe: Indicates which execution pipe is used for the micro-operations. I0 means integer

pipe 0. I0/1 means integer pipe 0 or 1. FP0 means floating point pipe 0 (ADD). FP1 means floating point pipe 1 (MUL). FP0/1 means either one of the two floating point pipes. Two micro-operations can execute simultaneously if they go to

different execution pipes.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes
Move instructions						
MOV	r,r	1	1	0.5	10/1	
MOV	r,i	1		0.5	10/1	
MOV	r,m	1	4	1	AGU	Any addr. mode
MOV	m,r	1	4	1	AGU	Any addr. mode
MOV	m8,r8H	1	7	1	AGU	AH, BH, CH, DH
MOV	m,i	1		1	AGU	
MOVNTI	m,r	1	6	1	AGU	
MOVZX, MOVSX	r,r	1	1	0.5	10/1	
MOVZX, MOVSX	r,m	1	5	1		
MOVSXD	r64,r32	1	1	0.5		
MOVSXD	r64,m32	1	5	1		
CMOVcc	r,r	1	1	0.5	10/1	
CMOVcc	r,m	1		1		
XCHG	r,r	2	1	1	10/1	
XCHG	r,m	3	20			Timing dep. on hw
XLAT		2	5			
PUSH	r	1		1		

			Вор	Jul		
PUSH	i	1		1 1		
PUSH	m	3		2		
PUSHF(D/Q)		9		6		
				9		
PUSHA(D)		9				
POP	r	1		1		
POP	m	4		4		
POPF(D/Q)		29		22		
POPA(D)		9		8		
LEA	r16,[m]	2	3	2	10	Any address size
LEA	r32/64,[m]	1	1	0.5	10/1	no scale, no offset
LEA	r32/64,[m]	1	2-4	1	10	w. scale or offset
LEA	1	1	Z- -	0.5	10/1	RIP relative
	r64,[m]		_		10/1	RIF Telative
LAHF		4	4	2	10/4	
SAHF		1	1	0.5	10/1	
SALC		1	1			
BSWAP	r	1	1	0.5	10/1	
PREFETCHNTA	m	1		1	AGU	
PREFETCHT0/1/2	m	1		1	AGU	
PREFETCH	m	1		1	AGU	AMD only
SFENCE		4		~45	AGU	, ,
LFENCE		1		1	AGU	
MFENCE		4		~45	AGU	
MIFENCE		4		743	AGU	
A .::414:- :4:						
Arithmetic instruction			_	0.5	10/4	
ADD, SUB	r,r/i	1	1	0.5	10/1	
ADD, SUB	r,m	1		1		
ADD, SUB	m,r	1		1		
ADC, SBB	r,r/i	1	1	1	10/1	
ADC, SBB	r,m	1		1		
ADC, SBB	m,r/i	1	6-7			
CMP	r,r/i	1	1	0.5	10/1	
CMP	r,m	1		1		
INC, DEC, NEG	r	1	1	0.5	10/1	
INC, DEC, NEG	m .	1	6		, .	
AAA		9	5			
AAS		9	10			
DAA		12	7			
DAS		16	8			
AAD		4	5			
AAM		33	23	23		
MUL, IMUL	r8/m8	1	3	1	10	
MUL, IMUL	r16/m16	3	3-5		10	latency ax=3, dx=5
MUL, IMUL	r32/m32	2	3-4	2	10	latency eax=3, edx=4
MUL, IMUL	r64/m64	2	6-7		10	latency rax=6, rdx=7
IMUL	r16,r16/m16	1	3	1	10	
IMUL	r32,r32/m32	1	3	1	10	
IMUL	r64,r64/m64	1	6	4	10	
IMUL	r16,(r16),i	2	4	3	10	
IMUL	r32,(r32),i	1	3	1	10	
IMUL	r64,(r64),i	1	7	4	10	
DIV	r8/m8		27		10	
		1		27		
DIV	r16/m16	1	33	33	10	
DIV	r32/m32	1	49	49	10	

DIV		a	04	04	10	1
DIV	r64/m64	1	81	81	10	
IDIV	r8/m8	1	29	29	10	
IDIV	r16/m16	1	37	37	10	
IDIV	r32/m32	1	55	55	10	
IDIV	r64/m64	1	81	81	10	
CBW, CWDE, CDQE		1	1		10/1	
CWD, CDQ, CQO		1	1		I0/1	
Logic instructions						
AND, OR, XOR	r,r	1	1	0.5	10/1	
AND, OR, XOR	r,m	1		1		
AND, OR, XOR	m,r	1		1		
TEST	r,r	1	1	0.5	10/1	
TEST	r,m	1		1		
NOT	r	1	1	0.5	10/1	
NOT	m	1		1		
SHL, SHR, SAR	r,i/CL	1	1	0.5	IO/1	
ROL, ROR	r,i/CL	1	1	0.5	10/1	
RCL, RCR	r,1	1	1	1	10/1	
RCL	r,i	9	5	5		
RCR	r,i	7	4	4		
RCL	r,CL	9	6	5		
RCR	r,CL	9	5	4		
SHL,SHR,SAR,ROL,	•					
ROR	m,i /CL	1	7	1		
RCL, RCR	m,1	1	7	1		
RCL	m,i	10		~15		
RCR	m,i	9	18	~14		
RCL	m,CL	9		15		
RCR	m,CL	8		15		
SHLD, SHRD	r,r,i	6	3	3		
SHLD, SHRD	r,r,cl	7	4	4		
SHLD, SHRD	m,r,i/CL	8	18	15		
BT	r,r/i	1		0.5		
BT	m,i	1		1		
BT	m,r	5		3		
BTC, BTR, BTS	r,r/i	2	2	1		
BTC	m,i	5		15		
BTR, BTS	m,i	4-5		15		
BTC	m,r	8	16	13		
BTR, BTS	m,r	8	15	15		
BSF, BSR	r,r	11	6	6		
BSF, BSR	r,m	11		6		
POPCNT	r,r/m	9	12	5		SSE4.A/SSE4.2
LZCNT	r,r/m	8	5			SSE4.A, AMD only
SETcc	r	1	1	0.5		
SETcc	m	1		1		
CLC, STC		1		0.5	10/1	
CMC		1	1	0.5	10/1	
CLD		1		1	10	
STD		2		2	10,11	
Control transfer instru	ctions					

JMP	short/near	1		2		
JMP	r	1		2		
JMP	m(near)	1		2		
Jcc	short/near	1		1/2 - 2		recip. t. = 2 if jump
J(E/R)CXZ	short	2		1 - 2		recip. t. = 2 if jump
LOOP	short	8		4		
CALL	near	2		2		
CALL	r	2		2		
CALL	m(near)	5		2		
RET		1		~3		
RET	i	4		~4		
BOUND	m	8		4		values for no jump
INTO		4		2		values for no jump
String instructions						
LODS		4		~3		
REP LODS		5		~3		values are per count
STOS		4		2		
REP STOS		2				best case 6-7 B/clk
MOVS		7		5		
REP MOVS		2				best case 5 B/clk
SCAS		5		3		
REP SCAS		6		3		values are per count
CMPS		7		4		
REP CMPS		6		3		values are per count
Other						
NOP (90)		1	0	0.5	10/1	
Long NOP (0F 1F)		1	0	0.5	10/1	
PAUSE		6		6	10, 1	
ENTER		i,0	12		36	
ENTER		a,b	10+6b		34+6b	
LEAVE		2		3		32 bit mode
CPUID		30-52	70-830			
RDTSC		26		87		
RDPMC		14		8		

Floating point x87 instructions

loating point xor instructions									
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes			
Move instructions									
FLD	r	1	2	0.5	FP0/1				
FLD	m32/64	1	6	1	FP0/1				
FLD	m80	7	14	5					
FBLD	m80	21	30	35					
FST(P)	r	1	2	0.5	FP0/1				
FST(P)	m32/64	1	6	1	FP1				
FSTP	m80	16	19	9					
FBSTP	m80	217	177	180					
FXCH	r	1	0	1	FP1				
FILD	m	1	9	1	FP1				
FIST(T)(P)	m	1	6	1					

FLDZ, FLD1		1		1	FP1
FCMOVcc	st0,r	12	7	7	FP0/1
FFREE	r	1		1	FP1
FINCSTP, FDECSTP		1	1	1	FP1
FNSTSW	AX	2	~20	10	FP1
FNSTSW	m16	2	~20	10	FP1
FNSTCW	m16	3		2	FP0
FLDCW	m16	12		10	FP1
LDOVV	11110	'2		10	
Arithmetic instruction	 e				
FADD(P),FSUB(R)(P)	r	1	3	1	FP0
FADD(P),FSUB(R)(P)	m '	1	3	1	FP0
FIADD,FISUB(R)	m	2	3	3	FP0,FP1
FMUL(P)	r	1	5	3	FP1
` '		1	5	3	FP1
FMUL(P)	m 		5	3	
FIMUL	m	2	40	40	FP1
FDIV(R)(P)	r	1	19	19	FP1
FDIV(R)(P)	m	1		19	FP1
FIDIV(R)	m	2		19	FP1
FABS, FCHS		1	2	2	FP1
FCOM(P), FUCOM(P)	r	1		1	FP0
FCOM(P), FUCOM(P)	m	1		1	FP0
FCOMPP, FUCOMPP		1		1	FP0
FCOMI(P)	r	1	2	2	FP0
FICOM(P)	m	2		1	FP0, FP1
FTST		1		1	FP0
FXAM		2		2	FP1
FRNDINT		5	11		FP0, FP1
FPREM		1	11-16		FP1
FPREM1		1	11-19		FP1
Math					
FSQRT		1	31		FP1
FLDPI, etc.		1		1	FP0
FSIN		4-44	27-105	27-105	FP0, FP1
FCOS		11-51	51-94	51-94	FP0, FP1
FSINCOS		11-75	48-110	48-110	FP0, FP1
FPTAN		~45	~113	~113	FP0, FP1
FPATAN		9-75	49-163	49-163	FP0, FP1
FSCALE		5	8	40 100	FP0, FP1
FXTRACT		7	9		FP0, FP1
F2XM1		30-56	~60		FP0, FP1
FYL2X		8	29		FP0, FP1
FYL2XP1		12	44		FP0, FP1
FILZAFI		12	44		FFU, FF I
Other					
FNOP	_	1	0	0.5	FP0, FP1
(F)WAIT		1	0	0.5	ALU
FNCLEX		9		30	FP0, FP1
FNINIT				78	· ·
		26			FP0, FP1
FNSAVE	m m	85		163	FP0, FP1
FRSTOR	m 	80		123	FP0, FP1
FXSAVE	m	71		105	FP0, FP1

FXRSTOR	m	111	118	FP0. FP1	
	111		1 110	I FULLE I	

Integer MMX and XMM instructions

	nteger MMX and XMM instructions								
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes			
Move instructions									
MOVD	r32, mm	1	7	1	FP0				
MOVD	mm, r32	1	7	3	FP0/1				
MOVD	mm,m32	1	5	1	FP0/1				
MOVD	r32, xmm	1	6	1	FP0				
MOVD	xmm, r32	3	6	3	FP1				
MOVD	xmm,m32	2	5	1	FP1				
MOVD	m32,mm/x	1	6	2	FP1				
	, ,					Moves 64 bits.			
MOVD (MOVQ)	r64,mm/x	1	7	1	FP0	Name differs			
MOVD (MOVQ)	mm,r64	2	7	3	FP0/1	do.			
MOVD (MOVQ)	xmm,r64	3	7	3	FP0/1	do.			
MOVQ	mm,mm	1	1	0.5	FP0/1				
MOVQ	xmm,xmm	2	1	1	FP0/1				
MOVQ	mm,m64	1	5	1	FP0/1				
MOVQ	xmm,m64	2	5	1	FP1				
MOVQ	m64,mm/x	1	6	2	FP1				
MOVDQA	xmm,xmm	2	1	1	FP0/1				
MOVDQA		2	6	2	AGU				
MOVDQA	xmm,m	2	6	3	FP1				
	m,xmm	2	6-9	3 2-5.5	AGU				
MOVDQU, LDDQU	xmm,m	1	1						
MOVDQU	m,xmm	2	6-9	3-6	FP1				
MOVDQ2Q	mm,xmm	1	1 1	0.5	FP0/1				
MOVQ2DQ	xmm,mm	2	1 1	1	FP0/1				
MOVNTQ	m,mm	1	13	1.5	FP1				
MOVNTDQ	m,xmm	2	13	3	FP1				
PACKSSWB/DW	,	_		0.5	ED0/4				
PACKUSWB	mm,r/m	1	1	0.5	FP0/1				
PACKSSWB/DW				0	ED0/4				
PACKUSWB	xmm,r/m	3	2	2	FP0/1				
PUNPCKH/LBW/WD/	nana r/na	1	4	0.5					
DQ	mm,r/m	1	1	0.5					
PUNPCKH/LBW/WD/ DQ	xmm,r/m	2	1	1					
PUNPCKHQDQ	xmm,r/m	2	1 1	1	FP0, FP1				
PUNPCKLQDQ	xmm,r/m	1	1 1	0.5	FP0/1				
PSHUFB		1	2	1	FP0/1	Suppl. SSE3			
PSHUFB	mm,mm		3	3		1 ''			
	xmm,xmm	6			FP0/1	Suppl. SSE3			
PSHUFD	xmm,xmm,i	3	2	2	FP0/1				
PSHUFW	mm,mm,i	1	1	0.5	FP0/1				
PSHUFL/HW	xmm,xmm,i	2	2	2	FP0/1	0			
PALIGNR	xmm,xmm,i	20	19	12	FP0/1	Suppl. SSE3			
MASKMOVQ	mm,mm	32	146-1400	130-1170	FP0, FP1				
MASKMOVDQU	xmm,xmm	64	279-3000	260-2300	FP0, FP1				
PMOVMSKB	r32,mm/x	1	8	2	FP0				
PEXTRW	r32,mm/x,i	2	12	2	FP0, FP1				
PINSRW	mm,r32,i	2	10	6	FP0/1				
PINSRW	xmm,r32,i	3	10		FP0/1				

INSERTQ	xmm,xmm	3	3-4	3	FP0, FP1	SSE4.A, AMD only
INSERTQ	xmm,xmm,i,i	3	3-4	3	FP0, FP1	SSE4.A, AMD only
EXTRQ	xmm,xmm	1	1	1	FP0/1	SSE4.A, AMD only
EXTRQ	xmm,xmm,i,i	1	2	2	FP0/1	SSE4.A, AMD only
Arithmetic instruction	S					
PADDB/W/D/Q						
PADDSB/W						
PADDUSB/W PSUBB/						
W/D/Q PSUBSB/W	mm,r/m	1	1	0.5	FP0/1	
PSUBUSB/W PADDB/W/D/Q	111111,1/111	J	I	0.5	FF0/I	
PADDSB/W/D/Q PADDSB/W ADDUSB/						
W PSUBB/W/D/Q						
PSUBSB/W						
PSUBUSB/W	xmm,r/m	2	1	1	FP0/1	
PHADD/SUBW/SW/D	mm,r/m	1	1	0.5	FP0/1	Suppl. SSE3
PHADD/SUBW/SW/D	xmm,r/m	2	4	1	FP0/1	Suppl. SSE3
PCMPEQ/GT B/W/D	mm,r/m	1	1	0.5	FP0/1	
PCMPEQ/GT B/W/D	xmm,r/m	2	1	1	FP0/1	
PMULLW PMULHW						
PMULHUW						
PMULUDQ	mm,r/m	1	2	1	FP0	
PMULLW PMULHW						
PMULHUW)	2	_	2	ED0	
PMULUDQ	xmm,r/m	2 1	2 2	2	FP0	Cumpl CCE2
PMULHRSW	mm,r/m	2	2	1 2	FP0 FP0	Suppl. SSE3
PMULHRSW	xmm,r/m		2	1	FP0	Suppl. SSE3
PMADDWD PMADDWD	mm,r/m	1 2	2	2	FP0 FP0	
PMADDUBSW	xmm,r/m mm,r/m	1	2	1	FP0	Suppl. SSE3
	1 ' 1	2	2	2		l
PMADDUBSW	xmm,r/m	1	1	0.5	FP0 FP0/1	Suppl. SSE3
PAVGB/W PAVGB/W	mm,r/m	2		0.5	FP0/1 FP0/1	
PMIN/MAX SW/UB	xmm,r/m mm,r/m	1	1 1	· ·	FP0/1 FP0/1	
PMIN/MAX SW/UB	'	2	1	0.5	FP0/1	
PABSB/W/D	xmm,r/m mm,r/m	1		0.5	FP0/1	Suppl. SSE3
PABSB/W/D	xmm,r/m	2		1	FP0/1	Suppl. SSE3
PSIGNB/W/D	mm,r/m	1	1	0.5	FP0/1	Suppl. SSE3
PSIGNB/W/D	xmm,r/m	2	1 1	1	FP0/1	Suppl. SSE3
PSADBW	mm,r/m	1	2	2	FP0	очррі. оодо
PSADBW	xmm,r/m	2	2	2	FP0, FP1	
ו אממאא	AIIIII,I/III	۷		_	וויט, רדו	
Logic						
PAND PANDN POR						
PXOR	mm,r/m	1	1	0.5	FP0/1	
PAND PANDN POR						
PXOR	xmm,r/m	2	1	1	FP0/1	
PSLL/RL W/D/Q						
PSRAW/D	mm,i/mm/m	1	1	1	FP0/1	
PSLL/RL W/D/Q		^			ED0//	
PSRAW/D	xmm,i/xmm/m	2	1	1	FP0/1	
PSLLDQ, PSRLDQ	xmm,i	2	1	1	FP0/1	
Othor						
Other	J		l			

EMMS	1	0.5	FP0/1	
		0.5	FFU/I	

Floating point XMM instructions

loating point XMM instructions									
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes			
Move instructions									
MOVAPS/D	r,r	2	1	1	FP0/1				
MOVAPS/D	r,m	2	6	2	AGU				
MOVAPS/D	m,r	2	6	3	FP1				
MOVUPS/D	r,r	2	1	1	FP0/1				
MOVUPS/D	r,m	2	6-9	2-6	AGU				
MOVUPS/D	m,r	2	6-9	3-6	FP1				
MOVSS/D	r,r	1	1	0.5	FP0/1				
MOVSS/D	r,m	2	6	2	FP1				
MOVSS/D	m,r	1	5	2	FP1				
MOVHLPS, MOVLHPS	,.	'		_					
INIOVITEI O, INIOVEITI O	r,r	1	1	0.5	FP0/1				
MOVHPS/D, MOVLPS/	• ,•	'		0.0	110/1				
D NOVINGOLI G	r,m	1	6	2	AGU				
MOVHPS/D, MOVLPS/	,,,,,			_					
D	m,r	1	5	3	FP1				
MOVNTPS/D	m,r	2	12	3	FP1				
MOVNTSS/D	m,r	1	12	2	FP1	SSE4.A, AMD only			
MOVDDUP	r,r	2	2	_ 1	FP0/1	SSE3			
MOVDDUP	r,m64	2	7	2	FP0/1	SSE3			
MOVSHDUP,	1,1110-1	_	,	_	11 0/1	0020			
MOVSLDUP	r,r	2	1	1	FP0/1				
MOVSHDUP,	• ,•	-		•	110/1				
MOVSLDUP	r,m	2	12	3	AGU				
MOVMSKPS/D	r32,r	1	~6	2	FP0				
SHUFPS/D	r,r/m,i	3	2	2	FP0/1				
UNPCK H/L PS/D	r,r/m	2	1	1	FP0/1				
ON ONTIFE OF	1,1/111		'	'	11 0/1				
Conversion									
CVTPS2PD	r,r/m	2	5	2	FP1				
CVTPD2PS	r,r/m	4	5	3	FP0, FP1				
CVTSD2SS	r,r/m	3	5	3	FP0, FP1				
CVTSS2SD	r,r/m	1	4	1	FP1				
CVTDQ2PS	r,r/m	2	4	4	FP1				
CVTDQ2PD	r,r/m	2	5	2	FP1				
CVT(T)PS2DQ	r,r/m	2	4	4	FP1				
CVT(T)PD2DQ	r,r/m	4	6	3	FP0, FP1				
CVTPI2PS	xmm,mm	1	4	2	FP1				
CVTPI2PD	xmm,mm	2	5	2	FP1				
CVT(T)PS2PI	mm,xmm	1	4	1	FP1				
CVT(T)PD2PI	mm,xmm	3	6	2	FP0, FP1				
CVTSI2SS	xmm,r32		12	3	•				
		3			FP0, FP1				
CVTSI2SD	xmm,r32	2	11	3	FP1				
CVT(T)SS2SI	r32,xmm	2	12	1	FP0, FP1				
CVT(T)SD2SI	r32,xmm	2	11	1	FP0, FP1				
Arithmetic									
ADDSS/D SUBSS/D	r,r/m	1	3	1	FP0				

A D D D O / D O / D D O / D			۰ .		FD0	I
ADDPS/D SUBPS/D	r,r/m	2 2	3	2 2	FP0	0050
ADDSUBPS/D	r,r/m	2	3	2	FP0	SSE3
HADDPS/D HSUBPS/	r r/m	2	3	2	FP0	SSE3
D MULSS	r,r/m		2	1	FP1	SSES
	r,r/m	1			FP1	
MULSD	r,r/m	1	4	2		
MULPS	r,r/m	2	2	2	FP1	
MULPD	r,r/m	2	4	4	FP1	
DIVSS	r,r/m	1	13	13	FP1	
DIVPS	r,r/m	2	38	38	FP1	
DIVSD	r,r/m	1	17	17	FP1	
DIVPD	r,r/m	2	34	34	FP1	
RCPSS	r,r/m	1	3	1	FP1	
RCPPS	r,r/m	2	3	2	FP1	
MAXSS/D MINSS/D	r,r/m	1	2	1	FP0	
MAXPS/D MINPS/D	r,r/m	2	2	2	FP0	
CMPccSS/D	r,r/m	1	2	1	FP0	
CMPccPS/D	r,r/m	2	2	2	FP0	
COMISS/D UCOMISS/						
D	r,r/m	1		1	FP0	
Logic						
ANDPS/D ANDNPS/D	,				ED0/4	
ORPS/D XORPS/D	r,r/m	2	1	1	FP0/1	
Math						
SQRTSS	r,r/m	1	14	14	FP1	
SQRTPS	r,r/m	2	48	48	FP1	
SQRTSD	r,r/m	1	24	24	FP1	
SQRTPD	r,r/m	2	48	48	FP1	
RSQRTSS	r,r/m	1	3	1	FP1	
		2	3	2	FP1	
RSQRTPS	r,r/m	2	၂ ၁	2	FPI	
Other						
LDMXCSR	m	12		10	FP0, FP1	
STMXCSR	m	3		11	FP0, FP1	

AMD Jaguar

List of instruction timings and macro-operation breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB,

JNE, etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, xmm = 128 bit xmm register, m = any memory operand including

indirect operands, m64 means 64-bit memory operand, etc.

Ops: Number of micro-operations issued from instruction decoder to schedulers. In-

structions with more than 2 micro-operations are micro-coded.

Latency: This is the delay that the instruction generates in a dependency chain. The num-

bers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latencies listed do not include memory operands where the oper-

and is listed as register or memory (r/m).

The clock frequency varies dynamically, which makes it difficult to measure latencies. The values listed are measured after the execution of millions of similar instructions, assuming that this will make the processor boost the clock frequency

to the highest possible value.

Reciprocal through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/2 indicates that the execution units can handle 2 instructions per clock cycle in one thread. How-

ever, the throughput may be limited by other bottlenecks in the pipeline.

Execution pipe: Indicates which execution pipe is used for the micro-operations. 10 means integer

pipe 0. I0/1 means integer pipe 0 or 1. FP0 means floating point pipe 0 (ADD). FP1 means floating point pipe 1 (MUL). FP0/1 means either one of the two floating point pipes. Two micro-operations can execute simultaneously if they go to

different execution pipes.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes
Move instructions						
MOV	r,r	1	1	0.5	10/1	
MOV	r,i	1		0.5	10/1	
MOV	r8/16,m	1	4	1	AGU	Any addressing mode Any addressing
MOV	m,r8/16	1	4	1	AGU	mode
моу	r32/64,m	1	3	1	AGU	Any addressing mode Any addressing
MOV	m,r32/64	1	0	1	AGU	mode
MOV	m,i	1		1	AGU	
MOVNTI	m,r	1	6	1	AGU	
MOVZX, MOVSX	r,r	1	1	0.5	10/1	
MOVZX, MOVSX	r,m	1	4	1		
MOVSXD	r64,r32	1	1	0.5		
MOVSXD	r64,m32	1	3	1		
CMOVcc	r,r	1	1	0.5	10/1	
CMOVcc	r,m	1		1		

			ouge			
XCHG	r8,r8	3	2	2	10/1	
XCHG	r,r	2	1	1	10/1	
						Timing depends on
XCHG	r,m	3	16			hw
XLAT		2	5	3		
PUSH	r	1		1		
PUSH	i	1		1		
PUSH	m	2		1		
PUSH	SP	2		1		
PUSHF(D/Q)		9		6		
PUSHA(D)		9		8		
POP	r	1		1		
POP	m	3		2		
POP	SP	1		2		
POPF(D/Q)		29		18		
POPA(D)		9		8		
LEA	r16,[m]	2	3	2	10	Any address size
LEA	r32/64,[m]	1	1	0.5	10/1	1-2 comp., no scale
LEA	r32/64,[m]	1	2	1	10	3 comp. or scale
LEA	r64,[m]	1	_	0.5	10/1	RIP relative
LAHF	101,[]	4	3	2	10/ 1	Tui Tolauvo
SAHF		1	1	0.5	10/1	
SALC		1	1	1	10/1	
BSWAP	r	1	1	0.5	10/1	
MOVBE	r,m	1		1	10/1	MOVBE
MOVBE	m,r	1		1		MOVBE
PREFETCHNTA	m	1		~100	AGU	IVIOVEL
PREFETCHT0/1/2		1		~100	AGU	
PREFETCHW	m m	1		~100	AGU	
LFENCE	m	1		0.5	AGU	
MFENCE		4		~45	AGU	
SFENCE		4		~45	AGU	
SPENCE		4		~45	AGU	
Arithmetic instruction	S					
ADD, SUB	r,r/i	1	1	0.5	10/1	
ADD, SUB	r,m	1		1		
ADD, SUB	m,r	1	6	1		
ADC, SBB	r,r/i	1	1	1	10/1	
ADC, SBB	r,m	1		1		
ADC, SBB	m,r/i	1	8			
CMP	r,r/i	1	1	0.5	10/1	
CMP	r,m	1		1		
INC, DEC, NEG	r	1	1	0.5	10/1	
INC, DEC, NEG	m	1	6	1		
AAA		9	5			
AAS		9	8			
DAA		12	6			
DAS		16	8			
AAD		4	5			
AAM		8	14	13		
MUL, IMUL	r8/m8	1	3	1	10	
MUL, IMUL	r16/m16	3	3	3	10	
MUL, IMUL	r32/m32	2	3	2	10	
		_	1	-		

la av 11 - 12 av 11		_	۱ ۵	ı –	1 10	
MUL, IMUL	r64/m64	2	6	5	10	
IMUL	r16,r16/m16	1	3	1	10	
IMUL	r32,r32/m32	1	3	1	10	
IMUL	r64,r64/m64	1	6	4	10	
IMUL	r16,(r16),i	2	4	1	10	
IMUL	r32,(r32),i	1	3	1	10	
IMUL	r64,(r64),i	1	6	4	10	
DIV	r8/m8	1	11-14	11-14	10	
DIV	r16/m16	2	12-19	12-19	10	
DIV	r32/m32	2	12-27	12-27	10	
DIV	r64/m64	2	12-43	12-43	10	
IDIV	r8/m8	1	11-14	11-14	10	
1		2				
IDIV	r16/m16		12-19	12-19	10	
IDIV	r32/m32	2	12-27	12-27	10	
IDIV	r64/m64	2	12-43	12-43	10	
CBW, CWDE, CDQE		1	1		10/1	
CWD, CDQ, CQO		1	1		10/1	
Logic instructions						
AND, OR, XOR	r,i	1	1	0.5	10/1	
AND, OR, XOR	r,r	1	1	0.5	I0/1	
AND, OR, XOR	r,m	1		1		
AND, OR, XOR	m,r	1	6	1		
ANDN	r,r,r	1	1	0.5		BMI1
ANDN	r,r,m	2		1		BMI1
TEST	r,i	1	1	0.5	10/1	DIVILI
TEST		1	1	0.5	10/1	
1	r,r		l 		10/1	
TEST	r,m	1	_	1	10/4	
NOT	r	1	1	0.5	IO/1	
NOT	m	1	6	1		
SHL, SHR, SAR	r,i/CL	1	1	0.5	10/1	
ROL, ROR	r,i/CL	1	1	0.5	10/1	
RCL, RCR	r,1	1	1	1	10/1	
RCL	r,i	9	5	5		
RCR	r,i	7	4	4		
RCL	r,CL	9	5	5		
RCR	r,CL	7	4	4		
SHL,SHR,SAR,ROL,						
ROR	m,i /CL	1	6	1		
RCL, RCR	m,1	1		1		
RCL	m,i	10		11		
RCR	m,i	9		11		
RCL	m,CL	9		11		
RCR	m,CL	8		11		
SHLD, SHRD	r,r,i	6	3	3		
		7	4	4		
SHLD, SHRD	r,r,cl		4	11		
SHLD, SHRD	m,r,i/CL	8				
BT	r,r/i	1		0.5		
BT	m,i	1		1		
BT	m,r	5	_	3		
BTC, BTR, BTS	r,r/i	2	2	1		
BTC	m,i	5		11		
BTR, BTS	m,i	4		11		
						·

BTC, BTR, BTS BSF BSR BSF, BSR POPCNT LZCNT TZCNT BLSI BLSR BLSI BLSR BLSMSK BLSMSK BEXTR BEXTR BEXTR SETcc CLC, STC CMC CLD STD	m,r	8 7 8 8 1 1 2 3 2 3 1 1 1 1 1 1 1 1 1	4 4 1 1 2 2 2 1 1	11 4 4 4 0.5 0.5 1 2 0.5 1 0.5 1 0.5 1	10/1 10/1 10 10,11	SSE4A/SSE4.2 SSE4A/LZCNT BMI1 BMI1 BMI1 BMI1 BMI1 BMI1 BMI1
Control transfer instru	etions					
Control transfer instru JMP JMP JMP Jcc J(E/R)CXZ LOOP LOOPE LOOPNE CALL CALL CALL RET RET BOUND INTO	short/near r m(near) short/near short short short r m(near) i m	1 1 1 2 8 10 2 2 5 1 4 8		2 2 2 0.5 - 2 1 - 2 5 6 2 2 2 3 3 4		2 if jumping 2 if jumping values are for no jump values are for no jump
String instructions LODS REP LODS STOS REP STOS REP STOS MOVS REP MOVS REP MOVS SCAS REP SCAS CMPS REP CMPS Synchronization		4 ~5n 4 ~2n 2/16B 7 ~2n 2/16B 5 ~6n 7 ~6n		2 ~3n 2 ~n 1/16B 4 ~1.5n 1/16B 3 ~3n 4 ~3n		for small n best case for small n best case

LOCK ADD	m,r	1 1	19			
XADD	m,r	4	11			
LOCK XADD	m,r	4	16			
CMPXCHG	m,r8	5	11			
LOCK CMPXCHG	m,r8	5	16			
CMPXCHG	m,r16/32/64	6	11			
LOCK CMPXCHG	m,r16/32/64	6	17			
CMPXCHG8B	m64	18	11			
LOCK CMPXCHG8B	m64	18	19			
CMPXCHG16B	m128	28	32			
LOCK CMPXCHG16B	m128	28	38			
Other						
NOP (90)		1		0.5	10/1	
Long NOP (0F 1F)		1		0.5	10/1	
PAUSE		37		46		
ENTER		i,0	12		18	
ENTER		a,b	10+6b	17+3b		
LEAVE		2		3		32 bit mode
CPUID		30-59	70-230			
XGETBV		5		5		
RDTSC		34		41		
RDTSCP		34		42		rdtscp
RDPMC		30		27		
CRC32	r,r	3	3	2		
CRC32	r,m	4		2		

Floating point x87 instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
				throughput	pipe	
Move instructions						
FLD	r	1	2	0.5	FP0/1	
FLD	m32/64	1	4	1	FP0/1	
FLD	m80	7	9	5		
FBLD	m80	21	24	29		
FST(P)	r	1	2	0.5	FP0/1	
FST(P)	m32/64	1	3	1	FP1	
FSTP	m80	10	9	7		
FBSTP	m80	217	167	168		
FXCH	r	1	0	1	FP1	
FILD	m	1	8	1	FP1	
FIST(T)(P)	m	1	4	1	FP1	
FLDZ, FLD1		1		1	FP1	
FCMOVcc	st0,r	12	7	7	FP0/1	
FFREE	r	1		1	FP1	
FINCSTP, FDECSTP		1	1	1	FP1	
FNSTSW	AX	2		11	FP1	
FNSTSW	m16	2		11	FP1	
FNSTCW	m16	3		2	FP0	
FLDCW	m16	12		9	FP1	
Arithmetic instruction	S					

	ı		_		ı	1
FADD(P),FSUB(R)(P)	r	1	3	1	FP0	
FADD(P),FSUB(R)(P)	m	1		1	FP0	
FIADD,FISUB(R)	m	2		2	FP0,FP1	
FMUL(P)	r	1	5	3	FP1	
FMUL(P)	m	1		3	FP1	
FIMUL	m	1			FP1	
FDIV(R)(P)	r	1	22	22	FP1	
FDIV(R)(P)	m	1		22	FP1	
FIDIV(R)	m	2		22	FP1	
FABS, FCHS		1	2	2	FP1	
FCOM(P), FUCOM(P)	r	1		1	FP0	
FCOM(P), FUCOM(P)	m	1		1	FP0	
FCOMPP, FUCOMPP		1		1	FP0	
FCOMI(P)	r	1		2	FP0	
FICOM(P)	m	2		_ 1	FP0, FP1	
FTST		1		1	FP0	
FXAM		2		2	1FP1	
FRNDINT		5	8	4	FP0, FP1	
FPREM		1	11-54	·	FP1	
FPREM1		1	11-56		FP1	
I I IXEIVII		'	11-00			
Math						
FSQRT		1	35	35	FP1	
FLDPI, etc.		1		1	FP0	
FSIN		4-44	30-139	30-151	FP0, FP1	
FCOS		11-51	38-93		FP0, FP1	
FSINCOS		11-76	55-122	55-180	FP0, FP1	
FPTAN		11-45	55-177	55-177	FP0, FP1	
FPATAN		9-75	44-167	44-167	FP0, FP1	
FSCALE		5	27		FP0, FP1	
FXTRACT		7	9	6	FP0, FP1	
F2XM1		8	32-37		FP0, FP1	
FYL2X		8-51	30-120	30-120	FP0, FP1	
FYL2XP1		61	~160	~160	FP0, FP1	
Other						
FNOP		1		0.5	FP0/1	
(F)WAIT		1	0	0.5	ALU	
FNCLEX		9		32	FP0, FP1	
FNINIT		27		78	FP0, FP1	
FNSAVE	m	88	138-150	138-150	FP0, FP1	
FRSTOR	m	80	136	136	FP0, FP1	

Integer vector instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes
Move instructions						
MOVD	r32, mm	1	4	1	FP0	
MOVD	mm, r32	2	6	1	FP0/1	
MOVD	mm,m32	1	4	1	AGU	
MOVD	r32, x	1	4	1	FP0	
MOVD	x, r32	2	6	1	FP1	

MOVD	x,m32	1	4	1	AGU	
MOVD	m32,mm/x	1	3	1 1	FP1	
INIOVD	11132,11111/X	ı .	3	ļ	FFI	Moves 64 bits.Name
MOVD / MOVQ	r64,mm/x	1	4	1	FP0	of instruction differs
MOVQ	mm,r64	2	6	1	FP0/1	do.
MOVQ	x,r64	2	6	1	FP0/1	do.
MOVQ	· '	1	1	-	FP0/1 FP0/1	do.
	mm,mm		1	0.5		
MOVQ	X,X	1	1	0.5	FP0/1	
MOVQ	mm/x,m64	1	4	1	AGU	
MOVQ	m64,mm/x	1	3	1	FP1	
MOVDQA	X,X	1	1	0.5	FP0/1	
VMOVDQA	y,y	2	1	1	FP0/1	AVX
MOVDQA	x,m	1	4	1	AGU	
VMOVDQA	y,m	2	4	2	AGU	AVX
MOVDQA	m,x	1	3	1	FP1	
VMOVDQA	m,y	2	3	2	FP1	AVX
MOVDQU, LDDQU	x.m	1	4	1	AGU	
MOVDQU	m,x	1	3	1	FP1	
MOVDQ2Q	mm,x	1	1	0.5	FP0/1	
MOVQ2DQ	x,mm	1	1 1	0.5	FP0/1	
MOVNTQ	m,mm	1	429	2	FP1	
MOVNTDQ	m,x	1	429	2	FP1	
PACKSSWB/DW	,			_		
PACKUSWB	mm,r/m	1	1 1	0.5	FP0/1	
PACKSSWB/DW	,					
PACKUSWB	x,r/m	1	2	0.5	FP0/1	
PUNPCKH/LBW/WD/	,					
DQ	mm,r/m	1	1	0.5	FP0/1	
PUNPCKH/LBW/WD/						
DQ	x,r/m	1	2	0.5	FP0/1	
PUNPCKH/LQDQ	x,r/m	1	2	0.5	FP0/1	
PSHUFB	mm,mm	1	1	0.5	FP0/1	Suppl. SSE3
PSHUFB	x,x	3	4	2	FP0/1	Suppl. SSE3
PSHUFD	x,x,i	1	2	0.5	FP0/1	
PSHUFW	mm,mm,i	1	1 1	0.5	FP0/1	
PSHUFL/HW	x,x,i	1	1	0.5	FP0/1	
PALIGNR	x,x,i	1	2	0.5	FP0/1	Suppl. SSE3
PBLENDW	x,r/m	1	1	0.5	FP0/1	SSE4.1
MASKMOVQ	mm,mm	32	432	17	FP0, FP1	0021.1
MASKMOVDQU	x,x	64	43-2210	34	FP0, FP1	
PMOVMSKB	r32,mm/x	1	3	1	FP0	
PEXTRW	r32,mm/x,i	1	4	1	FP0	
PINSRW						
	mm,r32,i	2 2	8 7	1	FP0/1	
PINSRB/W/D/Q	x,r,i		'	1	FP0/1	
PINSRB/W/D/Q	x,m,i	1		1	FP0/1	00544
PEXTRB/W/D/Q	r,x,i	1	3	1	FP0	SSE4.1
PEXTRB/W/D/Q	m,x,i	1		1	FP1	SSE4.1
INSERTQ	X,X	3	2	2	FP0, FP1	SSE4A, AMD only
INSERTQ	x,x,i,i	3	2	2	FP0, FP1	SSE4A, AMD only
EXTRQ	X,X	1	1	0.5	FP0/1	SSE4A, AMD only
EXTRQ	x,x,i,i	1	1	0.5	FP0/1	SSE4A, AMD only
PMOVSXBW/BD/BQ/				_		
WD/WQ/DQ	x,x	1	2	0.5	FP0/1	SSE4.1

DMOV/ZVDM/DD/DO/	1	1	ı	I	I	I I
PMOVZXBW/BD/BQ/ WD/WQ/DQ	x,x	1	2	0.5	FP0/1	SSE4.1
VVD/VVQ/DQ	^,^	'		0.5	1 1 0/1	33L4.1
Arithmetic instruction	 e					
PADDB/W/D/Q	5					
PADDSB/W ADDUSB/						
W PSUBB/W/D/Q						
PSUBSB/W						
PSUBUSB/W	mm/x,r/m	1	1	0.5	FP0/1	
PHADD/SUBW/SW/D	mm,r/m	1	1	0.5	FP0/1	Suppl. SSE3
PHADD/SUBW/SW/D	x,r/m	1	2	0.5	FP0/1	Suppl. SSE3
PCMPEQ/GT B/W/D	mm,r/m	1	1	0.5	FP0/1	
PCMPEQ/GT B/W/D	x,r/m	1	1	0.5	FP0/1	
PCMPEQQ	mm/x,r/m	1	1	0.5	FP0/1	SSE4.1
PCMPGTQ	mm/x,r/m	1	1	0.5	FP0/1	SSE4.2
· ·		'	'	0.5	FFU/I	3324.2
PMULLW PMULHW PMULHUW						
PMULUDQ	mm/x,r/m	1	2	1	FP0	
PMULLD	x,r/m	3	4	2	FP0 FP1	SSE4.1
PMULDQ	x,r/m	1	2	1	FP0	SSE4.1
PMULHRSW	mm/x,r/m	1	2	1	FP0	Suppl. SSE3
PMADDWD	mm/x,r/m	1	2	1	FP0	- Suppi. 33E3
			2		FP0	Cuppl CCE2
PMADDUBSW	mm/x,r/m	1		•		Suppl. SSE3
PAVGB/W	mm/x,r/m	1	1	0.5	FP0/1	
PMIN/MAX SW/UB	mm/x,r/m	1	1	0.5	FP0/1	0 1 0050
PABSB/W/D	mm/x,r/m	1	1	0.5	FP0/1	Suppl. SSE3
PSIGNB/W/D	mm/x,r/m	1	1	0.5	FP0/1	Suppl. SSE3
PSADBW	mm/x,r/m	1	2	0.5	FP0/1	
MPSADBW	x,x,i	3	4	1	FP0/1	SSE4.1
Logic						
PAND PANDN POR						
PXOR	mm/x,r/m	1	1	0.5	FP0/1	
PSLL/RL W/D/Q	11111/73,17111			0.0	110/1	
PSRAW/D	mm,i/mm/m	1	1	0.5	FP0/1	
PSLL/RL W/D/Q	,,,,,		•	0.0		
PSRAW/D	x,x	1	2	0.5	FP0/1	
PSLL/RL W/D/Q	,		_			
PSRAW/D	x,i	1	1	0.5	FP0/1	
PSLLDQ, PSRLDQ	x,i	1	2	0.5	FP0/1	
PTEST	x,x/m	1	3	1	FP0	SSE4.1
String instructions	_		_	_		
PCMPESTRI	x,x,i	9	5	5	FP0/1	SSE4.2
PCMPESTRI	x,m,i	10		5	FP0/1	SSE4.2
PCMPESTRM	x,x,i	9	9	9	FP0/1	SSE4.2
PCMPESTRM	x,m,i	10		9	FP0/1	SSE4.2
PCMPISTRI	x,x,i	3	2	2	FP0/1	SSE4.2
PCMPISTRI	x,m,i	4		2	FP0/1	SSE4.2
PCMPISTRM	x,x,i	3	8	8	FP0/1	SSE4.2
PCMPISTRM	x,m,i	4		2	FP0/1	SSE4.2
Encryption						
Encryption PCLMULQDQ	v v/m i	1	່ ,	4	EDO	DCI MI II
	x,x/m,i	1	3 5	1	FP0	PCLMUL
AESDEC	X,X	2	၂ ၁	1	FP0/1	AES

AESDECLAST	X,X	2	5	1	FP0/1	AES	
AESENC	X,X	2	5	1	FP0/1	AES	
AESENCLAST	X,X	2	5	1	FP0/1	AES	
AESIMC	x,x	1	2	1	FP0/1	AES	
AESKEYGENASSIST	x,x,i	1	2	1	FP0/1	AES	
Other							
EMMS		1		0.5	FP0/1		

Floating point XMN		ns				
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes
Move instructions						
MOVAPS/D	X,X	1	1	0.5	FP0/1	
VMOVAPS/D	y,y	2	1	1	FP0/1	
MOVAPS/D	x,m	1	4	1	AGU	
VMOVAPS/D	y,m	2	4	2	AGU	
MOVAPS/D	m,x	1	3	1	FP1	
VMOVAPS/D	m,y	2	3	2	FP1	
MOVUPS/D	X,X	1	1	0.5	FP0/1	
VMOVUPS/D	y,y	2	1	1	FP0/1	
MOVUPS/D	x,m	1	4	1	AGU	
VMOVUPS/D	y,m	2	4	2	AGU	
MOVUPS/D	m,x	1	3	1	FP1	
VMOVUPS/D	m,y	2	3	2	FP1	
MOVSS/D	X,X	1	1	0.5	FP0/1	
MOVSS/D	x,m	1	4	1	AGU	
MOVSS/D	m,x	1	3	1	FP1	
MOVHLPS, MOVLHPS						
	X,X	1	2	2	FP0/1	
MOVHPS/D, MOVLPS/						
D	x,m	1	5	1	FP0/1	
MOVHPS/D, MOVLPS/				_	ED.4	
D MOVALTBOAR	m,x	1	4	1	FP1	
MOVNTPS/D	m,x	1	429	1	FP1	00544 445
MOVNTSS/D	m,x	1		1	FP1	SSE4A, AMD only
MOVDDUP	X,X	1	2	0.5	FP0/1	SSE3
MOVDDUP	x,m64	1		1	AGU	SSE3
VMOVDDUP	y,y	2	2	1	FP0/1	AVX
VMOVDDUP	y,m	2		2	AGU	AVX
MOVSH/LDUP	X,X	1	1	0.5	FP0/1	
MOVSH/LDUP	x,m	1		1	AGU	
VMOVSH/LDUP	y,y	2	1	1	FP0/1	AVX
VMOVSH/LDUP	y,m	2	_	2	AGU	AVX
MOVMSKPS/D	r32,x	1	3	1	FP0	
VMOVMSKPS/D	r32,y	1	3	1	FP0	AVX
SHUFPS/D	x,x/m,i	1	2	0.5	FP0/1	
VSHUFPS/D	y,y,y,i	2	2	1	FP0/1	AVX
UNPCK H/L PS/D	x,x/m	1	2	0.5	FP0/1	
VUNPCK H/L PS/D	y,y,y	2	2	1	FP0/1	AVX
EXTRACTPS	r32,x,i	1	3	1	FP0	
EXTRACTPS	m32,x,i	1	3	1	FP1	

			J			
VEXTRACTF128	x,y,i	1	1	0.5	FP0/1	AVX
VEXTRACTF128	m128,y,i	1	12	1	FP1	AVX
INSERTPS	x,x,i	1		1	FP0/1	
INSERTPS	x,m32,i	1	6	1	FP0/1	
VINSERTF128	y,y,x,i	2	1	1	FP0/1	AVX
VINSERTF128	y,y,m128,i	2	13	2	FP0/1	AVX
VMASKMOVPS/D	x,x,m128	1	15	1	FP0/1	>300 clk if mask=0
VMASKMOVPS/D	y,y,m256	2	15	2	FP0/1	>300 clk if mask=0
VMASKMOVPS/D	m128,x,x	19	21	16	FP1	AVX
VMASKMOVPS/D	m256,y,y	36	32	22	FP1	AVX
Conversion						
CVTPS2PD	x,x/m	1	3	1	FP1	
VCVTPS2PD	y,x/m	2	4	2	FP1	
CVTPD2PS	x,x/m	1	4	1	FP1	
VCVTPD2PS	x,y	3	6	2	FP0, FP1	
CVTSD2SS	x,x/m	2	5	8	FP1	
CVTSS2SD	x,x/m	2	4	7	FP1	
CVTDQ2PS/PD	x,x/m	1	4	1	FP1	
VCVTDQ2PS/PD	y,y	2	4	2	FP1	
CVT(T)PS2DQ	x,x/m	1	4	1	FP1	
VCVT(T)PS2DQ	y,y	2	4	2	FP1	
CVT(T)PD2DQ	x,x/m	1	4	1	FP1	
VCVT(T)PD2DQ	y,y	3	7	2	FP1	
CVTPI2PS	xmm,mm	1	4	1	FP1	
CVTPI2PD	xmm,mm	1	4	1	FP1	
CVT(T)PS2PI	mm,xmm	1	4	1	FP1	
CVT(T)PD2PI	mm,xmm	1	4	1	FP1	
CVTSI2SS	xmm,r32	2	9	1	FP1	
CVTSI2SD	xmm,r32	2	9	1	FP1	
CVT(T)SS2SI	r32,xmm	2	8	1	FP1	
CVT(T)SD2SI	r32,xmm	2	8	1	FP1	
VCVTPS2PH	x/m,x,i	1	4	1	FP1	F16C
VCVTPS2PH	x/m,y,i	3	6	2	FP0, FP1	F16C
VCVTPH2PS	x,x/m	1	4	1	FP1	F16C
VCVTPH2PS	y,x/m	2	5	2	FP1	F16C
Arithmetic						
ADDSS/D SUBSS/D	x,x/m	1	3	1	FP0	
ADDPS/D SUBPS/D	x,x/m	1	3	1	FP0	
VADDPS/D VSUBPS/D	y,y/m	2	3	2	FP0	
ADDSUBPS/D	x,x/m	1	3	1	FP0	SSE3
VADDSUBPS/D	y,y/m	2	3	2	FP0	
HADD/SUBPS/D	x,x/m	1	4	1	FP0	SSE3
VHADD/SUBPS/D	y,y/m	2	4	2	FP0	
MULSS/PS	x,x/m	1	2	1	FP1	
VMULPS	y,y/m	2	2	2	FP1	
MULSD/PD	x,x/m	1	4	2	FP1	
VMULPD	y,y/m	2	4	2	FP1	
DIVSS	x,x/m	1	14	14	FP1	
DIVPS	x,x/m	1	19	19	FP1	
VDIVPS	y,y/m	2	38	38	FP1	
DIVSD	x,x/m	1	19	19	FP1	
	1 2324111		0			I I

I		I .		l		
DIVPD	x,x/m	1	19	19	FP1	
VDIVPD	y,y/m	2	38	38	FP1	
RCPSS	x,x/m	1	2	1	FP1	
RCPPS	x,x/m	1	2	1	FP1	
VRCPPS	y,y/m	2	2	2	FP1	
MAXSS/D MINSS/D	x,x/m	1	2	1	FP0	
MAXPS/D MINPS/D	x,x/m	1	2	1	FP0	
VMAXPS/D VMINPS/D	y,y/m	2	2	2	FP0	
CMPccSS/D	x,x/m	1	2	1	FP0	
CMPccPS/D	x,x/m	1	2	1	FP0	
VCMPccPS/D		2	2	2	FP0	
	y,y/m					
(U)COMISS/D	x,x/m	1		1	FP0	
ROUNDSS/SD/PS/PD	x,x/m,i	1	4	1	FP1	
VROUNDSS/D/PS/D	y,y/m,i	2	4	2	FP1	
DPPS	x,x,i	5	11	4	FP0, FP1	SSE4.1
DPPS	x,m,i	6		4	FP0, FP1	SSE4.1
VDPPS	y,y,y,i	10	12	7	FP0, FP1	SSE4.1
VDPPS	y,m,i	12		7	FP0, FP1	SSE4.1
DPPD	x,x,i	3	9	3	FP0, FP1	SSE4.1
DPPD	x,m,i	4		3	FP0, FP1	SSE4.1
Logic						
ANDPS/D ANDNPS/D						
ORPS/D XORPS/D	x,x/m	1	1	0.5	FP0/1	
VANDPS/D, etc.	y,y/m	2	1	1	FP0/1	
Math	,					
SQRTSS	x,x/m	1	16	16	FP1	
SQRTPS	x,x/m	2	21	21	FP1	
VSQRTPS	y,y/m	2	42	42	FP1	
SQRTSD	x,x/m	1	27	27	FP1	
SQRTPD	x,x/m	2	27	27	FP1	
VSQRTPD	y,y/m	2	54	54	FP1	
RSQRTSS/PS	x,x/m	1	2	1	FP1	
VRSQRTPS	y,y/m	2	2	2	FP1	
Other						
LDMXCSR	m	12	9	8	FP0, FP1	
STMXCSR	m	3	13	12	FP0, FP1	
VZEROUPPER		21		30		32 bit mode
VZEROUPPER		37		46		64 bit mode
VZEROALL		41		58		32 bit mode
VZEROALL		73		90		64 bit mode
FXSAVE		66	66	66		32 bit mode
FXSAVE		58	58	58		64 bit mode
FXRSTOR		115	189	189		32 bit mode
FXRSTOR		123	198	197		64 bit mode
XSAVE		130	145	145		32 bit mode
XSAVE		114	129	129		64 bit mode
XRSTOR		219	342	342		32 bit mode
XRSTOR		251	375	375		64 bit mode
7.1.C 1 O 1 C		201	0,0	0,0		U-F DIL HIDGE

Intel Pentium and Pentium MMX

List of instruction timings

Explanation of column headings:

Operands r = register, accum = al, ax or eax, m = memory, i = immediate data, sr =

segment register, m32 = 32 bit memory operand, etc.

Clock cycles The numbers are minimum values. Cache misses, misalignment, and

exceptions may increase the clock counts considerably.

Pairability u = pairable in u-pipe, v = pairable in v-pipe, uv = pairable in either pipe,

np = not pairable.

Integer instructions (Pentium and Pentium MMX)

Integer instructions (Pen	1		D - 1 - 1 - 114 - 1
Instruction	Operands	Clock cycles	
NOP		1	uv
MOV	r/m, r/m/i	1	uv
MOV	r/m, sr	1	np
MOV	sr , r/m	>= 2 b)	np
MOV	m , accum	1	uv h)
XCHG	(E)AX, r	2	np
XCHG	r,r	3	np
XCHG	r, m	>15	np
XLAT		4	np
PUSH	r/i	1	uv
POP	r	1	uv
PUSH	m	2	np
POP	m	3	np
PUSH	sr	1 b)	np
POP	sr	>= 3 b)	np
PUSHF		3-5	np
POPF		4-6	np
PUSHA POPA		5-9 i)	np
PUSHAD POPAD		5	np
LAHF SAHF		2	np
MOVSX MOVZX	r , r/m	3 a)	np
LEA	r, m	1	uv
LDS LES LFS LGS LSS	m	4 c)	np
ADD SUB AND OR XOR	r , r/i	1	uv
ADD SUB AND OR XOR	r, m	2	uv
ADD SUB AND OR XOR	m , r/i	3	uv
ADC SBB	r , r/i	1	u
ADC SBB	r, m	2	u
ADC SBB	m , r/i	3	u
CMP	r , r/i	1	uv
CMP	m , r/i	2	uv
TEST	r,r	1	uv
TEST	m, r	2	uv
TEST	r , i	1	f)
TEST	m, i	2	np
INC DEC	r	1	uv
INC DEC	m	3	uv
NEG NOT	r/m	1/3	np
MUL IMUL	r8/r16/m8/m16	11	np

Intel Pentium

MUL IMUL	all other versions	9 d)	np
DIV	r8/m8	17	np
DIV	r16/m16	25	np
DIV	r32/m32	41	np
IDIV	r8/m8	22	np
IDIV	r16/m16	30	np
IDIV	r32/m32	46	np
CBW CWDE	102/11102	3	np
CWD CDQ		2	np
SHR SHL SAR SAL	r,i	1	u
SHR SHL SAR SAL	m,i	3	u
SHR SHL SAR SAL	r/m, CL	4/5	np
ROR ROL RCR RCL	r/m, 1	1/3	u
ROR ROL	r/m, i(><1)	1/3	np
ROR ROL	r/m, CL	4/5	np
RCR RCL	r/m, i(><1)	8/10	np
RCR RCL	r/m, CL	7/9	np
SHLD SHRD	r, i/CL	4 a)	np
SHLD SHRD	m, i/CL	5 a)	np
BT	r, r/i	4 a)	np
BT	m, i	4 a)	
BT	m, i	9 a)	np np
BTR BTS BTC	r, r/i	7 a)	np np
BTR BTS BTC	m, i	8 a)	np
BTR BTS BTC		14 a)	np np
BSF BSR	m, r	7-73 a)	np
SETCC	r , r/m r/m	,	np
JMP CALL	short/near	1/2 a)	np
JMP CALL	far	1 e) >= 3 e)	V
	short/near	7-3e) 1/4/5/6e)	np V
conditional jump CALL JMP	r/m	2/5 e	
RETN	1/111	2/5 e 2/5 e	np
RETN	i	2/5 e 3/6 e)	np
RETF	l I	3/0 e) 4/7 e)	np
RETF	i	5/8 e)	np
J(E)CXZ		4-11 e)	np
LOOP	short short	5-10 e)	np
BOUND	r, m	,	np
CLC STC CMC CLD STD	1 , 111	8 2	np
CLI STI		6-9	np
LODS		2	np
REP LODS		7+3*n g)	np np
STOS		7+3 ii g) 3	np np
REP STOS		10+n g)	np
MOVS		4	np
REP MOVS		12+n g)	np np
SCAS		4	np
REP(N)E SCAS		9+4*n g)	np np
CMPS		٠,	·
		5 8+4*n g)	np
REP(N)E CMPS BSWAP			np
CPUID	r	1 a) 13-16 a)	np np
RDTSC			np np
וטנוטט		6-13 a) j)	np

Intel Pentium

Notes:

a This instruction has a 0FH prefix which takes one clock cycle extra to de-

code on a P1 unless preceded by a multi-cycle instruction. versions with FS and GS have a 0FH prefix. see note a.

b versions with FS and GS have a 0FH prefix, see note a.

c versions with SS, FS, and GS have a 0FH prefix. see note a.

d versions with two operands and no immediate have a 0FH prefix, see

note a.

e high values are for mispredicted jumps/branches.

f only pairable if register is AL, AX or EAX.

g add one clock cycle for decoding the repeat prefix unless preceded by a

multi-cycle instruction (such as CLD).

h pairs as if it were writing to the accumulator.
i 9 if SP divisible by 4 (imperfect pairing).

j on P1: 6 in privileged or real mode; 11 in non-privileged; error in virtual

mode. On PMMX: 8 and 13 clocks respectively.

Floating point instructions (Pentium and Pentium MMX)

Explanation of column headings

Operands r = register, m = memory, m32 = 32-bit memory operand, etc.

Clock cycles The numbers are minimum values. Cache misses, misalignment,

denormal operands, and exceptions may increase the clock counts

considerably.

Pairability + = pairable with FXCH, np = not pairable with FXCH.

i-ov Overlap with integer instructions. i-ov = 4 means that the last four clock

cycles can overlap with subsequent integer instructions.

fp-ov Overlap with floating point instructions. fp-ov = 2 means that the last two

clock cycles can overlap with subsequent floating point instructions.

(WAIT is considered a floating point instruction here)

Instruction	Operand	Clock cycles	Pairability	i-ov	fp-ov
FLD	r/m32/m64	1	0	0	0
FLD	m80	3	np	0	0
FBLD	m80	48-58	np	0	0
FST(P)	r	1	np	0	0
FST(P)	m32/m64	2 m)	np	0	0
FST(P)	m80	3 m)	np	0	0
FBSTP	m80	148-154	np	0	0
FILD	m	3	np	2	2
FIST(P)	m	6	np	0	0
FLDZ FLD1		2	np	0	0
FLDPI FLDL2E etc.		5 s)	np	2	2
FNSTSW	AX/m16	6 q)	np	0	0
FLDCW	m16	8	np	0	0
FNSTCW	m16	2	np	0	0
FADD(P)	r/m	3	0	2	2
FSUB(R)(P)	r/m	3	0	2	2
FMUL(P)	r/m	3	0	2	2 n)
FDIV(R)(P)	r/m	19/33/39 p)	0	38 o)	2
FCHS FABS		1	0	0	0
FCOM(P)(P) FUCOM	r/m	1	0	0	0
FIADD FISUB(R)	m	6	np	2	2
FIMUL	m	6	np	2	2

Intel Pentium

FIDIV(R)	m	22/36/42 p)	np	38 o)	2
FICOM	m	4	np	0	0
FTST		1	np	0	0
FXAM		17-21	np	4	0
FPREM		16-64	np	2	2
FPREM1		20-70	np	2	2
FRNDINT		9-20	np	0	0
FSCALE		20-32	np	5	0
FXTRACT		12-66	np	0	0
FSQRT		70	np	69 o)	2
FSIN FCOS		65-100 r)	np	2	2
FSINCOS		89-112 r)	np	2	2
F2XM1		53-59 r)	np	2	2
FYL2X		103 r)	np	2	2
FYL2XP1		105 r)	np	2	2
FPTAN		120-147 r)	np	36 o)	0
FPATAN		112-134 r)	np	2	2
FNOP		1	np	0	0
FXCH	r	1	np	0	0
FINCSTP FDECSTP		2	np	0	0
FFREE	r	2	np	0	0
FNCLEX		6-9	np	0	0
FNINIT		12-22	np	0	0
FNSAVE	m	124-300	np	0	0
FRSTOR	m	70-95	np	0	0
WAIT		1	np	0	0

Notes

r

m	The value to store is need	ed one clock cycle in advance.

n 1 if the overlapping instruction is also an FMUL.

Cannot overlap integer multiplication instructions.

p FDIV takes 19, 33, or 39 clock cycles for 24, 53, and 64 bit precision re-

spectively. FIDIV takes 3 clocks more. The precision is defined by bit 8-9

of the floating point control word.

q The first 4 clock cycles can overlap with preceding integer instructions.

Clock counts are typical. Trivial cases may be faster, extreme cases may

be slower.

s May be up to 3 clocks more when output needed for FST, FCHS, or

FABS.

MMX instructions (Pentium MMX)

A list of MMX instruction timings is not needed because they all take one clock cycle, except the MMX multiply instructions which take 3. MMX multiply instructions can be pipelined to yield a throughput of one multiplication per clock cycle.

The EMMS instruction takes only one clock cycle, but the first floating point instruction after an EMMS takes approximately 58 clocks extra, and the first MMX instruction after a floating point instruction takes approximately 38 clocks extra. There is no penalty for an MMX instruction after EMMS on the PMMX.

There is no penalty for using a memory operand in an MMX instruction because the MMX arithmetic unit is one step later in the pipeline than the load unit. But the penalty comes when you store data from an MMX register to memory or to a 32-bit register: The data have to be ready one clock cycle in advance. This is analogous to the floating point store instructions.

All MMX instructions except EMMS are pairable in either pipe. Pairing rules for MMX instructions are described in manual 3: "The microarchitecture of Intel, AMD and VIA CPUs".

Intel Pentium II and Pentium III

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, xmm = 128 bit xmm

register, sr = segment register, m = memory, m32 = 32-bit memory operand, etc.

μορs: The number of μops that the instruction generates for each execution port.

p0: Port 0: ALU, etc. p1: Port 1: ALU, jumps

p01: Instructions that can go to either port 0 or 1, whichever is vacant first.

p2: Port 2: load data, etc.

p3: Port 3: address generation for store

p4: Port 4: store data

Latency: This is the delay that the instruction generates in a dependency chain. (This is

not the same as the time spent in the execution unit. Values may be inaccurate in situations where they cannot be measured exactly, especially with memory operands). The numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays by 50-150 clocks, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give a

similar delay.

Reciprocal throughput: The average number of clock cycles per instruction for a series of independent

instructions of the same kind.

Integer instructions (Pentium Pro, Pentium II and Pentium III)

Instruction	Operands				ops			Latency	Reciprocal
		p0	p1	p01	p2	рЗ	p4		throughput
MOV	r,r/i			1					
MOV	r,m				1				
MOV	m,r/i					1	1		
MOV	r,sr			1					
MOV	m,sr			1		1	1		
MOV	sr,r	8						5	
MOV	sr,m	7			1			8	
MOVSX MOVZX	r,r			1					
MOVSX MOVZX	r,m				1				
CMOVcc	r,r	1		1					
CMOVcc	r,m	1		1	1				
XCHG	r,r			3					
XCHG	r,m			4	1	1	1	high b)	
XLAT				1	1				
PUSH	r/i			1		1	1		
POP	r			1	1				
POP	(E)SP			2	1				
PUSH	m			1	1	1	1		
POP	m			5	1	1	1		
PUSH	sr			2		1	1		
POP	sr			8	1				
PUSHF(D)		3		11		1	1		
POPF(D)		10		6	1				
PUSHA(D)				2		8	8		
POPA(D)				2	8				

LALIE CALIE	I		I	۱,	I	I	I	I	1
LAHF SAHF		4		1				4 -\	
LEA	r,m	1						1 c)	
LDS LES LFS LGS									
LSS	m ,,			8	3				
ADD SUB AND OR XOR	r,r/i			1					
ADD SUB AND OR XOR	r,m			1	1				
ADD SUB AND OR XOR	m,r/i			1	1	1	1		
ADC SBB	r,r/i			2					
ADC SBB	r,m			2	1				
ADC SBB	m,r/i			3	1	1	1		
CMP TEST	r,r/i			1					
CMP TEST	m,r/i			1	1				
INC DEC NEG NOT	r			1					
INC DEC NEG NOT	m			1	1	1	1		
AAA AAS DAA DAS			1						
AAD		1		2				4	
AAM		1	1	2				15	
IMUL	r,(r),(i)	1						4	1
IMUL	(r),m	1			1			4	1
DIV IDIV	r8	2		1				19	12
DIV IDIV	r16 r32	3 3		1 1				23 39	21 37
DIV IDIV	m8	2		1	1			19	12
DIV IDIV	m16	2		1	1			23	21
DIV IDIV	m32	2			1			39	37
CBW CWDE	IIIOZ	_		1	'				
CWD CDQ		1							
SHR SHL SAR ROR		•							
ROL	r,i/CL	1							
SHR SHL SAR ROR	,								
ROL	m,i/CL	1			1	1	1		
RCR RCL	r,1	1		1					
RCR RCL	r8,i/CL	4		4					
RCR RCL	r16/32,i/CL	3		3					
RCR RCL	m,1	1		2	1	1	1		
RCR RCL	m8,i/CL	4		3	1	1	1		
RCR RCL	m16/32,i/CL	4		2	1	1	1		
SHLD SHRD	r,r,i/CL	2							
SHLD SHRD	m,r,i/CL	2		1	1	1	1		
ВТ	r,r/i			1					
ВТ	m,r/i	1		6	1				
BTR BTS BTC	r,r/i			1					
BTR BTS BTC	m,r/i	1		6	1	1	1		
BSF BSR	r,r		1	1					
BSF BSR	r,m		1	1	1				
SETcc	r			1					
SETcc	m 			1		1	1		_
JMP	short/near	. .	1						2
JMP	far	21			1				
JMP	r		1						2
JMP	m(near)	24	1		1				2
JMP	m(far)	21	4		2				
conditional jump	short/near		1						2

CALL	near		1	1		1	1		2
CALL	far	28			1	2	2		
CALL	r		1	2		1	1		2
CALL	m(near)		1	4	1	1	1		2 2
CALL	m(far)	28			2	2	2		
RETN	,		1	2	1				2
RETN	i		1	3	1				2
RETF	-	23			3				_
RETF	i	23			3				
J(E)CXZ	short		1	1					
LOOP	short	2	1	8					
LOOP(N)E	short	2	1	8					
ENTER	i,0	_	-	12		1	1		
ENTER	a,b	ca.	18	+4b		b-1	2b		
LEAVE	۵,۵	54.	.	2	1				
BOUND	r,m	7		6	2				
CLC STC CMC	,,	•		1	_				
CLD STD				4					
CLI		9							
STI		17							
INTO		''		5					
LODS					2				
REP LODS			10+6	i in	_				
STOS					1	1	1		
REP STOS			ca. 5	n	a)	•			
MOVS				1	3	1	1		
REP MOVS			ca. 6	1	a)	-	-		
SCAS				1	2				
REP(N)E SCAS			12+7	1	_				
CMPS				4	2				
REP(N)E CMPS			12+9	1					
BSWAP	r	1		1					
NOP (90)				1					0.5
Long NOP (0F 1F)				1					1
CPUID		23-48	I	•					•
RDTSC		31							
IN		18						>300	
OUT		18						>300	
PREFETCHNTA d)	m				1				
PREFETCHT0/1/2 d)	m m				1				
SFENCE d)						1	1		6
Notes			1		l	•	<u>'</u>		<u> </u>

Notes

a) Faster under certain conditions: see manual 3: "The microarchitecture of Intel,

AMD and VIA CPUs".

b) Has an implicit LOCK prefix.

c) 3 if constant without base or index register

d) P3 only.

Floating point x87 instructions (Pentium Pro, II and III)

Instruction	Operands			μα	ps	Latency	Reciprocal		
		p0	p1	p01	p2	р3	p4		throughput
FLD	r	1							

FLD	m32/64			1			1	
FLD	m80	2		2				
FBLD	m80	38		2				
FST(P)	r	1						
FST(P)	m32/m64				1	1	1	
FSTP	m80	2			2	2		
FBSTP	m80	165			2	2		
FXCH	r						0	⅓ f)
FILD	m	3		1			5	
FIST(P)	m	2			1	1	5	
FLDZ		1						
FLD1 FLDPI FLDL2E etc.		2						
FCMOVcc	r	2					2	
FNSTSW	AX	3					7	
FNSTSW	m16	1			1	1		
FLDCW	m16	1	1	1			10	
FNSTCW	m16	1 1			1	1		
FADD(P) FSUB(R)(P)	r	1 1			· ·		3	1
FADD(P) FSUB(R)(P)	m .	1 1		1			3-4	1
FMUL(P)	r	1 1					5	2 g)
FMUL(P)	m '			1			5-6	2 g)
FDIV(R)(P)	r	1 1		'			38 h)	37
		1 1		1			38 h)	37
FDIV(R)(P) FABS	m	1 1		1			30 11)	31
FCHS		3					2	
	_						2	
FCOM(P) FUCOM	r	1		4			1	
FCOM(P) FUCOM	m	1		1			1	
FCOMPP FUCOMPP		1 1	1				1	
FCOMI(P) FUCOMI(P)	r	1					1	
FCOMI(P) FUCOMI(P)	m	1		1			1	
FIADD FISUB(R)	m	6		1				
FIMUL	m	6		1				
FIDIV(R)	m	6		1				
FICOM(P)	m	6		1				
FTST		1					1	
FXAM		1					2	
FPREM		23						
FPREM1		33						
FRNDINT		30						
FSCALE		56						
FXTRACT		15						
FSQRT		1					69	e,i)
FSIN FCOS		17-97				27-103	e)	
FSINCOS		18-110				29-130	e)	
F2XM1		17-48				66	e)	
FYL2X		36-54				103	e)	
FYL2XP1		31-53				98-107	e)	
FPTAN		21-102				13-143	e)	
FPATAN		25-86				44-143	e)	
FNOP		1					,	
FINCSTP FDECSTP		1 1						
FFREE	r	1 1						
FFREEP	r	2						
\	'	- 1			I	I	l	l

FNCLEX		3			
FNINIT	13				
FNSAVE	141				
FRSTOR	72				
WAIT		2			

Notes:

e) Not pipelined

f) FXCH generates 1 μop that is resolved by register renaming without going to any

port.

g) FMUL uses the same circuitry as integer multiplication. Therefore, the combined

throughput of mixed floating point and integer multiplications is 1 FMUL + 1 IMUL

per 3 clock cycles.

h) FDIV latency depends on precision specified in control word: 64 bits precision

gives latency 38, 53 bits precision gives latency 32, 24 bits precision gives latency 18. Division by a power of 2 takes 9 clocks. Reciprocal throughput is 1/(la-

tency-1).

i) Faster for lower precision.

Integer MMX instructions (Pentium II and Pentium III)

Instruction	Operands			μ	ops	Latency	Reciprocal		
	•	p0	p1	p01		р3	p4]	throughput
MOVD MOVQ	r,r			1				1	0.5
MOVD MOVQ	mm,m32/64				1				1
MOVD MOVQ	m32/64,mm					1	1		1
PADD PSUB PCMP	mm,mm			1				1	0.5
PADD PSUB PCMP	mm,m64			1	1				1
PMUL PMADD	mm,mm	1						3	1
PMUL PMADD	mm,m64	1			1			3	1
PAND(N) POR PXOR	mm,mm			1				1	0.5
PAND(N) POR PXOR	mm,m64			1	1				1
PSRA PSRL PSLL	mm,mm/i		1					1	1
PSRA PSRL PSLL	mm,m64		1		1				1
PACK PUNPCK	mm,mm		1					1	1
PACK PUNPCK	mm,m64		1		1				1
EMMS		11						6 k)	
MASKMOVQ d)	mm,mm			1		1	1	2-8	2 - 30
PMOVMSKB d)	r32,mm		1					1	1
MOVNTQ d)	m64,mm					1	1		1 - 30
PSHUFW d)	mm,mm,i		1					1	1
PSHUFW d)	mm,m64,i		1		1			2	1
PEXTRW d)	r32,mm,i		1	1				2	1
PINSRW d)	mm,r32,i		1					1	1
PINSRW d)	mm,m16,i		1		1			2	1
PAVGB PAVGW d)	mm,mm			1				1	0.5
PAVGB PAVGW d)	mm,m64			1	1			2	1
PMIN/MAXUB/SW d)	mm,mm			1				1	0.5
PMIN/MAXUB/SW d)	mm,m64			1	1			2	1
PMULHUW d)	mm,mm	1						3	1
PMULHUW d)	mm,m64	1			1			4	1
PSADBW d)	mm,mm	2		1				5	2
PSADBW d)	mm,m64	2		1	1			6	2

Notes:

d) P3 only.

The delay can be hidden by inserting other instructions between EMMS and any subsequent floating point instruction.

Floating point XMM instructions (Pentium III)

Instruction	Operands		Latency	Reciprocal					
	•	р0	p1	p01	p2	р3	p4	7	throughput
MOVAPS	xmm,xmm		•	2		•	•	1	1
MOVAPS	xmm,m128				2			2	2
MOVAPS	m128,xmm					2	2	3	2
MOVUPS	xmm,m128				4			2	4
MOVUPS	m128,xmm		1			4	4	3	4
MOVSS	xmm,xmm		'	1		'		1	1
MOVSS	xmm,m32			1	1			1	1
MOVSS	m32,xmm			'		1	1	1	1
MOVHPS MOVLPS	xmm,m64			1		'	'		1
MOVHPS MOVLPS	m64,xmm			'		1	1		1
MOVLHPS MOVHLPS	xmm,xmm			1		'	'		1
MOVMSKPS	r32,xmm	1		'				1	1
MOVNTPS	m128,xmm	'				2	2	'	2 - 15
CVTPI2PS			2			2		3	
CVTPI2PS CVTPI2PS	xmm,mm xmm,m64		2		1			4	1 2
					1				
CVT(T)PS2PI	mm,xmm		2					3	1
CVTPS2PI	mm,m128		1		2			4	1
CVTSI2SS	xmm,r32		2		1			4	2
CVTSI2SS	xmm,m32		2		2			5	2
CVT(T)SS2SI	r32,xmm		1		1			3	1
CVTSS2SI	r32,m128		1		2			4	2
ADDPS SUBPS	xmm,xmm		2					3	2
ADDPS SUBPS	xmm,m128		2		2			3	2
ADDSS SUBSS	xmm,xmm		1					3	1
ADDSS SUBSS	xmm,m32		1		1			3	1
MULPS	xmm,xmm	2						4	2
MULPS	xmm,m128	2			2			4	2
MULSS	xmm,xmm	1						4	1
MULSS	xmm,m32	1			1			4	1
DIVPS	xmm,xmm	2						48	34
DIVPS	xmm,m128	2			2			48	34
DIVSS	xmm,xmm	1						18	17
DIVSS	xmm,m32	1			1			18	17
AND(N)PS ORPS XORPS	xmm,xmm		2					2	2
AND(N)PS ORPS XORPS	xmm,m128		2		2			2	2
MAXPŚ MINPS	xmm,xmm		2					3	2
MAXPS MINPS	xmm,m128		2		2			3	2
MAXSS MINSS	xmm,xmm		1					3	1
MAXSS MINSS	xmm,m32		1		1			3	1
CMPccPS	xmm,xmm		2					3	2
CMPccPS	xmm,m128		2		2			3	2
CMPccSS	xmm,xmm		1		-			3	1
CMPccSS	xmm,m32		1		1			3	1
COMISS UCOMISS	xmm,xmm		1		'			1	1
COMISS UCOMISS	xmm,m32		1		1			1	1
COMISS OCCIVISS	الاللاللا		1 1		'			1	1

SQRTPS	xmm,xmm	2					56	56
SQRTPS	xmm,m128	2			2		57	56
SQRTSS	xmm,xmm	2					30	28
SQRTSS	xmm,m32	2			1		31	28
RSQRTPS	xmm,xmm	2					2	2
RSQRTPS	xmm,m128	2			2		3	2
RSQRTSS	xmm,xmm	1					1	1
RSQRTSS	xmm,m32	1			1		2	1
RCPPS	xmm,xmm	2					2	2
RCPPS	xmm,m128	2			2		3	2
RCPSS	xmm,xmm	1					1	1
RCPSS	xmm,m32	1			1		2	1
SHUFPS	xmm,xmm,i		2	1			2	2
SHUFPS	xmm,m128,i		2		2		2	2
UNPCKHPS UNPCKLPS	xmm,xmm		2	2			3	2
UNPCKHPS UNPCKLPS	xmm,m128		2		2		3	2
LDMXCSR	m32	11					15	15
STMXCSR	m32	6					7	9
FXSAVE	m4096	116					62	
FXRSTOR	m4096	89					68	

Intel Pentium M, Core Solo and Core Duo

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, xmm =

128 bit xmm register, sr = segment register, m = memory, m32 =

32-bit memory operand, etc.

μορs fused domain: The number of μορs at the decode, rename, allocate and retire-

ment stages in the pipeline. Fused uops count as one.

μops unfused domain: The number of μops for each execution port. Fused μops count

as two.

p0: Port 0: ALU, etc.p1: Port 1: ALU, jumps

p01: Instructions that can go to either port 0 or 1, whichever is vacant

first.

p2: Port 2: load data, etc.

p3: Port 3: address generation for store

p4: Port 4: store data

Latency: This is the delay that the instruction generates in a dependency

chain. (This is not the same as the time spent in the execution unit. Values may be inaccurate in situations where they cannot be measured exactly, especially with memory operands). The numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays by 50-150 clocks, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give

a similar delay.

Reciprocal throughput: The average number of clock cycles per instruction for a series of

independent instructions of the same kind.

Integer instructions

Instruction	Operands	µops fused	μ	ops	unfus	sed d	Latency	Recipro- cal		
		domain	p0	p1	p01	p2	р3	p4		through- put
Move instructions										
MOV	r,r/i	1			1					0.5
MOV	r,m	1				1				1
MOV	m,r	1					1	1		1
MOV	m,i	2					1	1		1
MOV	r,sr	1			1					
MOV	m,sr	2			1		1	1		
MOV	sr,r	8	8						5	
MOV	sr,m	8	7			1			8	
MOVNTI	m,r32	2					1	1		2
MOVSX MOVZX	r,r	1			1				1	0.5
MOVSX MOVZX	r,m	1				1				1
CMOVcc	r,r	2	1		1				2	1.5
CMOVcc	r,m	2	1		1	1				
XCHG	r,r	3			3				2	1.5
XCHG	r,m	7			4	1	1	1	high b)	
XLAT		2			1	1				1

Pentium M

PUSH PUSH PUSH PUSH PUSH(D) PUSHA(D) POP POP POP POP POP POP POPA(D) LAHF SAHF SALC LEA BSWAP LDS LES LFS LGS LSS PREFETCHNTA PREFETCHTO/1/2 SFENCE/LFENCE/MFENCE	r i m sr r (E)SP m sr r,m r m m	1 2 2 16 18 1 3 2 10 17 10 1 2 11 1 2	10 1 1 1 1 18 18	1	1 11 2 9 6 2 1	1 1 1 1 1 8 3 1 1	1 1 1 8 1 1	1 1 1 1 8 1 1	1 1 2 8 2 7 1 1 >300 >300	1 1 6 8 1 16 7 1 1 1
Arithmetic instructions ADD SUB ADD SUB ADD SUB ADC SBB ADC SBB ADC SBB CMP CMP INC DEC NEG NOT INC DEC NEG NOT AAA AAS DAA DAS AAD AAM	r,r/i r,m m,r/i r,r/i r,m m,r/i r,r/i m,r m,i r	1 1 3 2 2 7 1 1 2 1 3 1 3	1 1	1 1	1 1 1 1 1 4 1 1 1 1 1 2 2	1 1 1 1 1 1	1 1	1 1	1 2 2 1 1 1	0.5 1 2 0.5 1 1 0.5
MUL IMUL MUL IMUL IMUL IMUL MUL IMUL MUL IMUL IMUL IMUL IMUL IMUL DIV IDIV DIV IDIV DIV IDIV DIV IDIV DIV IDIV DIV IDIV CBW CWDE	r8 r16/r32 r,r r,r,i m8 m16/m32 r,m r,m,i r8 r16 r32 m8 m16 m32	4 1 3 1 1 3 1 2 5 4 6 5 5 1	1 1 3 1 1 1 3 1 1 4 3 4 3 4 3	1 1	1 1 1 1 1 1	1 1 1 1 1 1			15 4 5 4 4 4 5 4 15-16 c) 15-24 c) 15-39 c) 15-39 c) 15-39 c)	1 1 1 1 1 1 1 1 12 12-20 c) 12-20 c) 12-20 c) 12-20 c)

AND OR XOR AND OR XOR AND OR XOR T, m T, m T, m T, m T, m T, m T, m T, m	CWD CDQ		1		1					1	1
AND OR XOR AND OR XOR AND OR XOR AND OR XOR AND OR XOR T, m TEST TEST TEST TEST TEST TEST TEST TO M,	Logic instructions										
AND OR XOR AND OR XOR Mn,r/i TEST TEST TEST Mn,r 1	AND OR XOR	r,r/i	1			1				1	0.5
AND OR XOR TEST	AND OR XOR		1			1	1			2	1
TEST	AND OR XOR		3			1	1	1	1		1
TEST	TEST	· ·	1			1				1	0.5
TEST	TEST	· ·				1	1				
SHR SHL SAR ROR ROL SHR SHL SAR ROR ROL SHR SHL SAR ROR ROL SHR SHL SAR ROR ROL SHR SHL SAR ROR ROL SHR SHL SAR ROR ROL SHR SHL SAR ROR ROL SHR SHL SAR ROR ROL SH SAR SH SAR SAR SAR SAR SAR SAR SAR SAR SAR SAR	TEST					1					
SHR SHL SAR ROR ROL RCR RCL RCR RCL RCR RCR RCR RCR RCR RCR RCR RCR RCR				1						1	
RCR RCL RCR RCR RCR RCR RCR RCL R6,I/CL R6,I/CL RCR RCR RCR RCR RCR RCR RCR RCR RCR R				1			1	1	1	-	
RCR RCL RCL RCL RCL RCL RCL RCR RCL RCR RCCR RCC RCC				1		1			-	2	2
RCL				1 -							_
RCR RCL RCR RCR RCR RCL RCR RCL RCL RC RC RCR RCT T 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1	1				1 -					
RCR RCL RCR RCR RMS,I/CL RCR RCL RCS,I/CL RCR RCCL RCR RCL RCR RCL RCCR RCL RCCR RCCL RCCL RCCR RCCR RCCR RCCL RCCR RCCR RCCR RCCL RCCR RCT N 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1			1 -							9
RCR RCL				1			1	1	1		
RCL RCR RCR RCL RCR RCR RCL RCR RCR RCL RCR RCR RCL RCR RCR RCL RCR RCR RCL RCR RCR RCR RCR RCR RCR RCR RCR RCR RCR	1	1						'	_		
RCR RCL m16/32,i/CL 10 5 2 1 1 1 2 2 2 3 5 5 2 1 1 1 2 2 2 3 5 5 5 5 5 2 3 5 5 5 5 5 5 5 5 5	1										
SHLD SHRD SHLD SHLD SHITER		1		1							
SHLD SHRD Mi,r,i/CL 4				1		_	'	'	'	2	2
BT r,r/i 1 1 7 1 1 1 1 1 1 1		1 1				1	1	1	1		
BT				'	1	'	'	' '	'	1	1
BT					'	7	1			!	'
BTR BTS BTC BT 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		·			1	'					
BTR BTS BTC BTR BTS BTC BTR BTS BTC BTR BTS BTC BTR BTS BTC BSF BSR							'				
BTR BTS BTC BSF BSR BSF BSR R,r,r R,r RSETcc R SETcc R R R R R R R R R R R R R R R R R R					ı	7	1	1	1	6	
BSF BSR F,r 2					1	′	1 -	-	_		
SEF BSR SETCC F						1	'	' '	'		
SETCC					-	1	1				
SETCC m 2 1 1 1 1 1 1 1 1 1						'	'				
CLC STC CMC								1	1		
CLD STD 4		""			-			'	1		1
Control transfer instructions JMP	1				1	1					
Short/near 1	CLD 31D		4			4					'
JMP far 22 21 1 1 28 JMP r 1 1 1 1 1 JMP m(near) 2 1 1 1 2 JMP m(far) 25 23 2 31 conditional jump short/near 1 1 1 1 J(E)CXZ short 2 1 1 1 1 LOOP short 11 2 1 8 6 6 LOOP(N)E short 11 2 1 8 6 6 CALL near 4 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 1 1 1 1 1 1 2 3 3 3 3 3 3 <t< td=""><td>Control transfer instructio</td><td>ns</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></t<>	Control transfer instructio	ns									
JMP r 1 1 1 1 2 1 1 2 31 1 2 31 32 32 32 32 32	JMP	short/near	1		1						1
JMP m(near) 2 1 1 2 31 JMP m(far) 25 23 2 31 conditional jump short/near 1 1 1 1 J(E)CXZ short 2 1 1 1 1 LOOP short 11 2 1 8 6 6 LOOP(N)E short 11 2 1 8 6 6 6 CALL near 4 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 30 2 30 3 <td>JMP</td> <td>far</td> <td>22</td> <td>21</td> <td></td> <td></td> <td>1</td> <td></td> <td></td> <td></td> <td>28</td>	JMP	far	22	21			1				28
JMP conditional jump m(far) short/near 25 23 2 2 31 J(E)CXZ short 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 <	JMP	r	1		1						1
conditional jump short/near 1 2 <td>JMP</td> <td>m(near)</td> <td>2</td> <td></td> <td>1</td> <td></td> <td>1</td> <td></td> <td></td> <td></td> <td>2</td>	JMP	m(near)	2		1		1				2
Short 2	JMP	m(far)	25	23			2				31
Short 11 2 1 8 6 6 6 6 6 6 6	conditional jump	short/near	1		1						1
Short 11 2 1 8	J(E)CXZ	short	2		1	1					1
CALL near 4 1 1 1 1 1 2 CALL far 32 27 1 2 27 CALL r 4 1 2 1 1 9 CALL m(near) 4 1 1 1 1 1 CALL m(far) 35 29 2 2 2 30 RETN 2 1 2 1 1 1 2 RETN i 3 1 1 1 2 RETF 27 24 3 3 3	LOOP	short	11	2	1	8					6
CALL far 32 27 1 2 2 27 CALL r 4 1 2 1 1 1 9 CALL m(near) 4 1 1 1 1 1 2 CALL m(far) 35 29 2 2 2 30 RETN 2 1 2 1 1 1 2 RETN 3 1 1 1 1 2 RETF 27 24 3 3 3 30	LOOP(N)E	short	11	2	1	8					6
CALL r 4 1 2 1 1 1 9 CALL m(near) 4 1 1 1 1 1 2 CALL m(far) 35 29 2 2 2 30 RETN 2 1 2 1 1 1 2 RETN i 3 1 1 1 1 2 RETF 27 24 3 3 30	CALL	near	4		1	1		1	1		2
CALL r 4 1 2 1 1 1 9 CALL m(near) 4 1 1 1 1 1 2 CALL m(far) 35 29 2 2 2 30 RETN 2 1 2 1 1 1 2 RETN i 3 1 1 1 1 2 RETF 27 24 3 3 30	CALL	far	32	27			1	2	2		
CALL m(near) 4 1 1 1 1 1 2 CALL m(far) 35 29 2 2 2 30 RETN 2 1 2 1 2 2 RETN 3 1 1 1 1 2 RETF 27 24 3 3 30	CALL	r	4		1	2		1	1		
CALL m(far) 35 29 2 2 2 30 RETN 2 1 2 1 2 1 2 RETN 3 1 1 1 1 2 RETF 27 24 3 3 30	CALL	m(near)			1		1	1	1		
RETN 2 1 2 1 2 2 RETN 2 2 RETF 2 30	CALL			29			2	2	2		
RETN i 3 1 1 1 2 2 30	RETN	, ,			1	2	1				
RETF 27 24 3 30	RETN	i			1		1				
	RETF			24							
	RETF	i	27	24			3				30

BOUND	r,m	15	7		6	2				8
INTO		5			5					4
String instructions										_
LODS		2				2				4
REP LODS		6n		•	10+6r	1				0.5
STOS		3				1	1	1		1
REP STOS		5n		(a. 5r	,				0.7
MOVS		6			1	3	1	1		0.7
REP MOVS		6n		(ca. 6r	a)				0.5
SCAS		3			1	2				1.3
REP(N)E SCAS		7n		•	12+7r	ı				0.6
CMPS		6			4	2				0.7
REP(N)E CMPS		9n		•	12+9r	1				0.5
Other										
NOP (90)		1			1					0.5
Long NOP (0F 1F)		1			1					1
PAUSE		2			2					
CLI			9							
STI			17							
ENTER	i,0	12			10		1	1		
ENTER	a,b		ca.	18	+4b		b-1	2b		
LEAVE		3			2	1				
CPUID		38-59	38-59)					ca. 130	
RDTSC		13	13							42

Notes:

a) Faster under certain conditions: see manual 3: "The microarchitecture of Intel, AMD and VIA CPUs".

b) Has an implicit LOCK prefix.

c) High values are typical, low values are for round divisors. Core Solo/Duo is more efficient than Pentium M in cases with round values that allow an early-

out algorithm.

Floating point x87 instructions

Instruction	Operands	μοps fused	μ	ops	unfus	sed d	omai	in	Latency	Recipro- cal
		domain	p0	p1	p01	p2	р3	p4		through- put
Move instructions										•
FLD	r	1	1						1	
FLD	m32/64	1				1			1	
FLD	m80	4	2			2				
FBLD	m80	40	38			2				
FST(P)	r	1	1							
FST(P)	m32/m64	1					1	1	1	
FSTP	m80	6	2				2	2		3
FBSTP	m80	169	165				2	2		167
FXCH	r	1							0	0.33 f)
FILD	m	4	3			1			5	2
FIST(P)	m	4	2				1	1	5	2
FISTTP g)	m	4	2				1	1	5	2

FLDZ FLD1 FLDPI FLDL2E etc. FCMOVcc FNSTSW FNSTSW FLDCW FNSTCW FINCSTP FDECSTP FFREE FFREEP FNSAVE FRSTOR	r AX m16 m16 m16	1 2 2 3 2 3 3 1 1 2 142 72	1 2 2 3 1 1 1 1 1 2 142 72		1	1	1	1 1	2 7	3 19 3 1 2 131 91
Arithmetic instructions FADD(P) FSUB(R)(P) FADD(P) FSUB(R)(P) FMUL(P) FMUL(P) FDIV(R)(P) FDIV(R)(P) FABS FCHS FCOM(P) FUCOM FCOM(P) FUCOM FCOMPP FUCOMPP FCOMI(P) FUCOMI(P) FIADD FISUB(R) FIMUL FIDIV(R) FICOM(P) FTST FXAM FPREM FPREM1 FRNDINT	r m r m r m r m	1 1 1 1 1 1 1 1 1 1 2 1 6 6 6 1 1 26 15	1 1 1 1 1 1 3 5 5 3	1 1 1 1 2 1	1 1 1	1 1 1 1 1 1			3 5 5 9-38 c) 9-38 c) 1 1 1 1 1 3 5 9-38 c)	1 1 2 2 8-37 c) 8-37 c) 1 1 1 1 3 3 3 8-37 c) 4 1
Math FSCALE FXTRACT FSQRT FSIN FCOS FSINCOS F2XM1 FYL2X FYL2XP1 FPTAN FPATAN Other FNOP WAIT			28 1 80-10 90-17 ~20 ~40 ~55 ~100 ~85	10	15			80-1 100- ~45 ~60 ~65 ~140 ~140	130	8 1 1
FNCLEX FNINIT		3 14	3 14	'	'					13 27

Notes:

- c) High values are typical, low values are for low precision or round divisors.
- f) FXCH generates 1 μop that is resolved by register renaming without going to
- g) SSE3 instruction only available on Core Solo and Core Duo.

Integer MMX and XMM instructions

Instruction	Operands	μορs fused	μ	ops	unfus	sed d	lomai	in	Latency	Recipro- cal
		domain	p0	p1	p01	p2	р3	p4		through- put
Move instructions										
MOVD	r32,mm	1			1				1	0.5
MOVD	mm,r32	1			1				1	0.5
MOVD	mm,m32	1				1				1
MOVD	m32,mm	1					1	1		1
MOVD	r32,xmm	1		1					1	1
MOVD	xmm,r32	2			2					1
MOVD	xmm,m32	2			1	1				1
MOVD	m32, xmm	1					1	1		1
MOVQ	mm,mm	1			1					0.5
MOVQ	mm,m64	1				1				1
MOVQ	m64,mm	1					1	1		1
MOVQ	xmm,xmm	2			2				1	1
MOVQ	xmm,m64	2			1	1			-	1
MOVQ	m64, xmm	1				-	1	1		1
MOVDQA	xmm, xmm	2			2		•	-	1	1
MOVDQA	xmm, m128	2			-	2				2
MOVDQA	m128, xmm	2				_	2	2		2
MOVDQU	xmm, m128	4			2	2	_	_		2-10
MOVDQU	m128, xmm	8			5-6	_	2-3	2-3		4-20
LDDQU g)	xmm, m128	4								2
MOVDQ2Q	mm, xmm	1		1					1	1
MOVQ2DQ	xmm,mm	2		1	1				1	1
MOVNTQ	m64,mm	1		ļ .	'		1	1		2
MOVNTDQ	m128,xmm	4					2	2		3
PACKSSWB/DW	111120,7111111	7					_	_		
PACKUSWB	mm,mm	1	1						1	1
PACKSSWB/DW		-							-	-
PACKUSWB	mm,m64	1	1			1			1	1
PACKSSWB/DW										
PACKUSWB	xmm,xmm	3	2	1					2	2
PACKSSWB/DW										
PACKUSWB	xmm,m128	4	1	1		2			2	2
PUNPCKH/LBW/WD/DQ	mm,mm	1	1						1	1
PUNPCKH/LBW/WD/DQ	mm,m64	1	1			1				1
PUNPCKH/LBW/WD/DQ	xmm,xmm	2	2						2	2
PUNPCKH/LBW/WD/DQ	xmm,m128	3	1			2				2
PUNPCKHQDQ	xmm,xmm	2		1	1				1	1
PUNPCKHQDQ	xmm, m128	3		1		2				1
PUNPCKLQDQ	xmm,xmm	1		1					1	1
PUNPCKLQDQ	xmm, m128	1				1				1
PSHUFW	mm,mm,i	1	1						1	1

PSHUFW	mm,m64,i	2	1			1				1
PSHUFD	xmm,xmm,i	3	2	1					2	2
PSHUFD	xmm,m128,i	4	1	1		2				2
PSHUFL/HW	xmm,xmm,i	2	1	1						1
PSHUFL/HW	xmm, m128,i	3		1		2				1
MASKMOVQ	mm,mm	3			1		1	1		
MASKMOVDQU	xmm,xmm	8		1			2	2		
PMOVMSKB	r32,mm	1	1						1	1
PMOVMSKB	r32,xmm	1	1	j)					1	1
PEXTRW	r32,mm,i	2	1	1					2	1
PEXTRW	r32,xmm,i	4	2	2					3	2
PINSRW	mm,r32,i	1	1						1	1
PINSRW	xmm,r32,i	2	2						1	2
Arithmetic instructions										
PADD/SUB(U)(S)B/W/D	mm,mm	1			1				1	0.5
PADD/SUB(U)(S)B/W/D	mm,m64	1				1			'	1
PADD/SUB(U)(S)B/W/D	xmm,xmm	2			2	'			1	1
PADD/SUB(U)(S)B/W/D	xmm,m128	4			2	2			'	2
PADDQ PSUBQ	· ·	2			2	~			2	1
PADDQ PSUBQ	mm,mm mm,m64	2			2	1				1
PADDQ PSUBQ	xmm,xmm	4			4	'			2	2
PADDQ PSUBQ	xmm,m128	6			4	2				2
PCMPEQ/GTB/W/D	1	1			1	~			1	0.5
	mm,mm				1	1				
PCMPEQ/GTB/W/D	mm,m64	1			1 2				4	1
PCMPEQ/GTB/W/D	xmm,xmm	2				_			1	1 2
PCMPEQ/GTB/W/D	xmm,m128	2			2	2				
PMULL/HW PMULHUW	mm,mm	1			1	_			3	1
PMULL/HW PMULHUW	mm,m64	1			1	1			3	1
PMULL/HW PMULHUW	xmm,xmm	2			2				3	2
PMULL/HW PMULHUW	xmm,m128	4			2	2			3	2
PMULUDQ	mm,mm	1	1						4	1
PMULUDQ	mm,m64	1	1			1			4	1
PMULUDQ	xmm,xmm	2	2						4	2
PMULUDQ	xmm,m128	4	2			2			4	2
PMADDWD	mm,mm	1			1				3	1
PMADDWD	mm,m64	1			1	1			3	1
PMADDWD	xmm,xmm	2			2				3	2
PMADDWD	xmm,m128	4			2	2			3	2
PAVGB/W	mm,mm	1			1				1	0.5
PAVGB/W	mm,m64	1			1	1			_	1
PAVGB/W	xmm,xmm	2			2	_			1	1
PAVGB/W	xmm,m128	4			2	2				2
PMIN/MAXUB/SW	mm,mm	1			1				1	0.5
PMIN/MAXUB/SW	mm,m64	1			1	1				1
PMIN/MAXUB/SW	xmm,xmm	2			2				1	1
PMIN/MAXUB/SW	xmm,m128	4			2	2				2
PSADBW	mm,mm	2			2				4	1
PSADBW	mm,m64	2			2	1			4	1
PSADBW	xmm,xmm	4			4				4	2
PSADBW	xmm,m128	6			4	2			4	2
Logic instructions										

PAND(N) POR PXOR	mm,mm	1			1			1	0.5
PAND(N) POR PXOR	mm,m64	1			1	1			1
PAND(N) POR PXOR	xmm,xmm	2			2			1	1
PAND(N) POR PXOR	xmm,m128	4			2	2			2
PSLL/RL/RAW/D/Q	mm,mm/i	1	1					1	1
PSLL/RL/RAW/D/Q	mm,m64	1	1			1			1
PSLL/RL/RAW/D/Q	xmm,i	2	2					2	2
PSLL/RL/RAW/D/Q	xmm,xmm	3	2	1				2	2
PSLL/RL/RAW/D/Q	xmm,m128	3		1		2			2
PSLL/RLDQ	xmm,i	4	3	1				3	3
Other									
EMMS		11			11			6 k)	6

Notes:

SSE3 instruction only available on Core Solo and Core Duo. g)

j) Also uses some execution units under port 1.

You may hide the delay by inserting other instructions between EMMS and any subsequent floating point instruction. k)

Floating point XMM instructions

Instruction	Operands	µops fused	μ	ops	unfus	sed d	oma	in	Latency	Recipro- cal
		domain	p0	p1	p01	p2	р3	p4		through- put
Move instructions										
MOVAPS/D	xmm,xmm	2			2				1	1
MOVAPS/D	xmm,m128	2				2			2	2
MOVAPS/D	m128,xmm	2					2	2	3	2
MOVUPS/D	xmm,m128	4				4			2	2
MOVUPS/D	m128,xmm	8			4		2	2	3	4
MOVSS/D	xmm,xmm	1		1					1	1
MOVSS/D	xmm,m32/64	2		1		1			1	1
MOVSS/D	m32/64,xmm	1					1	1	1	1
MOVHPS/D MOVLPS/D	xmm,m64	1		1		1			1	1
MOVHPS/D MOVLPS/D	m64,xmm	1					1	1	1	1
MOVLHPS MOVHLPS	xmm,xmm	1		1					1	1
MOVMSKPS/D	r32,xmm	1	1	j)					2	1
MOVNTPS/D	m128,xmm	2					2	2		3
SHUFPS/D	xmm,xmm,i	3	2	1					2	2
SHUFPS/D	xmm,m128,i	4	1	1		2				2
MOVDDUP g)	xmm,xmm	2							1	1
MOVSH/LDUP g)	xmm,xmm	2							2	2
MOVSH/LDUP g)	xmm,m128	4								
UNPCKH/LPS	xmm,xmm	4	2	2					3-4	5
UNPCKH/LPS	xmm,m128	4		2		2				5
UNPCKH/LPD	xmm,xmm	2		1	1				1	1
UNPCKH/LPD	xmm,m128	3		1	1	1				1
Conversion										
CVTPS2PD	xmm,xmm	4	2	2					3	3
CVTPS2PD	xmm,m64	4	1	2		1				3
CVTPD2PS	xmm,xmm	4	3	1					4	3

CVTPD2PS	xmm,m128	6	3	1		2		3
CVTSD2SS	xmm,xmm	2			2		4	2
CVTSD2SS	xmm,m64	3			2	1		2
CVTSS2SD	xmm,xmm	2	2				2	2
CVTSS2SD	xmm,m64	3	2			1		2
CVTDQ2PS	xmm,xmm	2			2		3	2
CVTDQ2PS	xmm,m128	4			2	2		2
CVT(T) PS2DQ	xmm,xmm	2			2		3	2
CVT(T) PS2DQ	xmm,m128	4			2	2		2
CVTDQ2PD	xmm,xmm	4			4		4	2
CVTDQ2PD	xmm,m64	5			4	1		2
CVT(T)PD2DQ	xmm,xmm	4			4		4	3
CVT(T)PD2DQ	xmm,m128	6			4	2		3
CVTPI2PS	xmm,mm	1		1			3	1
CVTPI2PS	xmm,m64	2		1		1		1
CVT(T)PS2PI	mm,xmm	1		1			3	1
CVT(T)PS2PI	mm,m128	2		1		1		1
CVTPI2PD	xmm,mm	4	2	2		_	5	2
CVTPI2PD	xmm,m64	5	2	2		1		2
CVT(T) PD2PI	mm,xmm	3	_	_	3		4	2
CVT(T) PD2PI	mm,m128	5			3	2		2
CVTSI2SS	xmm,r32	2	1	1		_	4	1
CVT(T)SS2SI	r32,xmm	2	'	1	1		4	1
CVT(T)SS2SI	r32,m32	3		1	1	1		1
CVTSI2SD	xmm,r32	2	1	1	'	'	4	1
CVTSI2SD	xmm,m32	3		1		1	-	1
CVT(T)SD2SI	r32,xmm	2	'	1	1	'	4	1
CVT(T)SD2SI	r32,m64	3		1	1	1	-	1
OV1(1)0D201	132,11104	3		'	'	'		•
Arithmetic								
ADDSS/D SUBSS/D	xmm,xmm	1			1		3	1
ADDSS/D SUBSS/D	xmm,m32/64	2			1	1	3	1
ADDPS/D SUBPS/D	xmm,xmm	2			2	'	3	2
ADDPS/D SUBPS/D	xmm,m128	4			2	2	3	2
ADDSUBPS/D g)	xmm,xmm	2			2		3	2
HADDPS HSUBPS g)	xmm,xmm	6?			?		7	4
HADDPD HSUBPD g)	xmm,xmm	3			3		4	2
MULSS	xmm,xmm	1	1				4	1
MULSD	xmm,xmm	1					5	2
MULSS	xmm,m32	2				1	4	1
MULSD	xmm,m64	2					5	2
MULPS		2	2			' '	4	2
MULPD	xmm,xmm	2	2				5	4
MULPS	xmm,xmm xmm,m128	4	2			2	4	2
MULPD		4	2			2	5	4
	xmm,m128						1	
DIVSS	xmm,xmm	1	1				9-18 c)	8-17 c)
DIVSD	xmm,xmm	1	1			,	9-32 c)	8-31 c)
DIVSS	xmm,m32	2	1			1	9-18 c)	8-17 c)
DIVSD	xmm,m64	2	1			1	9-32 c)	8-31 c)
DIVPS	xmm,xmm	2	2				16-34 c)	16-34 c)
DIVPD	xmm,xmm	2	2				16-62 c)	16-62 c)
DIVPS DIVPD	xmm,m128	4	2			2	16-34 c)	16-34 c)
11 11 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	xmm,m128	4	2			2	16-62 c)	16-62 c)

CMPccSS/D	xmm,xmm	1			1		1		3	1 1
CMPccSS/D	xmm,m32/64	2			1	1			_	1
CMPccPS/D	xmm,xmm	2			2				3	2
CMPccPS/D	xmm,m128	4			2	2			_	2
COMISS/D UCOMISS/D	xmm,xmm	1		1						1
COMISS/D UCOMISS/D	xmm,m32/64	2		1		1				1
MAXSS/D MINSS/D	xmm,xmm	1			1				3	1
MAXSS/D MINSS/D	xmm,m32/64	2			1	1			3	1
MAXPS/D MINPS/D	xmm,xmm	2			2				3	2
MAXPS/D MINPS/D	xmm,m128	4			2	2			3	2
RCPSS	xmm,xmm	1		1					3	1
RCPSS	xmm,m32	2		1		1				1
RCPPS	xmm,xmm	2		2					3	2
RCPPS	xmm,m128	4		2		2				2
Math										
SQRTSS	xmm,xmm	2	2						6-30	4-28
SQRTSS	xmm,m32	3	2			1				4-28
SQRTSD	xmm,xmm	1	1						5-58	4-57
SQRTSD	xmm,m64	2	1			1				4-57
SQRTPS	xmm,xmm	2	2						8-56	16-55
SQRTPD	xmm,xmm	2	2						16-114	16-114
SQRTPS	xmm,m128	4	2			2				16-55
SQRTPD	xmm,m128	4	2			2				16-114
RSQRTSS	xmm,xmm	1		1					3	1
RSQRTSS	xmm,m32	2		1		1			_	1
RSQRTPS	xmm,xmm	2		3					3	2
RSQRTPS	xmm,m128	4		2		2				2
Logic										
AND/ANDN/OR/XORPS/D	xmm,xmm	2			2				1	1
AND/ANDN/OR/XORPS/D	xmm,m128	4			2	2			'	1
		•			_	_				•
Other										
LDMXCSR	m32	9	9							20
STMXCSR	m32	6	6							12
FXSAVE	m4096	118	32				43	43		63
FXRSTOR	m4096	87	43			44				72

Notes:

c) High values are typical, low values are for round divisors.

g) SSE3 instruction only available on Core Solo and Core Duo.

j) Also uses some execution units under port 1.

Intel Core 2 (Merom, 65nm)

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, xmm = 128 bit xmm

register, mm/x = mmx or xmm register, sr = segment register, m = memory,

m32 = 32-bit memory operand, etc.

μορs fused domain: The number of μορs at the decode, rename, allocate and retirement stages in

the pipeline. Fused uops count as one.

μορs unfused domain: The number of μορs for each execution port. Fused μορs count as two. Fused

macro-ops count as one. The instruction has μ op fusion if the sum of the numbers listed under p015 + p2 + p3 + p4 exceeds the number listed under μ ops fused domain. An x under p0, p1 or p5 means that at least one of the μ ops listed under p015 can optionally go to this port. For example, a 1 under p015 and an x under p0 and p5 means one μ op which can go to either port 0 or port 5, whichever is vacant first. A value listed under p015 but nothing under p0, p1 and p5 means that it is not known which of the three ports these μ ops go to.

p015: The total number of μops going to port 0, 1 and 5.
p0: The number of μops going to port 0 (execution units).
p1: The number of μops going to port 1 (execution units).
p5: The number of μops going to port 5 (execution units).
p2: The number of μops going to port 2 (memory read).

p3: The number of μops going to port 3 (memory write address).p4: The number of μops going to port 4 (memory write data).

Unit: Tells which execution unit cluster is used. An additional delay of 1 clock cycle

is generated if a register written by a μop in the integer unit (int) is read by a μop in the floating point unit (float) or vice versa. flt—int means that an instruction with multiple μops receive the input in the float unit and delivers the output in the int unit. Delays for moving data between different units are included under latency when they are unavoidable. For example, movd eax,xmm0 has an extra 1 clock delay for moving from the XMM-integer unit to the general purpose integer unit. This is included under latency because it occurs regardless of which instruction comes next. Nothing listed under unit means that additional delays are either unlikely to occur or unavoidable and therefore included in the

latency figure.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give a similar delay. The time unit used is core clock cycles, not the reference clock cycles

given by the time stamp counter.

Reciprocal throughput: The average number of core clock cycles per instruction for a series of inde-

pendent instructions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μορs fused	μops	un	fuse	d d	oma	ain	Unit	Laten- cy	Reci- procal		
		do- main	p015	p015 p0 p1 p5 p2 p3 p4								through- put	
Move instructions MOV	r,r/i	1	1	х	х	х				int	1	0.33	

MOV a) MOV MOV MOV MOV MOV MOV MOV MOVNTI MOVSX MOVZX MOVSXD MOVSX MOVZX CMOVCC CMOVCC XCHG XCHG XLAT PUSH PUSH PUSH PUSH PUSH PUSH PUSH PUSH	r,m m,r m,i r,sr m,sr sr,r sr,m m,r r,r r,m r,r r,m r,r r,m r r m sr r (E/R)SP m sr	1 1 1 1 1 1 2 8 8 2 1 1 2 2 3 7 2 1 1 1 2 2 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1 2 1 1 1 1 2 1 1 1 1 2 1 1 1 1 1 2 1 1 2 1 1 2 1 2 1 1 2 1 1 2 1 2 1 2 1 2 1 1 2 1 2 1 2 1 2 1 1 2 1 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 1 2 1 2 1 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 2 1 2 1 2 1 2 1 2 1 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 2 1 1 1 2 1 2 1 2 2 1 2 1 2 1 2 2 1 2	4 3 1 2 2 3 x 1 1 15 9 3 9 23 2 1 2 11	x x x x x x x 1 1	x	x x x x x x 1	1 1 1 4 5 1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1	int int int int int int int int int int	2 3 3 1 2 high b) 4 3	1 1 1 1 16 16 2 0.33 1 1 2 1 1 1 7 8 1 1 7 0.33 1 1 1 7 0.33 1 1 1 7 0.33 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
PREFETCHNTA PREFETCHT0/1/2	m	1 1	2	x	x	x	1	1 1 1	1 1 1 1	int int	240	1 1
Arithmetic instructions ADD SUB ADD SUB ADD SUB ADC SBB ADC SBB ADC SBB CMP CMP INC DEC NEG NOT INC DEC NEG NOT	r,r/i r,m m,r/i r,r/i r,m m,r/i r,r/i m,r/i r	1 1 2 2 2 4 1 1 1 3	1 1 1 2 2 3 1 1 1	x x x x x x x x	x x x x x x x	x x x x x x x x	1 1 1 1	1 1 1	1 1 1	int int int int int int int int	1 6 2 7 1 1 1 6	0.33 1 1 2 2 0.33 1 0.33

ΛΛΛ ΛΛΩ DΛΛ DΛΩ :\		1	1	I	1					int	I	4
AAA AAS DAA DAS i) AAD i)		1 3	1 3		1	V				int int		1 1
AAD I) AAM i)		3 4	4	X	X	Х				int	17	'
MUL IMUL	r8	1	1		1					int	3	1 1
MUL IMUL	r16	3	3		-					int	5	1.5
MUL IMUL	r32	3	3	X	X	X					5	1.5
	1	ა 3	3	X	X	X				int	7	
MUL IMUL	r64			X	X	Х				int		4
IMUL	r16,r16	1	1		1					int	3	1
IMUL	r32,r32	1	1	,	1					int	3	1 1
IMUL	r64,r64	1	1	1						int	5	2
IMUL	r16,r16,i	1	1		1					int	3	1
IMUL	r32,r32,i	1	1	,	1					int	3	1 1
IMUL	r64,r64,i	1	1	1			4			int	5	2
MUL IMUL	m8	1	1		1		1			int	3	1 1
MUL IMUL	m16	3	3	Х	Х	Х	1			int	5	1.5
MUL IMUL	m32	3	3	X	Х	Х	1			int	5	1.5
MUL IMUL	m64	3	2	2			1			int	7	4
IMUL	r16,m16	1	1		1		1			int	3	1
IMUL	r32,m32	1	1	١.	1		1			int	3	1
IMUL	r64,m64	1	1	1			1			int	5	2
IMUL	r16,m16,i	1	1		1		1			int		2
IMUL	r32,m32,i	1	1	١.	1		1			int		1
IMUL	r64,m64,i	1	1	1			1			int	4.0	2
DIV IDIV	r8	3	3							int	18	12
DIV IDIV	r16	5	5							int	18-26	12-20 c)
DIV IDIV	r32	4	4							int	18-42	12-36 c)
DIV	r64	32	32							int	29-61	18-37 c)
IDIV	r64	56	56							int	39-72	28-40 c)
DIV IDIV	m8	4	3				1			int	18	12
DIV IDIV	m16	6	5				1			int	18-26	12-20 c)
DIV IDIV	m32	5	4				1			int	18-42	12-36 c)
DIV	m64	32	31				1			int	29-61	18-37 c)
IDIV	m64	56	55				1			int	39-72	28-40 c)
CBW CWDE CDQE		1	1	X	Х	Х				int	1	
CWD CDQ CQO		1	1	X		Х				int	1	
Logic instructions												
AND OR XOR	r,r/i	1	1			\ \				int	1	0.33
AND OR XOR	r,m	1	1	X	X	X	1			int	'	1 1
AND OR XOR		2	1	X	X	X	1	1	1	int	6	
TEST	m,r/i	1	1	X	X	X	1	ı	ı	int	1	0.33
TEST	r,r/i m,r/i	1	1	X	X	X	1			int	1	1
SHR SHL SAR	r,i/cl	1	1	X	^	X	'			int	1	0.5
SHR SHL SAR	m,i/cl	3	2	x		x	1	1	1	int	6	1
ROR ROL	r,i/cl	1	1	x		X		'	'	int	1	
ROR ROL	m,i/cl	3	2	x		X	1	1	1	int	6	
RCR RCL	r,1	2	2	X		X	'	'	'	int	2	2
RCR	r8,i/cl	9	9	X	X	X				int	12	4
RCL	r8,i/cl	8	8	X	x	X				int	11	
RCR RCL	r16/32/64,i/cl	6	6	X	X	X				int	11	
RCR RCL	m,1	4	3	X	X	X	1	1	1	int	7	
RCR	m8,i/cl	12	9	x	x	X	1	1	1	int	14	
RCL	m8,i/cl	11	8	x	x	X	1	1	1	int	13	
	1110,1/01		1	^	^	_^	'	'	' '			

RCR RCL SHLD SHRD SHLD SHRD BT BT BT BT BTR BTS BTC BTR BTS BTC BTR BTS BTC BTR BTS BTC STR BTS BTC CC SETCC CLC STC CMC CLD STD	m16/32/64,i/cl r,r,i/cl m,r,i/cl r,r/i m,r m,i r,r/i m,r r,r m,i r,r m,i r,r m,i r,m r	10 2 3 1 10 2 1 11 3 2 2 1 2 1 7 6	7 2 2 1 9 1 1 8 1 2 2 1 1 7 6	X	X	x x x x x x x x x x x x x x x x x x x	1 1 1 1 1 1	1 1 1 1	1 1 1 1 1	int int int int int int int int int int	13 2 7 1 1 5 6 2	1 5 1 1 2 1 1 0.33 4 14
Control transfer instruction JMP JMP JMP JMP JMP Conditional jump Fused compare/test and b J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL CALL CALL RETN RETN RETF RETF BOUND i) INTO i)	short/near far r m(near) m(far) short/near	1 30 1 1 31 1 1 2 11 11 3 43 3 4 44 1 3 32 32 15 5	1 30 1 1 29 1 1 2 11 11 2 43 2 3 42 1 x 30 30 13 5	x x x x	x x x x	1 1 1 1 1 1 1 x x x	1 2 1 1 2 2 2 2	1 1 1	1 1 1	int int int int int int int int int int	0 0 0 0	1-2 76 1-2 1-2 68 1 1-2 5 2 75 2 75 2 78 78 8 3
String instructions LODS REP LODS STOS REP STOS MOVS REP MOVS SCAS REP(N)E SCAS CMPS REP(N)E CMPS		3 4+7n - 4 8+5n - 8 1 7+7n - 4 7+8n - 7	2 20+1 5 1 13+n 3 17+7r 5	 2n 1 1 		 5 	1 1 1 1 1 2	 1 	 1 	int int int int int int int int int	1+5n - 2 7+2n - 0 1+3n - 0 3+8n - 2 2+7n - 2	1 0.55n 0.63n 1 23+6n 3

Other											
NOP (90)		1	1	х	Х	Х				int	0.33
Long NOP (0F 1F)		1	1	х	Х	Х				int	1
PAUSE		3	3	х	Х	Х				int	8
ENTER	i,0	12	10					1	1	int	8
ENTER	a,b									int	
LEAVE		3	2				1			int	
CPUID		46-100								int	180-215
RDTSC		29								int	64
RDPMC		23								int	54

Notes:

a) Applies to all addressing modes Has an implicit LOCK prefix. b)

Low values are for small results, high values for high results. c)

See manual 3: "The microarchitecture of Intel, AMD and VIA CPUs" for restrictions on macro-op fusion. e)

i) Not available in 64 bit mode.

Floating point x87 instructions

Instruction	Operands	μορs fused	µops	un	fuse	ed d	oma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	p5	p2	р3	p4			through- put
Move instructions												
FLD	r	1	1	1						float	1	1
FLD	m32/64	1	1				1			float	3	1
FLD	m80	4	2	2			2			float	4	3
FBLD	m80	40	38				2			float	45	20
FST(P)	r	1	1	1						float	1	1
FST(P)	m32/m64	1						1	1	float	3	1
FSTP	m80	7	3	X	X	Х		2	2	float	4	5
FBSTP	m80	170	166	X	X	Х		2	2	float	164	166
FXCH	r	1	0 f)							float	0	1
FILD	m	1	1	1			1			float	6	1
FIST	m	2	1		1			1	1	float	6	1
FISTP	m	3	1		1			1	1	float	6	1
FISTTP g)	m	3	1		1			1	1	float	6	1
FLDZ		1	1	1						float		1
FLD1		2	2	1	1					float		2
FLDPI FLDL2E etc.		2	2		2					float		2
FCMOVcc	r	2	2	2						float	2	2
FNSTSW	AX	1	1	1						float		1
FNSTSW	m16	2	1	1				1	1	float		2
FLDCW	m16	2	1				1			float		10
FNSTCW	m16	3	1					1	1	float		8
FINCSTP FDECSTP		1	1	1						float	1	1
FFREE(P)	r	2	2	2						float		2
FNSAVE	m	142								float	184	192
FRSTOR	m	78								float	169	177
Arithmetic instructions												
FADD(P) FSUB(R)(P)	r	1	1		1					float	3	1

FADD(P) FSUB(R)(P)	l m	1	1		1	1		float		1
FMUL(P)	r	1	1	1		-		float	5	2
FMUL(P)	m	1	1	1		1		float		2
FDIV(R)(P)	r	1	1	1				float	6-38 d)	5-37 d)
FDIV(R)(P)	m	1	1	1		1		float	' '	5-37 d)
FABS		1	1	1				float	1 1	1
FCHS		1	1	1				float	1 1	1
FCOM(P) FUCOM	r	1	1		1			float		1
FCOM(P) FUCOM	m	1	1		1	1		float		1
FCOMPP FUCOMPP		2	2	1	1			float		
FCOMI(P) FUCOMI(P)	r	1	1		1			float		1
FIADD FISUB(R)	m	2	2	1	1	1		float		2
FIMUL	m	2	2	2		1		float		2
FIDIV(R)	m	2	2	2		1		float		5-37 d)
FICOM(P)	m	2	2	1	1	1		float		2
FTST		1	1		1			float		1
FXAM		1	1		1			float		1
FPREM FPREM1		21-27	21-27					float	16-56	
FRNDINT		7-15	7-15					float	22-29	
Math										
FSCALE		27	27					float	41	
FXTRACT		82	82					float	170	
FSQRT		1	1					float	6-69	
FSIN FCOS		~96	~96					float	~96	
FSINCOS		~100	~100					float	~115	
F2XM1		~19	~19					float	~45	
FYL2X FYL2XP1		~53	~53					float	~96	
FPTAN		~98	~98					float	~136	
FPATAN		~70	~70					float	~119	
Other										
FNOP		1	1	1				float		1
WAIT		2	2					float		1
FNCLEX		4	4					float		15
FNINIT		15	15					float		63

Notes:

d) Round divisors or low precision give low values.

f) Resolved by register renaming. Generates no μops in the unfused domain.

g) SSE3 instruction set.

Integer MMX and XMM instructions

Instruction	Operands	µops fused	µops	un	fuse	ed d	oma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	р5	p2	р3	p4	-		through- put
Move instructions												
MOVD k)	r32/64,mm/x	1	1	Х	Х	Х				int	2	0.33
MOVD k)	m32/64,mm/x	1						1	1		3	1
MOVD k)	mm/x,r32/64	1	1	х		Х				int	2	0.5
MOVD k)	mm/x,m32/64	1					1			int	2	1
MOVQ	mm/x, mm/x	1	1	Х	X	Х				int	1	0.33
MOVQ	mm/x,m64	1					1			int	2	1

			ı			ı	1	ı	ı			
MOVQ	m64, mm/x	1						1	1		3	1
MOVDQA	xmm, xmm	1	1	X	Х	Х				int	1	0.33
MOVDQA	xmm, m128	1					1			int	2	1
MOVDQA	m128, xmm	1						1	1		3	1
MOVDQU	m128, xmm	9	4	X	Х	Х	1	2	2		3-8	4
MOVDQU	xmm, m128	4	2	X		Х	2			int	2-8	2
LDDQU g)	xmm, m128	4	2	x		Х	2			int	2-8	2
MOVDQ2Q	mm, xmm	1	1	x	х	Х				int	1	0.33
MOVQ2DQ	xmm,mm	1	1	X	х	Х				int	1	0.33
MOVNTQ	m64,mm	1						1	1			2
MOVNTDQ	m128,xmm	1						1	1			2
PACKSSWB/DW	mm,mm	1	1	1						int	1	1
PACKUSWB	mm,m64	1	1	1			1			int		1
PACKSSWB/DW	xmm,xmm	3	3							flt→int	3	2
PACKUSWB	xmm,m128	4	3				1			int		2
PUNPCKH/LBW/WD/DQ	mm,mm	1	1	1						int	1	1
PUNPCKH/LBW/WD/DQ	mm,m64	1	1	1			1			int		1
PUNPCKH/LBW/WD/DQ	xmm,xmm	3	3							flt→int	3	2
PUNPCKH/LBW/WD/DQ	xmm,m128	4	3				1			int		2
PUNPCKH/LQDQ	xmm,xmm	1	1							int	1	1
PUNPCKH/LQDQ	xmm, m128	2	1				1			int		1
PSHUFB h)	mm,mm	1	1			1				int	1	1
PSHUFB h)	mm,m64	2	1			1	1			int		1
PSHUFB h)	xmm,xmm	4	4							int	3	2
PSHUFB h)	xmm,m128	5	4				1			int		2
PSHUFW	mm,mm,i	1	1			1	-			int	1	1
PSHUFW	mm,m64,i	2	1			1	1			int	-	1
PSHUFD	xmm,xmm,i	2	2	x	x	1				flt→int	3	1
PSHUFD	xmm,m128,i	3	2	X	X	1	1			int		1
PSHUFL/HW	xmm,xmm,i	1	1	^	^	1	ľ			int	1	1
PSHUFL/HW	xmm, m128,i	2	1			1	1			int	•	1
PALIGNR h)	mm,mm,i	2	2	x	x	Х				int	2	1
PALIGNR h)	mm,m64,i	2	2	X	X	X	1			int	_	1
PALIGNR h)	xmm,xmm,i	2	2	x	x	X	'			int	2	1
PALIGNR h)	xmm,m128,i	2	2	x	x	x	1			int	2	1
MASKMOVQ	mm,mm	4		^	^	^				int		2-5
MASKMOVDQU	xmm,xmm	10								int		6-10
PMOVMSKB	r32,mm/x	10	1	1						int	2	
PEXTRW	r32,mm,i	2	2	'						int	2 3	1 1
PEXTRW		3	3									
	r32,xmm,i					4				int	5	1
PINSRW	mm,r32,i	1	1			1	4			int	2	1
PINSRW	mm,m16,i	2	1	l		1	1			int	0	1
PINSRW	xmm,r32,i	3	3	X	X	X	4			int	6	1.5
PINSRW	xmm,m16,i	4	3	Х	Х	Х	1			int		1.5
A!41												
Arithmetic instructions	· ,										,	0.5
PADD/SUB(U)(S)B/W/D	mm/x, mm/x	1	1	X		X				int	1	0.5
PADD/SUB(U)(S)B/W/D	mm/x,m	1	1	Х		Х	1			int	_	1
PADDQ PSUBQ	mm/x, mm/x	2	2	Х		Х				int	2	1
PADDQ PSUBQ	mm/x,m	2	2	Х		Х	1			int		1
PHADD(S)W	marea 105 105	F	E							:4	F	4
PHSUB(S)W h)	mm,mm	5	5							int	5	4

1	1									ı	ı
PHADD(S)W	0.4		_						. ,		_
PHSUB(S)W h)	mm,m64	6	5				1		int		4
PHADD(S)W	\(\(\text{ine ine } \)\(\text{ine ine } \)	7	_						:4	6	4
PHSUB(S)W h)	xmm,xmm	7	7						int	6	4
PHADD(S)W	vmm m100	0	7				4		int		4
PHSUB(S)W h)	xmm,m128	8					1		int	_	4
PHADDD PHSUBD h)	mm,mm	3	3						int	3	2
PHADDD PHSUBD h)	mm,m64	4	3				1		int	_	2
PHADDD PHSUBD h)	xmm,xmm	5	5						int	5	3
PHADDD PHSUBD h)	xmm,m128	6	5				1		int		3
PCMPEQ/GTB/W/D	mm/x,mm/x	1	1	X		Х			int	1	0.5
PCMPEQ/GTB/W/D	mm/x,m	1	1	X		Х	1		int		1
PMULL/HW PMULHUW	mm/x,mm/x	1	1		1				int	3	1
PMULL/HW PMULHUW	mm/x,m	1	1		1		1		int		1
PMULHRSW h)	mm/x,mm/x	1	1		1				int	3	1
PMULHRSW h)	mm/x,m	1	1		1		1		int		1
PMULUDQ	mm/x,mm/x	1	1		1				int	3	1
PMULUDQ	mm/x,m	1	1		1		1		int		1
PMADDWD	mm/x,mm/x	1	1		1				int	3	1
PMADDWD	mm/x,m	1	1		1		1		int		1
PMADDUBSW h)	mm/x,mm/x	1	1		1				int	3	1
PMADDUBSW h)	mm/x,m	1	1		1		1		int		1
PAVGB/W	mm/x,mm/x	1	1	x		Х			int	1	0.5
PAVGB/W	mm/x,m	1	1	x		Х	1		int		1
PMIN/MAXUB/SW	mm/x,mm/x	1	1	х		х			int	1	0.5
PMIN/MAXUB/SW	mm/x,m	1	1	х		х	1		int		1
PABSB PABSW PABSD	mm/x,mm/x	1	1	x		х			int	1	0.5
h)	mm/x,m	1	1	x		х	1		int		1
PSIGNB PSIGNW	mm/x,mm/x	1	1	x		х			int	1	0.5
PSIGND h)	mm/x,m	1	1	х		х	1		int		1
PSADBW	mm/x,mm/x	1	1		1				int	3	1
PSADBW	mm/x,m	1	1		1		1		int		1
Logic instructions											
PAND(N) POR PXOR	mm/x,mm/x	1	1	х	х	х			int	1	0.33
PAND(N) POR PXOR	mm/x,m	1	1	x	х	х	1		int		1
PSLL/RL/RAW/D/Q	mm,mm/i	1	1	1					int	1	1
PSLL/RL/RAW/D/Q	mm,m64	1	1	1			1		int		1
PSLL/RL/RAW/D/Q	xmm,i	1	1	1					int	1	1
PSLL/RL/RAW/D/Q	xmm,xmm	2	2	х	х				int	2	1
PSLL/RL/RAW/D/Q	xmm,m128	3	2	X	X		1		int		1
PSLL/RLDQ	xmm,i	2	2	X	X				int	2	1
	,										
Other											
EMMS		11	11	х	х	х			float		6
<u> </u>											

Notes:

g) h) SSE3 instruction set.

Supplementary SSE3 instruction set.

MASM uses the name MOVD rather than MOVQ for this instruction even when

k) moving 64 bits.

Floating point XMM instructions

Instruction	Operands	µops fused	μops	un	fuse	ed d	oma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	p5	p2	р3	p4	-	y	through
Move instructions												
MOVAPS/D	xmm,xmm	1	1	Х	Х	Х				int	1	0.33
MOVAPS/D	xmm,m128	1					1			int	2	1
MOVAPS/D	m128,xmm	1						1	1		3	1
MOVUPS/D	xmm,m128	4	2	1		1	2			int	2-4	2
MOVUPS/D	m128,xmm	9	4	X	X	X	1	2	2		3-4	4
MOVSS/D	xmm,xmm	1	1	X	X	X				int	1	0.33
MOVSS/D	xmm,m32/64	1 -	'	^	^	^	1			int	2	1
MOVSS/D	m32/64,xmm	1					'	1	1		3	1
MOVHPS/D MOVLPS/D	xmm,m64	2	1			1	1		'	int	3	1
MOVHPS/D	m64,xmm	2	1	1		ļ '	'	1	1		5	1
MOVLPS/D	m64,xmm	1	'	'				1	1		3	1
MOVLHPS MOVHLPS	xmm,xmm	1	1	1				'	'	float	1	1
MOVMSKPS/D	r32,xmm	1 1	'1	1						float	1	1 1
MOVNTPS/D	m128,xmm	1	'	'				1	1	lioat	'	2-3
SHUFPS	xmm,xmm,i	3	3		3			'	'	flt→int	3	2-3
SHUFPS	xmm,m128,i	4	3		3		1			flt→int	3	2
SHUFPD			1	1	3		'			float	1	1
	xmm,xmm,i	1		1 -			1				1	
SHUFPD	xmm,m128,i	2	1	1			'			float		1
MOVDDUP g)	xmm,xmm	1	1	1			_			int	1	1
MOVDDUP g)	xmm,m64	2	1	1			1			int		1
MOVSH/LDUP g)	xmm,xmm	1	1			1	١,			int	1	1
MOVSH/LDUP g)	xmm,m128	2	1			1	1			int		1
UNPCKH/LPS	xmm,xmm	3	3		3		١.			flt→int	3	2
UNPCKH/LPS	xmm,m128	4	3		3		1			int		2
UNPCKH/LPD	xmm,xmm	1	1	1						float	1	1
UNPCKH/LPD	xmm,m128	2	1	1			1			float		1
Conversion												
CVTPD2PS	xmm,xmm	2	2							float	4	1
CVTPD2PS	xmm,m128	2	2				1			float		1
CVTSD2SS	xmm,xmm	2	2							float	4	1
CVTSD2SS	xmm,m64	2	2				1			float		1
CVTPS2PD	xmm,xmm	2	2	2						float	2	2
CVTPS2PD	xmm,m64	2	2	2			1			float		2
CVTSS2SD	xmm,xmm	2	2							float	2	2
CVTSS2SD	xmm,m32	2	2	2			1			float		2
CVTDQ2PS	xmm,xmm	1	1		1					float	3	1
CVTDQ2PS	xmm,m128	1	1		1		1			float		1
CVT(T) PS2DQ	xmm,xmm	1	1		1					float	3	1
CVT(T) PS2DQ	xmm,m128	1	1		1		1			float		1
CVTDQ2PD	xmm,xmm	2	2	1	1		•			float	4	1
CVTDQ2PD	xmm,m64	3	2	'	'		1			float		1
CVT(T)PD2DQ	xmm,xmm	2	2				'			float	4	1
CVT(T)PD2DQ CVT(T)PD2DQ	xmm,m128	2	2				1			float		1
CVT(1)i b2bQ CVTPl2PS	xmm,mm	1	1		1		'			float	3	3
CVTPI2PS	xmm,m64	1	1		1		1			float		3
CVTP12P3 CVT(T)PS2PI	mm,xmm	1			1		'			float	3	1
CVT(T)PS2PI CVT(T)PS2PI	mm,m128	1	1		1		1			float	ا	1
OVI(1)F32F1	11111,111120	1	1	1	1	1	'	I		livat	1	1

CVTPI2PD	xmm,mm	2	2	1	1				float	4	1
CVTPI2PD	xmm,m64	2	2	1	1		1		float		1
CVT(T) PD2PI	mm,xmm	2	2	1	1				float	4	1
CVT(T) PD2PI	mm,m128	2	2	1	1	.	1		float		1
CVTSI2SS	xmm,r32	1	1		1				float	4	3
CVTSI2SS	xmm,m32	1	1		1		1		float		3
CVT(T)SS2SI	r32,xmm	1	1		1				float	3	1
CVT(T)SS2SI	r32,m32	1	1		1	.	1		float		1
CVTSI2SD	xmm,r32	2	2	1	1				float	4	3
CVTSI2SD	xmm,m32	2	1		1	.	1		float		3
CVT(T)SD2SI	r32,xmm	1	1		1		-		float	3	1
CVT(T)SD2SI	r32,m64	1	1		1		1		float		1
011(1)00201	102,11101	•	'		١.		•		noat		•
Arithmetic											
ADDSS/D SUBSS/D	xmm,xmm	1	1		1				float	3	1
ADDSS/D SUBSS/D	xmm,m32/64	1	1		1	.	1		float	3	1
		=					1				
ADDPS/D SUBPS/D	xmm,xmm	1	1		1				float	3	1
ADDPS/D SUBPS/D	xmm,m128	1	1		1		1		float		1
ADDSUBPS/D g)	xmm,xmm	1	1		1				float	3	1
ADDSUBPS/D g)	xmm,m128	1	1		1	'	1		float		1
HADDPS HSUBPS g)	xmm,xmm	6	6						float	9	3
HADDPS HSUBPS g)	xmm,m128	7	6			'	1		float		3
HADDPD HSUBPD g)	xmm,xmm	3	3						float	5	2
HADDPD HSUBPD g)	xmm,m128	4	3				1		float		2
MULSS	xmm,xmm	1	1	1					float	4	1
MULSS	xmm,m32	1	1	1			1		float		1
MULSD	xmm,xmm	1	1	1					float	5	1
MULSD	xmm,m64	1	1	1		.	1		float		1
MULPS	xmm,xmm	1	1	1					float	4	1
MULPS	xmm,m128	1	1	1		.	1		float		1
MULPD	xmm,xmm	1	1	1					float	5	1
MULPD	xmm,m128	1	1	1		.	1		float		1
DIVSS	xmm,xmm	1	1	1					float	6-18 d)	5-17 d)
DIVSS	xmm,m32	1	1	1		.	1		float		5-17 d)
DIVSD	xmm,xmm	1	1	1			•		float	6-32 d)	
DIVSD	xmm,m64	1	1	1			1		float	0 02 4)	5-31 d)
DIVPS	xmm,xmm	1	1	1			'		float	6-18 d)	5-31 d) 5-17 d)
DIVPS	xmm,m128	1		l .		l .	1			0-10 u)	′
	· '	=	1	1			1		float	6 20 4/	5-17 d)
DIVPD	xmm,xmm	1	1	1					float	6-32 d)	5-31 d)
DIVPD	xmm,m128	1	1	1			1		float		5-31 d)
RCPSS/PS	xmm,xmm	1	1		1				float	3	2
RCPSS/PS	xmm,m	1	1		1	'	1		float		2
CMPccSS/D	xmm,xmm	1	1		1				float	3	1
CMPccSS/D	xmm,m32/64	1	1		1	'	1		float		1
CMPccPS/D	xmm,xmm	1	1		1				float	3	1
CMPccPS/D	xmm,m128	1	1		1	'	1		float		1
COMISS/D UCOMISS/D	xmm,xmm	1	1		1				float	3	1
COMISS/D UCOMISS/D	xmm,m32/64	1	1		1	.	1		float		1
MAXSS/D MINSS/D	xmm,xmm	1	1		1				float	3	1
MAXSS/D MINSS/D	xmm,m32/64	1	1		1	.	1		float		1
MAXPS/D MINPS/D	xmm,xmm	1	1		1				float	3	1
MAXPS/D MINPS/D	xmm,m128	1	1		1	-	1		float		1
							*	,			

Math													
SQRTSS/PS	xmm,xmm	1	1	1						float	6-29	6-29	
SQRTSS/PS	xmm,m	2	1	1			1			float		6-29	
SQRTSD/PD	xmm,xmm	1	1	1						float	6-58	6-58	
SQRTSD/PD	xmm,m	2	1	1			1			float		6-58	
RSQRTSS/PS	xmm,xmm	1	1		1					float	3	2	
RSQRTSS/PS	xmm,m	1	1		1		1			float		2	
Logic													
AND/ANDN/OR/XORPS/D	xmm,xmm	1	1	х	Х	Х				int	1	0.33	
AND/ANDN/OR/XORPS/D	xmm,m128	1	1	х	х	х	1			int		1	
Other													
LDMXCSR	m32	14	13				1					42	
STMXCSR	m32	6	4					1	1			19	
FXSAVE	m4096	141									145	145	
FXRSTOR	m4096	119									164	164	

Notes:

d) Round divisors give low values.

g) SSE3 instruction set.

Intel Core 2 (Wolfdale, 45nm)

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, xmm = 128 bit

xmm register, mm/x = mmx or xmm register, sr = segment register, m = mem-

ory, m32 = 32-bit memory operand, etc.

μορs fused domain: The number of μορs at the decode, rename, allocate and retirement stages in

the pipeline. Fused µops count as one.

μops unfused domain: The number of μops for each execution port. Fused μops count as two. Fused

macro-ops count as one. The instruction has μ op fusion if the sum of the numbers listed under p015 + p2 + p3 + p4 exceeds the number listed under μ ops fused domain. An x under p0, p1 or p5 means that at least one of the μ ops listed under p015 can optionally go to this port. For example, a 1 under p015 and an x under p0 and p5 means one μ op which can go to either port 0 or port 5, whichever is vacant first. A value listed under p015 but nothing under p0, p1 and p5 means that it is not known which of the three ports these

µops go to.

p015: The total number of μops going to port 0, 1 and 5.
p0: The number of μops going to port 0 (execution units).
p1: The number of μops going to port 1 (execution units).
p5: The number of μops going to port 5 (execution units).
p2: The number of μops going to port 2 (memory read).

p3: The number of μops going to port 3 (memory write address).p4: The number of μops going to port 4 (memory write data).

Unit: Tells which execution unit cluster is used. An additional delay of 1 clock cycle

is generated if a register written by a μop in the integer unit (int) is read by a μop in the floating point unit (float) or vice versa. flt—int means that an instruction with multiple μops receive the input in the float unit and delivers the output in the int unit. Delays for moving data between different units are included under latency when they are unavoidable. For example, movd eax,xmm0 has an extra 1 clock delay for moving from the XMM-integer unit to the general purpose integer unit. This is included under latency because it occurs regardless of which instruction comes next. Nothing listed under unit means that additional delays are either unlikely to occur or unavoidable and therefore in-

cluded in the latency figure.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give a similar delay. The time unit used is core clock cycles, not the reference clock cycles

given by the time stamp counter.

Reciprocal throughput: The average number of core clock cycles per instruction for a series of inde-

pendent instructions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μοps fused	μορs unfused domain							Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	p5	p2	р3	p4			through- put
Move instructions												
MOV	r,r/i	1	1	Х	Х	х					1	0.33
MOV a)	r,m	1					1				2	1

MOV a)	m,r	1						1	1	3	1
MOV	m,i	1						1	1	3	1
MOV	r,sr	1					1				1
MOV	m,sr	2					1	1	1		1
MOV	sr,r	8	4	x	x	X	4	-			16
MOV	sr,m	8	3	X	^	X	5				16
MOVNTI	m,r	2	"	^		^		1	1		2
	111,1							'	'		
MOVSX MOVZX MOVSXD	r,r	1	1	X	X	x				1	0.33
MOVSX MOVZX	r16/32,m	1	'	^	^	^	1			1	
							-				1
MOVSX MOVSXD	r64,m	2	1	X	Х	X	1				1
CMOVcc	r,r	2	2	Х	X	X	١.			2	1
CMOVcc	r,m	2	2	X	X	X	1			_	_
XCHG	r,r	3	3	X	X	X				2	2
XCHG	r,m	7	Х				1	1	1	high b)	
XLAT		2	1				1			4	1
PUSH	r	1						1	1	3	1
PUSH	i	1						1	1		1
PUSH	m	2					1	1	1		1
PUSH	sr	2	1					1	1		1
PUSHF(D/Q)		17	15	x	x	x		1	1		7
PUSHA(D) i)		18	9					1	8		8
POP	r	1					1			2	1
POP	(E/R)SP	4	3				1			_	·
POP	m	2	"				1	1	1		1.5
POP	sr	10	9				1	'	'		1.5
POPF(D/Q)	51	24	23		.,		1			20	17
			1	X	X	X	1 -			20	7
POPA(D) i)		10	2				8			_	7
LAHF SAHF		1	1	Х	Х	X				1	0.33
SALC i)		2	2	X	X	Х				4	1
LEA a)	r,m	1	1	1						1	1
BSWAP	r	2	2	1		1				4	1
LDS LES LFS LGS LSS	m	11	11				1				17
PREFETCHNTA	m	1					1				1
PREFETCHT0/1/2	m	1					1				1
LFENCE		2						1	1		8
MFENCE		2						1	1		6
SFENCE		2						1	1		9
CLFLUSH	m8	4	2	1		1		1	1	120	90
IN											
OUT											
Arithmetic instructions											
ADD SUB	r,r/i	1	1	X	x	x				1	0.33
ADD SUB	r,m	1	1	x	x	x	1				1
ADD SUB	m,r/i	2	1	x	х	x	1	1	1	6	1
ADC SBB	r,r/i	2	2	X	X	X		-		2	2
ADC SBB	r,m	2	2	X	X	X	1			2	2
ADC SBB	m,r/i	4	3	x	X	x	1	1	1	7	_
CMP	r,r/i	1	1	x	x	x	'	'	'	1	0.33
CMP	m,r/i	1		x	x	x	1			1	1
INC DEC NEG NOT		1	1				'			1	0.33
INC DEC NEG NOT	r	3	1	X	X	X	1	1	1	6	1
HING DECINED INOT	m	ر	1	X	X	X	'	'	1	0	

AAA AAS DAA DAS i)		1	1		1							1
AAD i)		3	3	Х	Х	х						1
AAM i)		5	5	х	х	х					17	
MUL IMUL	r8	1	1		1						3	1
MUL IMUL	r16	3	3	Х	х	x					5	1.5
MUL IMUL	r32	3	3	х	х	x					5	1.5
MUL IMUL	r64	3	3	х	х	x					7	4
IMUL	r16,r16	1	1	^`	1	^					3	1
IMUL	r32,r32	1	1		1						3	1
IMUL	r64,r64	1	1	1	'						5	2
IMUL	r16-,16-,i	1	1	'	1						3	1
IMUL	r32,r32,i	1	1 1		1						3	1
IMUL	r64,r64,i	1	1	1	'						5	2
MUL IMUL	m8	1		'	4		4				3	1
			1		1	l	1				I I	
MUL IMUL	m16	3	3	Х	Х	Х	1				5	1.5
MUL IMUL	m32	3	3	X	Х	Х	1				5	1.5
MUL IMUL	m64	3	2	2			1				7	4
IMUL	r16,m16	1	1		1		1				3	1
IMUL	r32,m32	1	1		1		1				3	1
IMUL	r64,m64	1	1	1			1				5	2
IMUL	r16,m16,i	1	1		1		1					2
IMUL	r32,m32,i	1	1		1		1					1
IMUL	r64,m64,i	1	1	1			1					2
DIV IDIV	r8	4	4	1	2	1					9-18 c)	
DIV IDIV	r16	7	7	х	х	x					14-22 c)	
DIV IDIV	r32	7	7	2	3	2					14-23 c)	
DIV	r64	32-38	32-38	9	10	13					18-57 c)	
IDIV	r64	56-62	56-62	X	X	X					34-88 c)	
DIV IDIV	m8	4	3	1	2	^	1				9-18	
DIV IDIV	m16	7	7	2	3	2	1				14-22 c)	
DIV IDIV	m32	7	6	X	x	X	1				14-23 c)	
DIV	m64	32	31	X	x	x	1				34-88 c)	
IDIV	m64	56	55				1				1 /1	
	11104			X	X	X					39-72 c)	
CBW CWDE CDQE		1	1	X	Х	X					1	
CWD CDQ CQO		1	1	X		X					1	
Logic instructions	/:	4	4	.,	.,						4	0.00
AND OR XOR	r,r/i	1	1	Х	Х	Х					1	0.33
AND OR XOR	r,m	1	1	Х	Х	X	1					1
AND OR XOR	m,r/i	2	1	Х	Х	X	1	1	1		6	1
TEST	r,r/i	1	1	Х	Х	X					1	0.33
TEST	m,r/i	1	1	Х	Х	X	1					1
SHR SHL SAR	r,i/cl	1	1	Х		X					1	0.5
SHR SHL SAR	m,i/cl	3	2	Х		X	1	1	1		6	1
ROR ROL	r,i/cl	1	1	Х		Х					1	1
ROR ROL	m,i/cl	3	2	х		х	1	1	1		6	1
RCR RCL	r,1	2	2	х	х	х					2	2
RCR	r8,i/cl	9	9	х	х	х					12	
RCL	r8,i/cl	8	8	х	х	х					11	
RCR RCL	r,i/cl	6	6	х	х	x					11	
RCR RCL	m,1	4	3	X	X	X	1	1	1		7	
RCR	m8,i/cl	12	9	X	X	X	1	1	1		14	
RCL	m8,i/cl	11	8	X	X	X	1	1	1		13	
ı - -	, ,,,		1	٠.,		1			1 -	I		

RCR RCL	m,i/cl	10	7	x	x	x	1	1	1	13	
SHLD SHRD	r,r,i/cl	2	2	x	x	x				2	1
SHLD SHRD	m,r,i/cl	3	2	x	x	x	1	1	1	7	
BT	r,r/i	1	1	X	X	X		•	-	1	1
BT	m,r	9	8	X	X	X	1			•	4
BT	m,i	3	2				1				1
				X	X	X	'				' '
BTR BTS BTC	r,r/i	1	1 -	Х	X	X	١.,			1 1	
BTR BTS BTC	m,r	10	7	X	Х	X	1	1	1	5	
BTR BTS BTC	m,i	3	1	Х	X	X	1	1	1	6	
BSF BSR	r,r	2	2	X	1	X				2	1
BSF BSR	r,m	2	2	X	1	Х	1				1
SETcc	r	1	1	X	x	Х				1	1
SETcc	m	2	1	x	x	x		1	1		1
CLC STC CMC		1	1	x	x	x				1	0.33
CLD		6	6	Х	X	x					3
STD		6	6	X	X	X					14
015					^	^					17
Control transfer instructi	one										
JMP	short/near	1	1			1				0	1-2
						'					
JMP i)	far	30	30								76
JMP	r	1	1			1	١.			0	1-2
JMP	m(near)	1	1			1	1			0	1-2
JMP	m(far)	31	29				2				68
Conditional jump	short/near	1	1			1				0	1
Fused compare/test and bi	ranch e,i)	1	1			1				0	1
J(E/R)CXZ	short	2	2	х	x	1					1-2
LOOP	short	11	11	x	x	x					5
LOOP(N)E	short	11	11	Х	X	X					5
CALL	near	3	2	X	X	X		1	1		2
CALL i)	far	43	43	^	^	^		•	'		75
CALL		3	2					1	4		2
1	r	1	ı						1		
CALL	m(near)	4	3				1	1	1		2
CALL	m(far)	44	42			١.	2				75
RETN		1	1			1	1				2
RETN	i	3	1			1	1				2
RETF		32	30				2				78
RETF	i	32	30				2				78
BOUND i)	r,m	15	13				2				8
INTO i)		5	5								3
,											
String instructions											
LODS		3	2				1				1
REP LODS		4+7n-1	I	I	ı	I	1 -	l	l	1+5n-	21+3n
STOS		4	2				I	1	1	1.011	1 1
REP STOS		8+5n-2	I	l Dn	l	l	I	'	' '	7±2n	0.55n
				-11 	ı	I	ı	I	I	7 +211-	0.5511
MOVS		8	5	,		_					
DED MOVO		1	1	1		5					
REP MOVS		7+7n-1		I	I	ı		I	ı	1+3n-	0.63n
SCAS		4	3				1				1
REP(N)E SCAS		7+8n-1							1	3+8n-	23+6n
CMPS		7	5				2				3
REP(N)E CMPS		7+10n-	7+9n							2+7n-	22+5n

Other												ı
NOP (90)		1	1	Х	Х	х					0.33	ı
Long NOP (0F 1F)		1	1	Х	Х	х					1	l
PAUSE		3	3	Х	Х	х					8	l
ENTER	i,0	12	10					1	1		8	l
ENTER	a,b											l
LEAVE		3	2				1					l
CPUID		53-117									53-211	l
RDTSC		13									32	l
RDPMC		23									54	l

Notes:

a) Applies to all addressing modes Has an implicit LOCK prefix. b)

Low values are for small results, high values for high results. The reciprocal c)

throughput is only slightly less than the latency.

See manual 3: "The microarchitecture of Intel, AMD and VIA CPUs" for restrictions on macro-op fusion. e)

i) Not available in 64 bit mode.

Floating point x87 instructions

Instruction	Operands	µops fused	µops	un	fuse	ed d	oma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	р5	p2	р3	p4			through- put
Move instructions												
FLD	r	1	1	1						float	1	1
FLD	m32/64	1	1				1			float	3	1
FLD	m80	4	2	2			2			float	4	3
FBLD	m80	40	38	Х	Х	Х	2			float	45	20
FST(P)	r	1	1	1						float	1	1
FST(P)	m32/m64	1						1	1	float	3	1
FSTP	m80	7	3	Х	Х	Х		2	2	float	4	5
FBSTP	m80	171	167	Х	Х	Х		2	2	float	164	166
FXCH	r	1	0 f)							float	0	1
FILD	m	1	1		1		1			float	6	1
FIST	m	2	1		1			1	1	float	6	1
FISTP	m	3	1		1			1	1	float	6	1
FISTTP g)	m	3	1		1			1	1	float	6	1
FLDZ		1	1	1						float		1
FLD1		2	2	1	1					float		2
FLDPI FLDL2E etc.		2	2		2					float		2
FCMOVcc	r	2	2	2						float	2	2
FNSTSW	AX	1	1	1						float		1
FNSTSW	m16	2	1	1				1	1	float		2
FLDCW	m16	2	1				1			float		10
FNSTCW	m16	3	1			1		1	1	float		8
FINCSTP FDECSTP		1	1	1						float	1	1
FFREE(P)	r	2	2	Х	х	Х				float		2
FNSAVE	m	141	95	Х	Х	Х	7	23	23	float		142
FRSTOR	m	78	51	x	x	x	27			float		177
Arithmetic instructions												

FADD(P) FSUB(R)(P)	r	1	1		1				float	3	1
FADD(P) FSUB(R)(P)	m	1	1		1		1		float		1
FMUL(P)	r	1	1	1					float	5	2
FMUL(P)	m	1	1	1			1		float		2
FDIV(R)(P)	r	1	1	1					float	6-21 d)	5-20 d)
FDIV(R)(P)	m	1	1	1			1		float	6-21 d)	5-20 d)
FABS		1	1	1			•		float	1	1
FCHS		1	1	1					float	1	1
FCOM(P) FUCOM	r	1 1	1	'	1				float		1
FCOM(P) FUCOM	m .	1	1		1		1		float		1
FCOMPP FUCOMPP		2	2	1	1		'		float		•
FCOMI(P) FUCOMI(P)	r	1	1	'	1				float		1
FIADD FISUB(R)	m	2	2		2		1		float	3	2
FIMUL	m	2	2	1	1		1		float	5	2
FIDIV(R)	m	2	2	1	1		1		float	6-21	5-20 d)
FICOM(P)	m	2	2	'	2				float	0-21	2 2 a)
FTST	'''	1	1		1		'		float		1
FXAM		1 1			1				float		1
FPREM		26-29	' '	\ ,		\ <u>,</u>			float	13-40	1
FPREM1		28-35		X	X	X			float	18-41	
FRNDINT		17-19		X	X	X			float	10-41	
FRINDINI		17-19	I	Х	Х	Х			lloat	10-22	
Math											
FSCALE		28	28	Х	Х	х			float	43	
FXTRACT		53-84		Х	Х	x			float	~170	
FSQRT		1	1	1					float	6-20	
FSIN		18-85		Х	Х	x			float	32-85	
FCOS		76-100		х	Х	х			float	70-100	
		18-									
FSINCOS		105		Х	Х	х			float	38-107	
F2XM1		19	19	х	Х	х			float	45	
FYL2X FYL2XP1		57-65		х	Х	x			float	50-100	
FPTAN		19-100		x	Х	х			float	40-130	
FPATAN		23-87		x	Х	х			float	55-130	
Other											
FNOP		1	1	1					float		1
WAIT		2	2	Х	х	Х			float		1
FNCLEX		4	4		х	Х			float		15
FNINIT		15	15	х	х	х			float		63

Notes:

d) Round divisors or low precision give low values.

f) Resolved by register renaming. Generates no μops in the unfused domain.

g) SSE3 instruction set.

Integer MMX and XMM instructions

Instruction	Operands	μοps fused	μops	un	fuse	ed d	loma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	р5	p2	р3	p4			through- put
Move instructions	structions											
MOVD k)	r,mm/x	1	1	Х	Х	X				int	2	0.33

MOVE	I	4	I	ı	ı	I	I	ا ا	 a			1 4	ĺ
MOVD k)	m,mm/x	1						1	1	:4	3	1	
MOVD k)	mm/x,r	1	1	X		X	,			int	2	0.5	
MOVD k)	mm/x,m	1					1			int	2	1	
MOVQ	V,V	1	1	X	X	X	١,			int	1	0.33	
MOVQ	mm/x,m64	1					1	١,		int	2	1	
MOVQ	m64, mm/x	1						1	1		3	1	
MOVDQA	xmm, xmm	1	1	X	X	Х				int	1	0.33	
MOVDQA	xmm, m128	1					1	١.		int	2	1	
MOVDQA	m128, xmm	1						1	1		3	1	
MOVDQU	m128, xmm	9	4	X	Х	Х	1	2	2		3-8	4	
MOVDQU	xmm, m128	4	2	X		X	2			int	2-8	2	
LDDQU g)	xmm, m128	4	2	X		Х	2			int	2-8	2	
MOVDQ2Q	mm, xmm	1	1	X	Х	Х				int	1	0.33	
MOVQ2DQ	xmm,mm	1	1	X	Х	Х				int	1	0.33	
MOVNTQ	m64,mm	1						1	1			2	
MOVNTDQ	m128,xmm	1						1	1			2	
MOVNTDQA j)	xmm, m128	1					1				2	1	
PACKSSWB/DW													
PACKUSWB	mm,mm	1	1	1						int	1	1	
PACKSSWB/DW				١.									
PACKUSWB	mm,m64	1	1	1			1			int		1	
PACKSSWB/DW		4	_			,				4			
PACKUSWB	xmm,xmm	1	1			1				int	1	1	
PACKSSWB/DW	V/2000 100 100	4	_			4	4			:4		4	
PACKUSWB	xmm,m128	1	1			1	1			int	,	1	
PACKUSDW j)	xmm,xmm	1	1			1	,			int	1	1	
PACKUSDW j)	xmm,m	1	1	1		1	1			int	,	1	
PUNPCKH/LBW/WD/DQ	mm,mm	1	1	1			,			int	1	1	
PUNPCKH/LBW/WD/DQ	mm,m64	1	1	1		,	1			int		1	
PUNPCKH/LBW/WD/DQ	xmm,xmm	1	1			1	,			int	1	1	
PUNPCKH/LBW/WD/DQ	xmm,m128	1	1			1	1			int		1	
PUNPCKH/LQDQ	xmm,xmm	1	1			1				int	1	1	
PUNPCKH/LQDQ	xmm, m128	2	1			1	1			int		1	
PMOVSX/ZXBW j)	xmm,xmm	1	1			1				int	1	1	
PMOVSX/ZXBW j)	xmm,m64	1	1			1	1			int		1	
PMOVSX/ZXBD j)	xmm,xmm	1	1			1				int	1	1	
PMOVSX/ZXBD j)	xmm,m32	1	1			1	1			int		1	
PMOVSX/ZXBQ j)	xmm,xmm	1	1			1				int	1	1	
PMOVSX/ZXBQ j)	xmm,m16	1	1			1	1			int		1	
PMOVSX/ZXWD j)	xmm,xmm	1	1			1				int	1	1	
PMOVSX/ZXWD j)	xmm,m64	1	1			1	1			int		1	
PMOVSX/ZXWQ j)	xmm,xmm	1	1			1				int	1	1	
PMOVSX/ZXWQ j)	xmm,m32	1	1			1	1			int		1	
PMOVSX/ZXDQ j)	xmm,xmm	1	1			1				int	1	1	
PMOVSX/ZXDQ j)	xmm,m64	1	1			1	1			int		1	
PSHUFB h)	mm,mm	1	1			1				int	1	1	
PSHUFB h)	mm,m64	2	1			1	1			int		1	
PSHUFB h)	xmm,xmm	1	1			1				int	1	1	
PSHUFB h)	xmm,m128	1	1			1	1			int		1	
PSHUFW	mm,mm,i	1	1			1				int	1	1	
PSHUFW	mm,m64,i	2	1			1	1			int		1	
PSHUFD	xmm,xmm,i	1	1			1				int	1	1	
PSHUFD	xmm,m128,i	2	1			1	1			int		1	
PSHUFL/HW	xmm,xmm,i	1	1			1				int	1	1	

PSHUFUFUM	1											ı	1 1
PALIGNR h)	PSHUFL/HW	x, m128,i	2				1	1			int		· ·
PALIGNR h)	,	mm,mm,i		2							int	2	1
PALLENDVB)	PALIGNR h)	mm,m64,i	3	3			3	1			int		1
PBLENDVB	PALIGNR h)	xmm,xmm,i	1	1			1				int	1	1
PBLENDW	PALIGNR h)	xmm,m128,i	1	1			1	1			int		1
PBLENDW	PBLENDVB j)	x,x,xmm0	2	2			2				int	2	2
PBLENDW	PBLENDVB j)	x,m,xmm0	2	2			2	1			int		2
MASKMOVQ	PBLENDW j)	xmm,xmm,i	1	1			1				int	1	1 1
MASKMOVDQU	PBLENDW j)	xmm,m,i	1	1			1	1			int		1 1
MASKMOVDQU	MASKMOVQ	mm,mm	4	1	1			1	1	1	int		2-5
PMOVMSKB	MASKMOVDQU	1 1	10	4	1		3	2	2	3	int		6-10
PEXTRB	I	1 '	1	1	1							2	1 1
PEXTRB	I	1 '	2	2	x	x	x						1 1
PEXTRW PEXTRW PEXTRD m16,mm/x,i 2 2 2 x x x 1	27	1		2	x								
PEXTRW j)	27	1 1						1					
PEXTRD j)	I	1 ' 1				1			1	1			
PEXTRD	3,	1 ' ' 1			-		-					3	
PEXTRQ j,m)		1 1			^	^`			1	1			
PEXTRQ j,m)		1 1		1	_	v	-		'	'		3	
PINSRB j)		1 ' ' 1			^	^			1	1			
PINSRB j)		1 1							'	'		1	
PINSRW			-	-				1				'	
PINSRW		1 ' ' 1		-				'				2	
PINSRD j)		1 ' ' 1	•	-				1					
PINSRD j)				-				'				4	
PINSRQ j,m PINSRQ j,m xmm,r64,i	2,		•					4				'	
PINSRQ j,m	2,	1 1		-				'				4	
Arithmetic instructions	- ,	1 ' ' 1	-					,				'	
PADD/SUB(U)(S)B/W/D v,v 1 1 x x lint 1 0.5 PADD/SUB(U)(S)B/W/D mm/x,m 1 1 x x 1 int 1 1 x x 1 1 x x 1 int 1 1 1 x x 1 int 1 1 1 x x 1 int 1	PINSRQ J,m)	Xmm,m64,i	2	'			'				ını		I
PADD/SUB(U)(S)B/W/D v,v 1 1 x x lint 1 0.5 PADD/SUB(U)(S)B/W/D mm/x,m 1 1 x x 1 int 1 1 x x 1 1 x x 1 int 1 1 1 x x 1 int 1 1 1 x x 1 int 1	A widle we add a fire adversadile we												
PADD/SUB(U)(S)B/W/D mm/x,m 1 1 x x 1 int 1 1 x x 1 int 1 1 x x 1 int 1 1 PADDQ PSUBQ v,v 2 2 x x 1 int 2 1 int 2 1 int 2 1 int 1 1 PADDQ PSUBQ mm/x,m 2 2 x x 1 int 1		ļ l	4								:4		0.5
PADDQ PSUBQ v,v 2 2 x x int 2 1 PADDQ PSUBQ mm/x,m 2 2 x x 1 int 1 PHADD(S)W PHSUB(S)W h) v,v 3 3 1 2 1 int 3 2 PHADD(S)W pHSUB(S)W h) mm/x,m64 4 3 1 2 1 int 3 2 PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 3 2 PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 3 2 PCMPEQ/GTB/W/D v,v 1 1 x x 1 int 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1 int 1 1 PCMPEQQ j) xmm,m128 1 1 1 int 1 1 PMULL/HW PMU	` ' ' '	1 '						_				1	
PADDQ PSUBQ PHADD(S)W PHSUB(S)W h) mm/x,m 2 2 x 1 int 1 PHADD(S)W PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADD(S)W PHSUB(S)W h) mm/x,m64 4 3 1 2 1 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 1 int 3 2 PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1 int 1 </td <td></td> <td>1 1</td> <td>-</td> <td>1</td> <td></td> <td></td> <td></td> <td> 1</td> <td></td> <td></td> <td></td> <td></td> <td></td>		1 1	-	1				1					
PHADD(S)W PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADD(S)W PHSUB(S)W h) mm/x,m64 4 3 1 2 1 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1		1 '						,				2	
PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADD(S)W mm/x,m64 4 3 1 2 1 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1 int 1	I	mm/x,m	2	2	×		X	1			int		1
PHADD(S)W mm/x,m64 4 3 1 2 1 int 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1		.,,,	2	2	1		_				int	2	
PHSUB(S)W h) mm/x,m64 4 3 1 2 1 int 2 PHADDD PHSUBD h) v,v 3 3 1 2 1 int 3 2 PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 3 2 PCMPEQ/GTB/W/D v,v 1 1 x x 1 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1	, , ,	V,V	3	3	'		_				IIIL	3	
PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 3 2 PCMPEQ/GTB/W/D v,v 1 1 x x 1 int 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1 int 1 </td <td></td> <td>mm/y m64</td> <td>4</td> <td>3</td> <td>1</td> <td></td> <td>2</td> <td>1</td> <td></td> <td></td> <td>int</td> <td></td> <td>2</td>		mm/y m64	4	3	1		2	1			int		2
PHADDD PHSUBD h) mm/x,m64 4 3 1 2 1 int 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1 int 1 1 PCMPEQQ j) xmm,xmm 1 1 1 int 1 <td></td> <td>1 1</td> <td></td> <td></td> <td>1 -</td> <td></td> <td></td> <td>l '</td> <td></td> <td></td> <td></td> <td>3</td> <td></td>		1 1			1 -			l '				3	
PCMPEQ/GTB/W/D v,v 1 1 x x 1 1 0.5 PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1 int 1 PCMPEQQ j) xmm,xmm 1 1 1 int 1 1 PCMPEQQ j) xmm,m128 1 1 1 1 int 1 1 PMULL/HW PMULHUW v,v 1 1 1 1 int 3 1 PMULHRSW h) v,v 1 1 1 1 int 3 1 PMULLD j) xmm,xmm 4 4 2 2 int 5 2 PMULDQ j) xmm,m128 6 5 1 2 2 int 5 4 PMULDQ j) xmm,m128 1 1 1 1 int 3 1	,	I I			'			1					
PCMPEQ/GTB/W/D mm/x,m 1 1 x x 1 int 1 PCMPEQQ j) xmm,xmm 1 1 1 1 int 1 1 PMULL/HW PMULHUW v,v 1 1 1 1 int 3 1 PMULL/HW PMULHUW mm/x,m 1 1 1 1 int 3 1 PMULHRSW h) v,v 1 1 1 1 int 3 1 PMULLD j) xmm,xmm 4 4 2 2 int 5 2 PMULDQ j) xmm,xmm 1 1 1 int 3 1 PMULDQ j) xmm,xmm 1 1 1 int 5 4 PMULDQ j) xmm,m128 1 1 1 int 3 1	· ·	· · ·			'			'				1	
PCMPEQQ j) xmm,xmm 1	I	1		-				1				'	
PCMPEQQ j) xmm,m128 1	I	1			^			'				1	
PMULL/HW PMULHUW v,v 1		1 ' 1	-					4				'	
PMULL/HW PMULHUW mm/x,m 1	37	1 1	-			4		'				2	
PMULHRSW h) v,v 1 <		1	-					4				3	
PMULHRSW h) mm/x,m 1		1 1	-			-		'				_	
PMULLD j) xmm,xmm 4 4 4 2 2 int 5 2 PMULLD j) xmm,m128 6 5 1 2 2 1 int 5 4 PMULDQ j) xmm,xmm 1 1 1 1 int 3 1 PMULDQ j) xmm,m128 1 1 1 1 int 1	,	1	-					,				3	
PMULLD j) xmm,m128 6 5 1 2 2 1 int 5 4 PMULDQ j) xmm,xmm 1 1 1 1 int 3 1 PMULDQ j) xmm,m128 1 1 1 1 int 1	,	1	-	-			_	1				_	
PMULDQ j) xmm,xmm 1 1 1 1 int 3 1 PMULDQ j) xmm,m128 1 1 1 1 int 1		1	-	· ·		1							
PMULDQ j) xmm,m128 1 1 1 1 int 1			_		1		2	1					
		1 1	-			1 .						3	
PMULUDQ	3,	1 1	-			-		1					
	PMULUDQ	V,V	1	1		1					int	3	1

DMILLIDO		4	4	l	a	I	a	l I	1	int	1	1 4
PMULUDQ	mm/x,m	1	1 1		1 1		1			int	3	1
PMADDWD	V,V		1 -		•		 			int	3	•
PMADDWD	mm/x,m	1	1		1		1			int	•	1
PMADDUBSW h)	V,V	1	1		1					int	3	1
PMADDUBSW h)	mm/x,m	1	1		1		1			int		1
PAVGB/W	V,V	1	1	X		X				int	1	0.5
PAVGB/W	mm/x,m	1	1	X		X	1			int		1
PMIN/MAXSB j)	xmm,xmm	1	1	1						int	1	1
PMIN/MAXSB j)	xmm,m128	1	1	1			1			int		1
PMIN/MAXUB	V,V	1	1	X		х				int	1	0.5
PMIN/MAXUB	mm/x,m	1	1	X		х	1			int		1
PMIN/MAXSW	V,V	1	1	x		х				int	1	0.5
PMIN/MAXSW	mm/x,m	1	1	x		х	1			int		1
PMIN/MAXUW j)	xmm,xmm	1	1	1						int	1	1
PMIN/MAXUW j)	xmm,m	1	1				1			int		1
PMIN/MAXSD j)	xmm,xmm	1	1	1						int	1	1
PMIN/MAXSD j)	xmm,m128	1	1	1			1			int		1
PMIN/MAXUD j)	xmm,xmm	1	1	1						int	1	1
PMIN/MAXUD j)	xmm,m128	1	1	1			1			int	·	1
PHMINPOSUW j)	xmm,xmm	4	4	•		4	•			int	4	4
PHMINPOSUW j)	xmm,m128	4	4			4	1			int	-	4
PABSB PABSW PABSD h	· ·	1	1	x		X	'			int	1	0.5
PABSB PABSW PABSD	V,V		'	^		^				1110	ı	0.5
h)	mm/x,m	1	1	X		x	1			int		1
PSIGNB PSIGNW	, ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	•	'	^		^	'					
PSIGND h)	V,V	1	1	x		X				int	1	0.5
PSIGNB PSIGNW		•	•			^					-	0.0
PSIGND h)	mm/x,m	1	1	x		x	1			int		1
PSADBW	v,v	1	1		1					int	3	1
PSADBW	mm/x,m	1	1		1		1			int		1
MPSADBW j)	xmm,xmm,i	3	3		1	2	•			int	5	2
MPSADBW j)	xmm,m,i	4	3		1	2	1			int	O	2
ivii c/\bbvv j/	XIIIIII,III,I	-			'	_	'					_
Logic instructions												
PAND(N) POR PXOR	V,V	1	1	x	x	x				int	1	0.33
PAND(N) POR PXOR	mm/x,m	1		X	x	x	1			int	'	1
PTEST j)	xmm,xmm	2	2	1	x	x	'			int	1	1
PTEST j)	· '	2	2	1			1			int	'	1
1	xmm,m128		1	1 .	X	X	'				4	· ·
PSLL/RL/RAW/D/Q	mm,mm/i	1		1			4			int	1	1
PSLL/RL/RAW/D/Q	mm,m64	1	1	1			1			int	4	1
PSLL/RL/RAW/D/Q	xmm,i	1	1	1						int	1	1
PSLL/RL/RAW/D/Q	xmm,xmm	2	2	X		Х				int	2	1
PSLL/RL/RAW/D/Q	xmm,m128	3	2	X		Х	1			int		1
PSLL/RLDQ	xmm,i	1	1	X		Х				int	1	1
Othor												
Other EMMS		11	11	V	V	x				float		6
Notes:		1.1	11	Х	X	_ ^				nual		U

Notes:

g) SSE3 instruction set.

h) Supplementary SSE3 instruction set.

j) SSE4.1 instruction set

k) MASM uses the name MOVD rather than MOVQ for this instruction even

when moving 64 bits

m) Only available in 64 bit mode

Floating point XMM instructions

Instruction	instructions Operands	µops fused	μops	un	fus	ed d	lom	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	p5	p2	рЗ	p4			through- put
Move instructions												
MOVAPS/D	xmm,xmm	1	1	Х	Х	Х				int	1	0.33
MOVAPS/D	xmm,m128	1					1			int	2	1
MOVAPS/D	m128,xmm	1						1	1		3	1
MOVUPS/D	xmm,m128	4	2	1		1	2			int	2-4	2
MOVUPS/D	m128,xmm	9	4	Х	Х	X	1	2	2		3-4	4
MOVSS/D	xmm,xmm	1	1	Х	X	X				int	1	0.33
MOVSS/D	x,m32/64	1					1			int	2	1
MOVSS/D	m32/64,x	1						1	1		3	1
MOVHPS/D MOVLPS/D	xmm,m64	2	1			1	1			int	3	1
MOVHPS/D	m64,xmm	2	1	1				1	1		5	1
MOVLPS/D	m64,xmm	1						1	1		3	1
MOVLHPS MOVHLPS	xmm,xmm	1	1	1				•	-	float	1	1
MOVMSKPS/D	r32,xmm	1	1	1						float	1	1
MOVNTPS/D	m128,xmm	1 1	'	ļ '				1	1	noat	'	2-3
SHUFPS	xmm,xmm,i		1			1		'	'	int	1	1
SHUFPS	xmm,m128,i		i			1	1			int	'	1
SHUFPD	xmm,xmm,i	1	1	1		'	'			float	1	1
SHUFPD	xmm,m128,i		1	1			1			float	'	1
BLENDPS/PD j)		1	1	'		1	'			int	1	1
2,	xmm,xmm,i		1			'	1				1	
BLENDPS/PD j)	xmm,m128,i	1	1			1	1			int		1
BLENDVPS/PD j)	x,x,xmm0	2	2			2				int	2	2
BLENDVPS/PD j)	x,m,xmm0	2	2			2	1			int		2
MOVDDUP g)	xmm,xmm	1	1	1			١.			int	1	1
MOVDDUP g)	xmm,m64	2	1	1		١.	1			int		1
MOVSH/LDUP g)	xmm,xmm	1	1			1				int	1	1
MOVSH/LDUP g)	xmm,m128	2	1			1	1			int		1
UNPCKH/LPS	xmm,xmm	1	1			1				int	1	1
UNPCKH/LPS	xmm,m128	1	1			1	1			int		1
UNPCKH/LPD	xmm,xmm	1	1	1						float	1	1
UNPCKH/LPD	xmm,m128	2	1	1			1			float		1
EXTRACTPS j)	r32,xmm,i	2	2	Х	Х	X				int	4	1
EXTRACTPS j)	m32,xmm,i	2	1			1		1	1	int		1
INSERTPS j)	xmm,xmm,i	1	1			1				int	1	1
INSERTPS j)	xmm,m32,i	2	1			1	1			int		1
Conversion												
CVTPD2PS	xmm,xmm	2	2	1	1					float	4	1
CVTPD2PS	xmm,m128	2	2	1	1		1			float		1
CVTSD2SS	xmm,xmm	2	2	1	1					float	4	1
CVTSD2SS	xmm,m64	2	2	1	1		1			float		1
CVTPS2PD	xmm,xmm	2	2	2						float	2	2
CVTPS2PD	xmm,m64	2	2	2			1			float		2
CVTSS2SD	xmm,xmm	2	2	2						float	2	2
CVTSS2SD	xmm,m32	2	2	2			1			float		2
CVTDQ2PS	xmm,xmm	1	1	-	1		'			float	3	1

					_							
CVTDQ2PS	xmm,m128	1	1		1		1		float		1	l
CVT(T) PS2DQ	xmm,xmm	1	1		1				float	3	1	ı
CVT(T) PS2DQ	xmm,m128	1	1		1		1		float		1	l
CVTDQ2PD	xmm,xmm	2	2	1	1				float	4	1	l
CVTDQ2PD	xmm,m64	2	2	1	1		1		float		1	l
CVT(T)PD2DQ	xmm,xmm	2	2	1	1				float	4	1	l
CVT(T)PD2DQ	xmm,m128	2	2	1	1		1		float		1	l
CVTPI2PS	xmm,mm	1	1		1				float	3	3	ı
CVTPI2PS	xmm,m64	1	1		1		1		float		3	l
CVT(T)PS2PI	mm,xmm	1	1		1		-		float	3	1	l
CVT(T)PS2PI	mm,m128	1	1		1		1		float		1	l
CVTPI2PD	xmm,mm	2	2	1	1				float	4	1	ı
CVTPI2PD	xmm,m64	2	2	1	1		1		float		1	l
CVT(T) PD2PI	mm,xmm	2	2	1	1				float	4	1	l
CVT(T) PD2PI	mm,m128	2	2	1	1		1		float		1	l
CVTSI2SS	xmm,r32	1	1	'	1		'		float	4	3	l
CVTSI2SS	xmm,m32	1	1		1		1		float		3	l
CVT(T)SS2SI	r32,xmm	1	1		1		'		float	3	1	l
CVT(T)SS2SI	r32,m32	1	1		1		1		float	5	1	l
CVTSI2SD	xmm,r32	2	2	1	1		'		float	4	3	l
CVTSI2SD	xmm,m32	2	1	'	1		1		float		3	ı
CVT(T)SD2SI	r32,xmm	1	1		1		'		float	3	1	l
CVT(T)SD2SI	r32,m64		1		1		1		float	5	1	l
GV1(1)3B231	132,11104	'	'		'		'		iloat		!	l
Arithmetic												l
ADDSS/D SUBSS/D	xmm,xmm	1	1		1				float	3	1	l
ADDSS/D SUBSS/D	x,m32/64	1	1		1		1		float		1	l
ADDPS/D SUBPS/D	xmm,xmm	1	1		1		'		float	3	1	l
ADDPS/D SUBPS/D	xmm,m128	1	1		1		1		float		1	l
ADDSUBPS/D g)	xmm,xmm	1	1		1		'		float	3	1	l
ADDSUBPS/D g)	xmm,m128	1	1		1		1		float		1	l
HADDPS HSUBPS g)	xmm,xmm	3	3		1	2	'		float	7	3	l
HADDPS HSUBPS g)	xmm,m128	4	3		1	2	1		float	'	3	l
HADDPD HSUBPD g)	xmm,xmm	3	3	x	X	X	'		float	6	1.5	l
HADDPD HSUBPD g)	xmm,m128	4	3	x	x	X	1		float		1.5	l
MULSS	xmm,xmm	1	1	1	^	^	'		float	4	1.0	l
MULSS	xmm,m32	1	1	1			1		float		1	l
MULSD	xmm,xmm	1	1	1			'		float	5	1	ı
MULSD	xmm,m64	1	1	1			1		float	5	1	l
MULPS	xmm,xmm	1	1	1			'		float	4	1	l
MULPS	xmm,m128	1	1	1			1		float		1	l
MULPD	xmm,xmm	1		1			'		float	5	1	l
MULPD	xmm,m128	1		1			1		float	3	1	l
DIVSS	xmm,xmm			1			'		float	6-13 d)	5-12 d)	l
DIVSS	xmm,m32			1			1		float	0-13 d)	5-12 d) 5-12 d)	l
DIVSD	xmm,xmm		1	1			'		float	6-21 d)	5-12 d) 5-20 d)	l
DIVSD	xmm,m64	1		1			1		float	0-21 u)	5-20 d) 5-20 d)	l
DIVPS		1	1	1			'		float	6-13 d)	5-20 d) 5-12 d)	l
	xmm,xmm			1 -			4			0-13 u)		l
DIVPS	xmm,m128	1	1	1			1		float	6 24 4	5-12 d)	l
DIVPD	xmm,xmm	1	1	1			4		float	6-21 d)	5-20 d)	
DIVPD	xmm,m128	1	1	1	4		1		float	2	5-20 d)	
RCPSS/PS	xmm,xmm	1	1		1				float	3	2	l
RCPSS/PS	xmm,m	1	1		1		1		float		2	ı

CMPccSS/D	xmm,xmm	1	1		1					float	3	1
CMPccSS/D	x,m32/64	1	1		1		1			float		1
CMPccPS/D	xmm,xmm	1	1		1					float	3	1
CMPccPS/D	xmm,m128	1	1		1		1			float		1
COMISS/D UCOMISS/D	xmm,xmm	1	1		1					float	3	1
COMISS/D UCOMISS/D	x,m32/64	1	1		1		1			float		1
MAXSS/D MINSS/D	xmm,xmm	1	1		1					float	3	1
MAXSS/D MINSS/D	x,m32/64	1	1		1		1			float		1
MAXPS/D MINPS/D	xmm,xmm	1	1		1					float	3	1
MAXPS/D MINPS/D	xmm,m128	1	1		1		1			float		1
ROUNDSS/D j)	xmm,xmm,i	1	1		1					float	3	1
ROUNDSS/D j)	xmm,m128,i	1	1		1		1			float		1
ROUNDPS/D j)	xmm,xmm,i	1	1		1					float	3	1
ROUNDPS/D j)	xmm,m128,i	1	1		1		1			float		1
DPPS j)	xmm,xmm,i	4	4	2	2					float	11	3
DPPS j)	xmm,m128,i	4	4	2	2		1			float		3
DPPD j)	xmm,xmm,i	4	4	Х	Х	Х				float	9	3
DPPD j)	xmm,m128,i	4	4	Х	Х	Х	1			float		3
Math										_		
SQRTSS/PS	xmm,xmm	1	1	1						float	6-13	5-12
SQRTSS/PS	xmm,m	2	1	1			1			float		5-12
SQRTSD/PD	xmm,xmm	1	1	1						float	6-20	5-19
SQRTSD/PD	xmm,m	2	1	1			1			float		5-19
RSQRTSS/PS	xmm,xmm	1	1		1					float	3	2
RSQRTSS/PS	xmm,m	1	1		1		1			float		2
Logic												
AND/ANDN/OR/XORPS/D	xmm,xmm	1	1	Х	Х	Х				int	1	0.33
AND/ANDN/OR/XORPS/D	xmm,m128	1	1	Х	X	Х	1			int		1
Other												
LDMXCSR	m32	13	12	x	x	x	1					38
STMXCSR	m32	10	8	X	X	X	•	1	1			20
FXSAVE	m4096	151	67	X	X	X	8	38	38			145
FXRSTOR	m4096	121	74	х	х	х	47					150

Notes:

d) Round divisors give low values.

g) SSE3 instruction set.

Intel Nehalem

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, xmm = 128 bit xmm

register, mm/x = mmx or xmm register, sr = segment register, m = memory,

m32 = 32-bit memory operand, etc.

μορs fused domain: The number of μορs at the decode, rename, allocate and retirement stages in

the pipeline. Fused uops count as one.

μορs unfused domain: The number of μορs for each execution port. Fused μορs count as two. Fused

macro-ops count as one. The instruction has μ op fusion if the sum of the numbers listed under p015 + p2 + p3 + p4 exceeds the number listed under μ ops fused domain. An x under p0, p1 or p5 means that at least one of the μ ops listed under p015 can optionally go to this port. For example, a 1 under p015 and an x under p0 and p5 means one μ op which can go to either port 0 or port 5, whichever is vacant first. A value listed under p015 but nothing under p0, p1 and p5 means that it is not known which of the three ports these μ ops go to.

p015: The total number of μops going to port 0, 1 and 5.
p0: The number of μops going to port 0 (execution units).
p1: The number of μops going to port 1 (execution units).
p5: The number of μops going to port 5 (execution units).
p2: The number of μops going to port 2 (memory read).

p3: The number of μops going to port 3 (memory write address).p4: The number of μops going to port 4 (memory write data).

Domain: Tells which execution unit domain is used: "int" = integer unit (general purpose

registers), "ivec" = integer vector unit (SIMD), "fp" = floating point unit (XMM and x87 floating point). An additional "bypass delay" is generated if a register written by a μ op in one domain is read by a μ op in another domain. The bypass delay is 1 clock cycle between the "int" and "ivec" units, and 2 clock cy-

cles between the "int" and "fp", and between the "ivec" and "fp" units.

The bypass delay is indicated under latency only where it is unavoidable because either the source operand or the destination operand is in an unnatural domain such as a general purpose register (e.g. eax) in the "ivec" domain. For example, the PEXTRW instruction executes in the "int" domain. The source operand is an xmm register and the destination operand is a general purpose register. The latency for this instruction is indicated as 2+1, where 2 is the latency of the instruction itself and 1 is the bypass delay, assuming that the xmm operand is most likely to come from the "ivec" domain. If the xmm operand comes from the "fp" domain then the bypass delay will be 2 rather than one. The flags register can also have a bypass delay. For example, the COMISS instruction (floating point compare) executes in the "fp" domain and returns the result in the integer flags. Almost all instructions that read these flags execute in the "int" domain. Here the latency is indicated as 1+2, where 1 is the latency of the instruction itself and 2 is the bypass delay from the "fp" domain to the "int" domain.

The bypass delay from the memory read unit to any other unit and from any unit to the memory write unit are included in the latency figures in the table. Where the domain is not listed, the bypass delays are either unlikely to occur or unavoidable and therefore included in the latency figure.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give a similar delay. The time unit used is core clock cycles, not the reference clock cycles

given by the time stamp counter.

Reciprocal throughput: The average number of core clock cycles per instruction for a series of inde-

pendent instructions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μοps fused								Do- main	Laten- cy	procal
		do- main	p015	p0	p1	р5	p2	р3	p4			through- put
Move instructions												
MOV	r,r/i	1	1	Х	Х	х				int	1	0.33
MOV a)	r,m	1					1			int	2	1
MOV a)	m,r	1						1	1	int	3	1
MOV	m,i	1						1	1	int	3	1
MOV	r,sr	1					1			int		1
MOV	m,sr	2					1	1	1	int		1
MOV	sr,r	6	3	Х	х	х	3			int		13
MOV	sr,m	6	2	Х		х	4			int		14
MOVNTI	m,r	2						1	1	int	~270	1
MOVSX MOVZX	,											
MOVSXD	r,r	1	1	Х	х	х				int	1	0.33
MOVSX MOVZX												
MOVSXD	r,m	1					1			int		1
CMOVcc	r,r	2	2	Х	Х	х				int	2	1
CMOVcc	r,m	2	2	Х	Х	х	1			int		
XCHG	r,r	3	3	Х	Х	х				int	2	2
XCHG	r,m	7	Х				1	1	1	int	20 b)	
XLAT		2	1				1			int	5	1
PUSH	r	1						1	1	int	3	1
PUSH	i	1						1	1	int		1
PUSH	m	2					1	1	1	int		1
PUSH	sr	2	1					1	1	int		1
PUSHF(D/Q)		3	2	X	х	х		1	1	int		1
PUSHA(D) i)		18	2	X	1	x		8	8	int		8
POP	r	1					1			int	2	1
POP	(E/R)SP	3	2	x	1	x	1			int		5
POP	m	2					1	1	1	int		1
POP	sr	7	2				5			int		15
POPF(D/Q)		8	7	x	x	Х	1			int		14
POPA(D) i)		10	2	^	^		8			int		8
LAHF SAHF		1	1	x	x	х				int	1	0.33
SALC i)		2	2	x	x	X				int	4	1
LEA a)	r,m	1	1	^	1	^				int	1	1
BSWAP	r32	1	1		1					int	1	1 1
BSWAP	r64	1	1		1					int	3	1 1
LDS LES LFS LGS LSS	m	9	3		X	x	6			int		15
PREFETCHNTA		1	3	X	^	^	1			int		1
FREFEIGHNIA	m	1				1				1111		'

PREFETCHT0/1/2 LFENCE MFENCE SFENCE	m	1 2 3 2	1	x	x	x	1	1 1 1	1 1 1 1	int int int int		1 9 23 5
Arithmetic instructions												
ADD SUB	r,r/i	1	1		v	v				int	1	0.33
ADD SUB	r,m	1		X	X	X	1			int	'	
ADD SUB	m,r/i	2	1	X	X	X	1	1	1	int	6	1
ADC SBB	r,r/i	2	2				'	'	'	int	2	2
ADC SBB		2	2	X	X	X	1			int	2	2
ADC SBB	r,m m r/i	4	3	X	X	X	1	1	1	int	7	2
CMP	m,r/i	1	1	X	X	X	'	'	'		1	0.33
CMP	r,r/i	1	1	X	X	X	1			int int		1
INC DEC NEG NOT	m,r/i	1	1	X	X	X	'				1 1	0.33
INC DEC NEG NOT	r	3		X	X	X	1	4	1	int	6	_
	m	1	1 1	X	1	Х	'	1	'	int	3	1
AAA AAS DAA DAS i)			1		-	.,				int		1
AAD i)		3 5	3	X	X	X				int	15	2 7
AAM i)	" 0		5	X	X	Х				int	20	
MUL IMUL	r8	1	1		1					int	3	1
MUL IMUL	r16	3	3	X	X	X				int	5	2
MUL IMUL	r32	3	3	Х	Х	Х				int	5	2
MUL IMUL	r64	3	3	X	X	Х				int	3	2
IMUL	r16,r16	1	1		1					int	3	1
IMUL	r32,r32	1	1	,	1					int	3	1
IMUL	r64,r64	1	1	1						int	3	1
IMUL	r16,r16,i	1	1		1					int	3	1
IMUL	r32,r32,i	1	1	,	1					int	3	1
IMUL	r64,r64,i	1	1	1	4		4			int	3	2
MUL IMUL MUL IMUL	m8 m16	1 3	3	,,	1	.,	1			int	5	1 2
MUL IMUL	m32	3	3	X	X	X	1			int int	5	2
MUL IMUL	m64	3	2	2 2	Х	Х	1			int	3	2
IMUL	r16,m16	1	1	-	1		1			int	3	1
IMUL	r32,m32	1	1		1		1			int	3	1
IMUL	r64,m64	1	1	1	'		1			int	3	1
IMUL	r16,m16,i	1	1	'	1		1			int		1
IMUL	r32,m32,i	1	1		1		1			int		1
IMUL	r64,m64,i	1	1	1	'		1			int		1
DIV c)	r8	4	4	1	2	1	•			int	11-21	7-11
DIV c)	r16	6	6	X	4	X				int	17-22	7-12
DIV c)	r32	6	6	X	3	Х				int	17-28	7-17
DIV c)	r64	~40	Х	x	Х	х				int	28-90	19-69
IDIV c)	r8	4	4	1	2	1				int	10-22	7-11
IDIV c)	r16	8	8	x	5	Х				int	18-23	7-12
IDIV c)	r32	7	7	х	3	Х				int	17-28	7-17
IDIV c)	r64	~60	х	х	Х	Х				int	37-100	26-86
CBW CWDE CDQE		1	1	х	х	Х				int	1	1
CWD CDQ CQO		1	1	х		Х				int	1	1
POPCNT ℓ)	r,r	1	1		1					int	3	1
POPCNT ()	r,m	1	1		1		1			int		1
CRC32 ()	r,r	1	1		1					int	3	1
CRC32 ()	r,m	1	1		1		1			int		1

Logic instructions								l					
AND OR XOR	Logic instructions												
AND OR XOR AND OR XOR MR, fi 2		r,r/i	1	1	х	х	Х				int	1	0.33
AND OR XOR TEST	AND OR XOR		1	1	х	х	Х	1			int		1
TEST	AND OR XOR		2	1	х	х	х	1	1	1	int	6	1
TEST	TEST		1	1	х	х	Х				int	1	0.33
SHR SHL SAR	TEST		1	1	x	х	х	1			int		1
SHR SHL SAR	1		1		x		х					1	0.5
ROR ROL	1							1	1	1		6	
ROR ROL RCR RCL r,1 2 2 2 x x x x x x x	ROR ROL	,	1	1	x		х				int	1	1
RCR RCL R,1 2 2 2 x x x x x x x			3				х	1	1	1	int	6	
RCR r8,i/cl 8	1				x	х	х						
RCL	1						х						
RCR RCL													
RCR RCL	1												12-13
RCR	1							1	1	1			
RCL m8,i/cl 11 8 x x x 1 1 1 int 14 RCR RCL m16/32/64,i/cl 10 7 x x x 1 1 int 15 SHLD r,r,i/cl 2 2 x x x 1 1 int 15 SHD m,r,i/cl 3 2 x x x 1 1 int 4 1 SHRD m,r,i/cl 3 2 x x x 1 1 int 4 1 SHRD m,r,i/cl 3 2 x x x 1 int 4 1 SHRD m,r,i/cl 3 2 x x x 1 int 4 1 BT m,r 0 7 x x x 1 1 1 1 1 1 1 1	1	· ·											
RCR RCL	1	· ·											
SHLD								_		'			
SHLD	1	·							'	'			1
SHRD	1			1				1	1	1			•
SHRD	1			1				'	'	'			1
BT	1							1	1	1			'
BT	1					^		'	'	'			1
BT	1		-					1					
BTR BTS BTC r,r/i 1 1 x x x l int 1 1 x x x l int 1 1 x x x x l int 1 1 1 int 6 BFR BTS BTC m,r 1 1 1 1 1 int 6 BFBS BSR r,r 1 1 1 1 int 3 1 BSF BSR r,r 1 1 1 1 int 3 1 SETcc r 1 1 x x x x int 1 1 x x x int 1 1 x x x int 1 1 x x x x 1 1 x x x x x x x x x x x x x x x x x x <th< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></th<>													
BTR BTS BTC m,r 10 7 x x 1 1 1 int 6 BTR BTS BTC m,i 3 3 x x 1 1 1 int 6 BSF BSR r,r 1 1 1 1 1 int 3 1 SETcc r 1 1 x x x 1 1 int 1 1 SETcc m 2 1 x x x 1 1 int 1 1 SETcc m 2 1 x x x int 1 1 x x x int 1 1 x x x int 1 1 x x x x int 0 2 x x x x x x x x x x x x x x </td <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>'</td> <td></td> <td></td> <td></td> <td>1</td> <td></td>								'				1	
BTR BTS BTC m,i 3 3 x x 1 1 int 6 BSF BSR r,r 1 1 1 1 1 1 int 3 1 SETcc r 1 1 x x x 1 1 int 1 1 SETcc m 2 1 x x x 1 1 int 1	1					v		1	1	4			'
BSF BSR r,r 1	1					^		-		'			
BSF BSR					^	1	^	١.	'	'			1
SETCC	1		-					1					-
SETCC	1				v	'	v	١.					-
CLC STC CMC 1 1 x x x int 1 0.33 CLD 2 2 x x x int 4 4 STD 2 2 x x x int 4 4 STD 3 2 2 x x x int 4 4 STD 3 3 3 x x x int 4 5 Control transfer instructions 3 3 3 3 int 5 5 JMP short/near 1 1 1 int 0 2 2 3						_			1	1			
CLD 2 2 x x x x int 4 STD 2 2 x x x x x int 5 Control transfer instructions JMP short/near 1 1 1 int 0 2 JMP int far 31 31 31 int 0 2 JMP int m(near) 1 1 1 1 int 0 2 JMP int m(far) 31 31 11 int 0 2 JMP m(far) m(far) 31 31 11 int 0 2 JMP m(far) short/near 1 1 1 int 0 2 JMP m(far) short/near 1 1 1 int 0 2 LOOP(E/R)CXZ short 2 2 x x 1 int 0 2	1	111							'	'		1	
STD													
Control transfer instructions 1 1 1 1 int 0 2 JMP i) far 31 31 31 int 0 2 JMP r 1 1 1 1 int 0 2 JMP m(near) 1 1 1 1 int 0 2 JMP m(far) 31 31 31 11 int 0 2 JMP m(far) 31 31 1 1 int 0 2 Fused compare/test and branch e) 1 1 1 1 int 0 2 LOOP short 2 2 x <td></td> <td></td> <td></td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>				1									
JMP short/near 1 1 1 1 1 1 1 67 JMP i) far 31 31 31 31 31 int 0 2 JMP m(near) 1 1 1 1 int 0 2 JMP m(far) 31 31 31 11 int 0 2 JMP m(far) 31 31 11 int 0 2 JMP m(far) 31 31 11 int 0 2 JMP m(far) 31 31 11 1 int 0 2 Fused compare/test and branch e) 1 1 1 1 int 0 2 J(E/R)CXZ short 2 2 x x int 2 LOOP short 11 11 x x x int 7 CALL	315		2	_	^	^	^				IIIC		3
JMP short/near 1 1 1 1 1 1 1 67 JMP i) far 31 31 31 31 31 int 0 2 JMP m(near) 1 1 1 1 int 0 2 JMP m(far) 31 31 31 11 int 0 2 JMP m(far) 31 31 11 int 0 2 JMP m(far) 31 31 11 int 0 2 JMP m(far) 31 31 11 1 int 0 2 Fused compare/test and branch e) 1 1 1 1 int 0 2 J(E/R)CXZ short 2 2 x x int 2 LOOP short 11 11 x x x int 7 CALL	Control transfer instructi	ons											
JMP i) far 31 31 1 1 1 67 JMP r 1 1 1 1 1 int 0 2 JMP m(far) 31 31 31 11 int 0 2 JMP m(far) 31 31 31 11 int 0 2 JMP m(far) 31 31 31 11 1 int 0 2 Fused compare/test and branch e) 1 1 1 1 int 0 2 J(E/R)CXZ short 2 2 x x int 4 LOOP short 6 <td< td=""><td></td><td></td><td>1</td><td>1</td><td></td><td></td><td>1</td><td></td><td></td><td></td><td>int</td><td>0</td><td>2</td></td<>			1	1			1				int	0	2
JMP r 1	1												
JMP m(near) 1	,						1					0	
JMP m(far) 31 31 11 int 73 Conditional jump short/near 1 1 1 1 int 0 2 Fused compare/test and branch e) 1 1 1 1 int 0 2 J(E/R)CXZ short 2 2 x x 1 int 2 LOOP short 6 6 x x x int 4 LOOP(N)E short 11 11 x x x int 7 CALL near 2 2 ? 1 1 1 int 7 CALL i) far 46 46 9 int 74 CALL r 3 2 ? 1 1 1 int 2 CALL m(near) 4 3 ? 1 1 1 int 7			=				_	1					
Conditional jump short/near 1 1 1 1 1 1 0 2 Fused compare/test and branch e) 1 1 1 1 1 int 0 2 J(E/R)CXZ short 2 2 x x 1 int 2 LOOP short 6 6 x x x int 4 LOOP(N)E short 11 11 x x x int 7 CALL near 2 2 ? 1 1 1 int 2 CALL i) far 46 46 9 int 74 CALL r 3 2 ? 1 1 1 int 2 CALL m(near) 4 3 ? 1 1 1 int 2		, ,	31				-	11					
Fused compare/test and branch e) 1 <	1	, ,					1					0	
J(E/R)CXZ short 2 2 x x 1 int 2 LOOP short 6 6 x x x int 4 LOOP(N)E short 11 11 x x int 7 CALL near 2 2 ? 1 1 1 int 2 CALL i) far 46 46 9 int 74 CALL r 3 2 ? 1 1 1 int 2 CALL m(near) 4 3 ? 1 1 1 int 2	· ·	l	1				1						
LOOP short 6 6 x x x x x int 4 LOOP(N)E short 11 11 x x x int 7 CALL near 2 2 ? 1 1 1 int 2 CALL i) far 46 46 9 int 74 CALL r 3 2 ? 1 1 1 int 2 CALL m(near) 4 3 ? ? 1 1 1 int 2	-	,	-		х	Х							
LOOP(N)E short 11 11 x x x x int 7 CALL near 2 2 ? ? 1 1 1 int 2 CALL i) far 46 46 9 int 74 CALL r 3 2 ? ? 1 1 int 2 CALL m(near) 4 3 ? ? 1 1 1 int 2	` '						-						
CALL near 2 2 ? 1 1 1 int 2 CALL i) far 46 46 46 9 int 74 CALL r 3 2 ? 1 1 1 int 2 CALL m(near) 4 3 ? ? 1 1 1 int 2													
CALL i) far 46 46 9 int 74 CALL r 3 2 ? 1 1 1 int 2 CALL m(near) 4 3 ? ? 1 1 1 int 2	1								1	1			
CALL r 3 2 ? 1 1 1 int 2 CALL m(near) 4 3 ? ? 1 1 1 int 2								9					
CALL m(near) 4 3 ? ? 1 1 1 1 int 2					?	?	1		1	1			
						1	_	1		'			
		, ,		1									

RETN		1	1			1	1			int		2
RETN	i	3	2			1	1			int		2
RETF		39	39							int		120
RETF	i	40	40							int		124
BOUND i)	r,m	15	13				2			int		7
INTO i)	,	4	4							int		5
String instructions												
LODS		2	1	х	Х	Х	1			int		1
REP LODS		11+4n	'						.	int	40+12n	
STOS		3	1	Х	Х	Х		1	1	int		1
REP STOS	small n	60+n							.	int	12+n	
REP STOS	large n	2.5/16	bytes							int	1 clk / 1	6 bytes
MOVS		5	2	х	Х	Х	1	1	1	int		4
REP MOVS	small n	13+6n	•							int	12+n	
REP MOVS	large n	2/16 by	tes/							int	1 clk / 1	6 bytes
SCAS		3	2	х	х	Х	1			int		1
REP SCAS		37+6n								int	40+2n	
CMPS		5	3	х	х	Х	2			int		4
REP CMPS		65+8n		I					.	int	42+2n	
Other												
NOP (90)		1	1	Х	Х	Х				int		0.33
Long NOP (0F 1F)		1	1	Х	Х	Х				int		1
PAUSE		5	5	х	Х	Х				int		9
ENTER	a,0	11	9	х	Х	Х	1	1	1	int		8
ENTER	a,b	34+7b								int	79+5b	
LEAVE		3	3				1			int		5
CPUID		25-100								int	~200	~200
RDTSC		22								int		24
RDPMC		28								int		40-60

Notes:

a) Applies to all addressing modesb) Has an implicit LOCK prefix.

c) Low values are for small results, high values for high results.

e) See manual 3: "The microarchitecture of Intel, AMD and VIA CPUs" for restric-

tions on macro-op fusion.

i) Not available in 64 bit mode.

ℓ) SSE4.2 instruction set.

Floating point x87 instructions

Instruction	Operands	μοps fused	µops	un	fuse	ed d	lom	Do- main	Laten- cy	Reci- procal		
		do- main	p015	p0	p1	р5	p2	р3	p4			through- put
Move instructions												
FLD	r	1	1	1						float	1	1
FLD	m32/64	1	1				1			float	3	1
FLD	m80	4	2	1	1		2			float	4	2
FBLD	m80	41	38	Х	Х	Х	3			float	45	20
FST(P)	r	1	1	1						float	1	1
FST(P)	m32/m64	1						1	1	float	4	1

FSTP	m80	7	3	х	х	х		2	2	float	5	5
FBSTP	m80	208	204	X	X	X		2	2	float	242	245
FXCH	r	1	0 f)	^	^	_ ^		_	-	float	0	1
FILD	m '	1	1		1		1			float	6	1
FIST(P)		3	1		1		'	1	1	float	7	1
, ,	m m	3	1		1			1			7	
FISTTP g)	m m			,	'			'	'	float	/	1
FLDZ		1	1	1	١,					float		1
FLD1		2	2	1	1					float		2
FLDPI FLDL2E etc.		2	2		2					float		2
FCMOVcc	r	2	2	2						float	2+2	2
FNSTSW	AX	2	2							float		1
FNSTSW	m16	3	2					1	1	float		2
FLDCW	m16	2	1				1			float	7	31
FNSTCW	m16	2	1	1				1	1	float	5	1
FINCSTP FDECSTP		1	1	1						float	1	1
FFREE(P)	r	2	2	Х	Х	Х				float		4
FNSAVE	m m	143	89	Х	Х	Х	8	23	23	float	178	178
FRSTOR	m	79	52	х	Х	Х	27			float	156	156
Arithmetic instructions										<i>a</i> ,		4
FADD(P) FSUB(R)(P)	r	1	1		1					float	3	1
FADD(P) FSUB(R)(P)	m m	1	1		1		1			float	_	1
FMUL(P)	r	1	1	1						float	5	1
FMUL(P)	m m	1	1	1			1			float		1
FDIV(R)(P)	r	1	1	1						float	7-27 d)	7-27 d)
FDIV(R)(P)	m m	1	1	1			1			float	7-27 d)	7-27 d)
FABS		1	1	1						float	1	1
FCHS		1	1	1						float	1	1
FCOM(P) FUCOM	r	1	1		1					float		1
FCOM(P) FUCOM	m m	1	1		1		1			float		1
FCOMPP FUCOMPP		2	2	1	1					float		1
FCOMI(P) FUCOMI(P)	r	1	1		1					float		1
FIADD FISUB(R)	m	2	2		2		1			float	3	2
FIMUL	m	2	2	1	1		1			float	5	2
FIDIV(R)	m	2	2	1	1		1			float	7-27 d)	7-27 d)
FICOM(P)	m	2	2		2		1			float		1
FTST		1	1		1					float		1
FXAM		1	1		1					float		1
FPREM		25	25	Х	Х	Х				float	14	
FPREM1		35	35	х	Х	Х				float	19	
FRNDINT		17	17	х	Х	Х				float	22	
Math		2.								o .	40	
FSCALE		24	24	Х	Х	Х				float	12	
FXTRACT		17	17	X	Х	Х				float	13	
FSQRT		1	1	1						float	~27	
FSIN		~100	~100	Х	Х	Х				float	40-100	
FCOS		~100	~100	Х	Х	Х				float	40-100	
FSINCOS		~100	~100	Х	Х	Χ				float	~110	
F2XM1		19	19	Х	Х	Χ				float	58	
FYL2X FYL2XP1		~55	~55	х	Х	Х				float	~80	
FPTAN		~100	~100	х	Х	Х				float	~115	
FPATAN		~82	~82	х	Х	Х				float	~120	

Other									
FNOP	1	1	1				float	1	
WAIT	2	2	Х	х	Х		float	1	
FNCLEX	3	3		Х	Χ		float	17	
FNINIT	~190	~190	х	х	Х		float	77	

Notes:

d) Round divisors or low precision give low values.

f) Resolved by register renaming. Generates no μops in the unfused domain.

g) SSE3 instruction set.

Integer MMX and XMM instructions

Instruction	Operands	μορs fused	μops	un	fus	ed d	oma	ain		Do- main	Laten- cy	procal
		do- main	p015	p0	p1	р5	p2	р3	p4			through- put
Move instructions												
MOVD k)	r32/64,mm/x	1	1	Х	Х	х				int	1+1	0.33
MOVD k)	m32/64,mm/x	1						1	1		3	1
MOVD k)	mm/x,r32/64	1	1	Х	Х	х				ivec	1+1	0.33
MOVD k)	mm/x,m32/64	1					1				2	1
MOVQ	mm/x, mm/x	1	1	Х	х	х				ivec	1	0.33
MOVQ	mm/x,m64	1					1				2	1
MOVQ	m64, mm/x	1						1	1		3	1
MOVDQA	xmm, xmm	1	1	Х	х	х				ivec	1	0.33
MOVDQA	xmm, m128	1					1				2	1
MOVDQA	m128, xmm	1						1	1		3	1
MOVDQU	xmm, m128	1	1				1				2	1
MOVDQU	m128, xmm	1	1					1	1		3	1
LDDQU g)	xmm, m128	1	1				1				2	1
MOVDQ2Q	mm, xmm	1	1	X	X	x				ivec	1	0.33
MOVQ2DQ	xmm,mm	1	1	X	X	x				ivec	1	0.33
MOVNTQ	m64,mm	1						1	1		~270	2
MOVNTDQ	m128,xmm	1						1	1		~270	2
MOVNTDQA j)	xmm, m128	1					1				2	1
PACKSSWB/DW	,											
PACKUSWB	mm,mm	1	1		1					ivec	1	1
PACKSSWB/DW												
PACKUSWB	mm,m64	1	1		1		1					2
PACKSSWB/DW												
PACKUSWB	xmm,xmm	1	1	Х		Х				ivec	1	0.5
PACKSSWB/DW												
PACKUSWB	xmm,m128	1	1	Х		X	1					2
PACKUSDW j)	xmm,xmm	1	1	Х		X				ivec	1	2
PACKUSDW j)	xmm,m	1	1	Х		Х	1					2
PUNPCKH/LBW/WD/DQ	mm/x, mm/x	1	1	X		Х				ivec	1	0.5
PUNPCKH/LBW/WD/DQ	mm/x,m	1	1	Х		Х	1					2
PUNPCKH/LQDQ	xmm,xmm	1	1	Х		X				ivec	1	0.5
PUNPCKH/LQDQ	xmm, m128	2	1	Х		X	1					1
PMOVSX/ZXBW j)	xmm,xmm	1	1	Х		Х				ivec	1	1
PMOVSX/ZXBW j)	xmm,m64	1	1	Х		Х	1					2
PMOVSX/ZXBD j)	xmm,xmm	1	1	Х		X				ivec	1	1
PMOVSX/ZXBD j)	xmm,m32	1	1	Х		Х	1					2

			1									,
PMOVSX/ZXBQ j)	xmm,xmm	1	1	Х		Χ				ivec	1	1
PMOVSX/ZXBQ j)	xmm,m16	1	1	х		Χ	1					2
PMOVSX/ZXWD j)	xmm,xmm	1	1	х		Х				ivec	1	1
PMOVSX/ZXWD j)	xmm,m64	1	1	х		Х	1					2
PMOVSX/ZXWQ j)	xmm,xmm	1	1	х		Х				ivec	1	1
PMOVSX/ZXWQ j)	xmm,m32	1	1	х		Х	1					2
PMOVSX/ZXDQ j)	xmm,xmm	1	1	х		Х				ivec	1	1
PMOVSX/ZXDQ j)	xmm,m64	1	1	x		Х	1					2
PSHUFB h)	mm/x, mm/x	1	1	x		Х				ivec	1	0.5
PSHUFB h)	mm/x,m	2	1	x		Х	1					1
PSHUFW	mm,mm,i	1	1	X		Х				ivec	1	0.5
PSHUFW	mm,m64,i	2	1	X		Х	1				·	1
PSHUFD	xmm,xmm,i	1	1	X		Х				ivec	1	0.5
PSHUFD	xmm,m128,i	2	1	X		X	1			1700	•	1
PSHUFL/HW	xmm,xmm,i	1	1	X		X				ivec	1	0.5
PSHUFL/HW	xmm, m128,i	2	1	X		X	1			1000		1
PALIGNR h)	mm/x,mm/x,i	1	1	x		X				ivec	1	1
PALIGNR h)	mm/x,m,i	2	1				1			IVEC	•	1
,		2	2	1 X		X	1			ivoo	2	
PBLENDVB j)	x,x,xmm0		2	1 .		1	4			ivec	2	1
PBLENDVB j)	xmm,m,xmm0	3		1		1	1				4	1
PBLENDW j)	xmm,xmm,i	1	1	Х		Х				ivec	1	0.5
PBLENDW j)	xmm,m,i	2	1	X		Х	1	_				1
MASKMOVQ	mm,mm	4	1	1			1	1	1	ivec		2
MASKMOVDQU	xmm,xmm	10	4	X	Х	Х	2	2	X	ivec		7
PMOVMSKB	r32,mm/x	1	1	1						float	2+2	1
PEXTRB j)	r32,xmm,i	2	2	X	Х	Χ				ivec	2+1	1
PEXTRB j)	m8,xmm,i	2	2	Х		Χ						1
PEXTRW	r32,mm/x,i	2	2	X	Х	Χ				ivec	2+1	1
PEXTRW j)	m16,mm/x,i	2	2	X		Х		1	1			1
PEXTRD j)	r32,xmm,i	2	2	X	Х	Х				ivec	2+1	1
PEXTRD j)	m32,xmm,i	2	1	Х		Χ		1	1			1
PEXTRQ j,m)	r64,xmm,i	2	2	Х	Х	Χ				ivec	2+1	1
PEXTRQ j,m)	m64,xmm,i	2	1	Х		Χ		1	1			1
PINSRB j)	xmm,r32,i	1	1	Х		Χ				ivec	1+1	1
PINSRB j)	xmm,m8,i	2	1	х		Χ	1					1
PINSRW	mm/x,r32,i	1	1	х		Х				ivec	1+1	1
PINSRW	mm/x,m16,i	2	1	х		Х	1					1
PINSRD j)	xmm,r32,i	1	1	х		Х				ivec	1+1	1
PINSRD j)	xmm,m32,i	2	1	х		Х	1					1
PINSRQ j,m)	xmm,r64,i	1	1	Х		Х				ivec	1+1	1
PINSRQ j,m)	xmm,m64,i	2	1	х		Х	1					1
3. ,	, ,											
Arithmetic instructions												
PADD/SUB(U)(S)B/W/D/												
Q	mm/x, mm/x	1	1	Х		Х				ivec	1	0.5
PADD/SUB(U)(S)B/W/D/												
Q	mm/x,m	1	1	Х		Х	1					2
PHADD/SUB(S)W/D h)	mm/x, mm/x	3	3	Х		Х				ivec	3	1.5
PHADD/SUB(S)W/D h)	mm/x,m64	4	3	Х		Х	1					3
PCMPEQ/GTB/W/D	mm/x,mm/x	1	1	Х		Х				ivec	1	0.5
PCMPEQ/GTB/W/D	mm/x,m	1	1	х		Х	1					2
PCMPEQQ j)	xmm,xmm	1	1	Х		Х				ivec	1	0.5
PCMPEQQ j)	xmm,m128	1	1	Х		Х	1					2
	, .==		1	1 - 7	1	·	I	ı			ı İ	ļ

PCMPGTQ () xmm,xmm 1 1 1 1 ivec 3	1
PCMPGTQ ℓ) xmm,m128 1 1 1 1 1	1
PMULL/HW PMULHUW mm/x,mm/x 1 1 1 ivec 3	1
PMULL/HW PMULHUW mm/x,m 1 1 1 1 1	1
PMULHRSW h) mm/x,mm/x 1 1 1 ivec 3	1
PMULHRSW h) mm/x,m 1 1 1 1	1
PMULLD j) xmm,xmm 2 2 2 2 ivec 6	2
PMULLD j) xmm, m128 3 2 2 1	
PMULDQj) xmm,xmm 1 1 1 ivec 3	1
PMULDQ j)	1
PMULUDQ mm/x,mm/x 1 1 1 ivec 3	1
PMULUDQ mm/x,m 1 1 1 1 1 1 1 1 1	1
PMADDWD mm/x,mm/x 1 1 1 ivec 3	1
PMADDWD mm/x,m 1 1 1 1 1 1 1 1 1	1
PMADDUBSW h) mm/x,mm/x 1 1 1 ivec 3	1
PMADDUBSW h) mm/x,m 1 1 1 1 1 1	1
PAVGB/W mm/x,mm/x 1 1 x x i ivec 1	0.5
PAVGB/W mm/x,m 1 1 x x 1	1
PMIN/MAXSB j) xmm,xmm 1 1 x x x i ivec 1	1
PMIN/MAXSB j)	2
PMIN/MAXUB mm/x,mm/x 1 1 x x i ivec 1	0.5
PMIN/MAXUB mm/x,m 1 1 x x 1	2
PMIN/MAXSW mm/x,mm/x 1 1 x x x ivec 1	0.5
	2
	1
, , , , , , , , , , , , , , , , , , ,	2
, , , , , , , , , , , , , , , , , , , ,	
PMIN/MAXU/SD j) xmm,xmm 1 1 x x ivec 1	1
PMIN/MAXU/SD j) xmm,m128 1 1 x x 1	2
PHMINPOSUW j) xmm,xmm 1 1 1 ivec 3	1
PHMINPOSUW j) xmm,m128 1 1 1 1 1	3
PABSB PABSW PABSD mm/x,mm/x 1 1 x x ivec 1	0.5
	0.5
PABSB PABSW PABSD	1
PSIGNB PSIGNW	1
PSIGND h) mm/x,mm/x 1 1 x x ivec 1	0.5
PSIGNB PSIGNW	0.0
PSIGND h mm/x,m	2
PSADBW mm/x,mm/x 1 1 1 ivec 3	1
PSADBW mm/x,m 1 1 1 1	3
MPSADBW j) xmm,xmm,i 3 3 x x x ivec 5	1
MPSADBW j) xmm,m,i 4 3 x x x 1	2
PCLMULQDQ n) xmm,xmm,i 12	8
AESDEC, AESDECLAST,	
AESENC, AESENCLAST	
n)	
xmm,xmm	~2
AESIMC n) xmm,xmm ~5	~2
AESKEYGENASSIST n) xmm,xmm,i ~5	~2
Logic instructions	
PAND(N) POR PXOR mm/x,mm/x 1 1 x x x i ivec 1	0.33
PAND(N) POR PXOR mm/x,m 1 1 x x x 1	1
PTEST j) xmm,xmm 2 2 x x x ivec 3	1

PTEST j)	xmm,m128	2	2	Х	Х	Х	1			1	
PSLL/RL/RAW/D/Q	mm,mm/i	1	1		1			ivec	1	1	
PSLL/RL/RAW/D/Q	mm,m64	1	1		1		1			2	
PSLL/RL/RAW/D/Q	xmm,i	1	1		1			ivec	1	1	
PSLL/RL/RAW/D/Q	xmm,xmm	2	2	х	1	Х		ivec	2	2	
PSLL/RL/RAW/D/Q	xmm,m128	3	2	х	1	Х	1			1	
PSLL/RLDQ	xmm,i	1	1	х		Х		ivec	1	1	
String instructions											
PCMPESTRI ℓ)	xmm,xmm,i	8	8	х	Х	Х		ivec	14	5	
PCMPESTRI ()	xmm,m128,i	9	8	Х	Х	Х	1	ivec	14	6	
PCMPESTRM ℓ)	xmm,xmm,i	9	9	х	Х	Х		ivec	7	6	
PCMPESTRM ()	xmm,m128,i	10	10	х	Х	Х	1	ivec	7	6	
PCMPISTRI ()	xmm,xmm,i	3	3	Х	Х	Х		ivec	8	2	
PCMPISTRI ℓ)	xmm,m128,i	4	4	х	Х	Х	1	ivec	8	2	
PCMPISTRM ℓ)	xmm,xmm,i	4	4	х	Х	Х		ivec	7	2	
PCMPISTRM ℓ)	xmm,m128,i	6	5	х	х	Х	1	ivec	7	5	
Other											
EMMS		11	11	х	х	Х		float		6	

Notes:

g) SSE3 instruction set.

h) Supplementary SSE3 instruction set.

j) SSE4.1 instruction set

k) MASM uses the name MOVD rather than MOVQ for this instruction even when

moving 64 bits

ℓ) SSE4.2 instruction set

m) Only available in 64 bit mode

n) Only available on newer models

Floating point XMM instructions

Instruction	Operands	µops fused	μops	un	fus	ed d	loma	ain		Do- main	Laten- cy	Reci- procal
		do- main	p015	p0	p1	р5	p2	р3	p4			through- put
Move instructions												
MOVAPS/D	xmm,xmm	1	1			1				float	1	1
MOVAPS/D	xmm,m128	1					1				2	1
MOVAPS/D	m128,xmm	1						1	1		3	1
MOVUPS/D	xmm,m128	1					1				2	1-4
MOVUPS/D	m128,xmm	1						1	1		3	1-3
MOVSS/D	xmm,xmm	1	1			1					1	1
MOVSS/D	xmm,m32/64	1					1				2	1
MOVSS/D	m32/64,xmm	1						1	1		3	1
MOVHPS/D MOVLPS/D	xmm,m64	2	1			1	1				3	2
MOVH/LPS/D	m64,xmm	2	1			1		1	1		5	1
MOVLHPS MOVHLPS	xmm,xmm	1	1			1				float	1	1
MOVMSKPS/D	r32,xmm	1	1	1						float	1+2	1
MOVNTPS/D	m128,xmm	1						1	1		~270	2
SHUFPS/D	xmm,xmm,i	1	1			1				float	1	1
SHUFPS/D	xmm,m128,i	2	1			1	1			float		1
BLENDPS/PD j)	xmm,xmm,i	1	1			1				float	1	1

BLENDPS/PD j) BLENDVPS/PD j) BLENDVPS/PD j) MOVDDUP g) MOVDDUP g) MOVSH/LDUP g) MOVSH/LDUP g) UNPCKH/LPS/D UNPCKH/LPS/D EXTRACTPS j) EXTRACTPS j) INSERTPS j)	xmm,m128,i x,x,xmm0 xmm,m,xmm0 xmm,xmm xmm,m64 xmm,xmm xmm,m128 xmm,xmm xmm,m128 r32,xmm,i m32,xmm,i xmm,xmm,i	2 2 3 1 1 1 1 1 1 1 1 2 1 3	1 2 2 1 1 1 1 1 1 1 1 2			1 2 2 1 1 1 1 1 1 1 1 2	1 1 1 1 1 1	1	1	float float float float float float float float float	2 1 2 1 1 1+2	1 2 2 1 1 1 1 1 1 1 1	
Conversion													
CVTPD2PS	xmm,xmm	2	2		1	1				float	4	1	
CVTPD2PS	xmm,m128	2	2		1		1			float		1	
CVTSD2SS	xmm,xmm	2	2		1	1				float	4	1	
CVTSD2SS	xmm,m64	2	2	?	?	?	1			float		1	
CVTPS2PD	xmm,xmm	2	2	1		1				float	2	1	
CVTPS2PD	xmm,m64	2	2	1		1	1			float		1	
CVTSS2SD	xmm,xmm	1	1	1						float	1	1	
CVTSS2SD	xmm,m32	1	1	1			1			float		2	
CVTDQ2PS	xmm,xmm	1	1		1					float	3+2	1	
CVTDQ2PS	xmm,m128	1	1		1		1			float		1	
CVT(T) PS2DQ	xmm,xmm	1	1		1					float	3+2	1	
CVT(T) PS2DQ	xmm,m128	1	1		1		1			float		1	
CVTDQ2PD	xmm,xmm	2	2		1	1				float	4+2	1	
CVTDQ2PD	xmm,m64	2	2		1	1	1			float		1	
CVT(T)PD2DQ	xmm,xmm	2	2		1	1				float	4+2	1	
CVT(T)PD2DQ	xmm,m128	2	2		1	1	1			float	0.0	1	
CVTPI2PS	xmm,mm	1	1		1		,			float	3+2	3	
CVTPI2PS	xmm,m64	1	1		1		1			float	0.0	3	
CVT(T)PS2PI	mm,xmm	1	1		1		4			float	3+2	1	
CVT(T)PS2PI	mm,m128	1 2	1		1	4	1			float	6	1	
CVTPI2PD	xmm,mm	2	2		1	1	1			ivec/float	6	1	
CVT/T/ PD2DI	xmm,m64	2	2	,,	1	-	'			6 1 4 /3,	6	1	
CVT(T) PD2PI	mm,xmm mm,m128	2	2	X	1	X	1			float/ivec	6	1 1	
CVT(T) PD2PI CVTSI2SS	xmm,r32	1	1	X	1	Х	'			float	3+2	3	
CVTSI2SS CVTSI2SS	xmm,m32	1	1		1		1			float	3+2	3	
CVT(T)SS2SI	r32,xmm	1	1		1		'			float	3+2	1	
CVT(T)SS2SI	r32,m32	1	1		1		1			float	312	1	
CVT(1)33231 CVTSI2SD	xmm,r32	2	2	1	1		'			float	4+2	3	
CVTSI2SD	xmm,m32	2	1	'	1		1			float	7.2	3	
CVT(T)SD2SI	r32,xmm	1	1		1		'			float	3+2	1	
CVT(T)SD2SI	r32,m64	1	1		1		1			float	0.2	1	
Arithmetic													
ADDSS/D SUBSS/D	xmm,xmm	1	1		1					float	3	1	
ADDSS/D SUBSS/D	xmm,m32/64	1	1		1		1			float		1	
ADDPS/D SUBPS/D	xmm,xmm	1	1		1					float	3	1	
ADDPS/D SUBPS/D	xmm,m128	1	1		1		1			float		1	

1	I		1 .	1		1	1	I				. 1
ADDSUBPS/D g)	xmm,xmm	1	1		1					float	3	1
ADDSUBPS/D g)	xmm,m128	1	1		1		1			float		1
HADDPS HSUBPS g)	xmm,xmm	3	3		1	2				float	5	2
HADDPS HSUBPS g)	xmm,m128	4	3		1	2	1			float		2
HADDPD HSUBPD g)	xmm,xmm	3	3		1	2				float	3	2
HADDPD HSUBPD g)	xmm,m128	4	3		1	2	1			float		2
MULSS MULPS	xmm,xmm	1	1	1						float	4	1
MULSS MULPS	xmm,m	1	1	1			1			float		1
MULSD MULPD	xmm,xmm	1	1	1						float	5	1
MULSD MULPD	xmm,m	1	1	1			1			float		1
DIVSS DIVPS	xmm,xmm	1	1	1						float	7-14	7-14
DIVSS DIVPS	xmm,m	1	1	1			1			float		7-14
DIVSD DIVPD	xmm,xmm	1	1	1						float	7-22	7-22
DIVSD DIVPD	xmm,m	1	1	1			1			float		7-22
RCPSS/PS	xmm,xmm	1	1		1		'			float	3	2
RCPSS/PS	xmm,m	1	1		1		1			float		2
CMPccSS/D CMPccPS/D	, Allini,iii	'	'		'		ļ '			lloat		_
CIVIF CC33/D CIVIF CCF3/D	xmm,xmm	1	1		1					float	3	1
CMPccSS/D CMPccPS/D	, , , , , , , , , , , , , , , , , , ,	'	'		'					lloat		•
CIVII CCGG/D CIVII CCI G/D	xmm,m	2	1		1		1			float		1
COMISS/D UCOMISS/D	xmm,xmm	1	1		1		-			float	1+2	1
COMISS/D UCOMISS/D	xmm,m32/64	1	1		1		1			float	1 . 2	1
MAXSS/D MINSS/D	xmm,xmm	1			1		١.			float	3	1
MAXSS/D MINSS/D	xmm,m32/64	1			1		1			float	5	1
MAXPS/D MINPS/D		1	1		1		'			float	3	1
	xmm,xmm	1	1		1		1			float	3	1
MAXPS/D MINPS/D	xmm,m128	l I	'				'			lloat		I
ROUNDSS/D ROUNDPS/	vmm vmm i	1	1		1					floot	3	1
D j)	xmm,xmm,i	l I	'							float	3	I
ROUNDSS/D ROUNDPS/	xmm,m128,i	2	1		1		1			float		1
D j) DPPS j)	xmm,xmm,i	4	4	1	2	1	'			float	11	2
	xmm,m128,i	6		-			1				11	2
DPPS j)	· '		5	X	X	X	'			float	0	4
DPPD j)	xmm,xmm,i	3	3	X	X	X	1			float	9	1
DPPD j)	xmm,m128,i	4	3	Х	Х	Х	1			float		3
Math										G	7.40	7.40
SQRTSS/PS	xmm,xmm	1	1	1						float	7-18	7-18
SQRTSS/PS	xmm,m	2	1	1			1			float	7.00	7-18
SQRTSD/PD	xmm,xmm	1	1	1						float	7-32	7-32
SQRTSD/PD	xmm,m	2	1	1			1			float		7-32
RSQRTSS/PS	xmm,xmm	1	1		1					float	3	2
RSQRTSS/PS	xmm,m	1	1		1		1			float		2
Logic												
AND/ANDN/OR/XORPS/D	xmm,xmm	1	1			1				float	1	1
AND/ANDN/OR/XORPS/D	xmm,m128	1	1			1	1			float		1
Other												
LDMXCSR	m32	6	6		x	v	1					5
	m32	2	1	X	^	1		1	1			5 1
STMXCSR	m4096	141	-			-	F	38	л 38		90	-
FXSAVE			141	X	X	X	5	ან	30		90	90
FXRSTOR Notes:	m4096	112	90	Х	Χ	Х	42					100

Notes:

g) SSE3 instruction set.

Intel Sandy Bridge

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm reg-

ister, mm/x = mmx or xmm register, y = 256 bit ymm register, same = same register for both operands. m = memory operand, m32 = 32-bit memory operand,

etc.

μops fused domain: The number of μops at the decode, rename, allocate and retirement stages in

the pipeline. Fused µops count as one.

μops unfused domain: The number of μops for each execution port. Fused μops count as two. Fused

macro-ops count as one. The instruction has μ op fusion if the sum of the numbers listed under p015 + p23 + p4 exceeds the number listed under μ ops fused domain. A number indicated as 1+ under a read or write port means a 256-bit read or write operation using two clock cycles for handling 128 bits each cycle. The port cannot receive another read or write μ op in the second clock cycle, but a read port can receive an address-calculation μ op in the second clock cycle. An x under p0, p1 or p5 means that at least one of the μ ops listed under p015 can optionally go to this port. For example, a 1 under p015 and an x under p0 and p5 means one μ op which can go to either port 0 or port 5, whichever is vacant first. A value listed under p015 but nothing under p0, p1 and p5 means that

it is not known which of the three ports these µops go to.

p015: The total number of μops going to port 0, 1 and 5.
p0: The number of μops going to port 0 (execution units).
p1: The number of μops going to port 1 (execution units).
p5: The number of μops going to port 5 (execution units).

p23: The number of μops going to port 2 or 3 (memory read or address calculation).

p4: The number of μops going to port 4 (memory write data).

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Where hyperthreading is enabled, the use of the same execution units in the other thread leads to inferior performance. Denormal numbers, NAN's and infinity do not increase the latency. The time unit used is core clock cycles, not the reference clock cycles given by the

time stamp counter.

Reciprocal throughput: The average number of core clock cycles per instruction for a series of inde-

pendent instructions of the same kind in the same thread.

The latencies and throughputs listed below for addition and multiplication using full size YMM registers are obtained only after a warm-up period of a thousand instructions or more. The latencies may be one or two clock cycles longer and the reciprocal throughputs double the values for shorter sequences of code.

There is no warm-up effect when vectors are 128 bits wide or less.

Integer instructions

Instruction	Operands	μops						Reci-	Com-		
		fused do- main	p015	p0	p1	р5	p23	p4		procal through- put	ments
Move instructions											
MOV	r,r/i	1	1	Х	Х	Х			1		
MOV	r,m	1					1		2	0.5	all ad- dressing modes

			,								
MOV	m,r	1					1	1	3	1	
MOV	m,i	1					1	1		1	
MOVNTI	m,r	2					1	1	~350	1	
MOVSX MOVZX	r,r	1	1	×	Х	Х			1		
MOVSXD	- ,-		•	^		, ,			•		
MOVSX MOVZX	r,m	1					1			0.5	
MOVSXD	,,									0.0	
CMOVcc	r,r	2	2	X	Х	х			2	1	
CMOVcc	r,m	2	2	X	X	Х	1		_	1	
XCHG	r,r	3	3	X	X	X			2	1	
XCHG		8	X	^	^	^	2	1	25	'	implicit
ACITO	r,m	0	^				_	'	23		lock
XLAT		3	2				1		7	1	IOCK
PUSH	r	1	_				1	1	3	1	
	r :								3		
PUSH	i 	1					1	1		1	
PUSH	m	2	_				2	1		1	
PUSHF(D/Q)		3	2	X	Х	Х	1	1		1	
PUSHA(D)		16	0				8	8		8	not 64 bit
POP	r	1					1		2	0.5	
POP	(E/R)SP	1	0				1			0.5	
POP	m	2					2	1		1	
POPF(D/Q)		9	8	X	х	Х	1			18	
POPA(D)		18	10				8			9	not 64 bit
LAHF SAHF		1	1						1	1	
SALC		3	3						1	1	not 64 bit
LEA	r,m	1	1	X	х				1	0.5	simple
LEA	r,m	1	1	^	1				3	1	complex
	1,111	'	'		'				3	'	or rip rel-
											ative
BSWAP	r32	1	1		1				1	1	
BSWAP	r64	2	2		2				2	1	
PREFETCHNTA	m	1	_		_		1		_	0.5	
PREFETCHT0/1/2		1					1			0.5	
	m										
LFENCE		2					1	1		4	
MFENCE		3	1				1	1		33	
SFENCE		2					1	1		6	
Arithmetic instructions											
ADD SUB	r,r/i	1	1	X	Х	Х			1		
ADD SUB	r,m	1	1	X	Х	Х	1			0.5	
ADD SUB	m,r/i	2	1	X	Х	Х	2	1	6	1	
SUB	r,same	1	0						0	0.25	
ADC SBB	r,r/i	2	2	X	х	Х			2	1	
ADC SBB	r,m	2	2	x	х	х	1		2	1	
ADC SBB	m,r/i	4	3	x	х	х	2	1	7	1.5	
CMP	r,r/i	1	1	×	х	х			1		
CMP	m,r/i	1	1	X	Х	Х	1		1	0.5	
INC DEC NEG NOT	r	1	1	X	X	Х			1	0.0	
INC DEC NEG NOT	m	3	1	X	X	X	2	1	6	2	
AAA AAS	""	2	2	^	^	^	_	'	4	_	not 64 bit
DAA DAS		3	3						4		not 64 bit
AAD		3	3						2		not 64 bit
AAM	_	8	8						20	11	not 64 bit
MUL IMUL	r8	1	1		1				3	1	

			,	J							
MUL IMUL	r16	4	4						4	2	
MUL IMUL	r32	3	3						4	2	
MUL IMUL	r64	2	2						3	1	
IMUL	r,r	1	1		1				3	1	
IMUL	r16,r16,i	2	2						4	1	
IMUL	r32,r32,i	1	1		1				3	1	
IMUL	r64,r64,i	1	1		1				3	1	
MUL IMUL	m8	1	1		1		1		3	1	
MUL IMUL	m16	4	3		_		1			2	
MUL IMUL	m32	3	2				1			2	
MUL IMUL	m64	2	1				1			2	
IMUL	r,m	1	1		1		1			1	
IMUL	r16,m16,i	2	2		ļ '		1			1	
IMUL	r32,m32,i	1	1		1		1			1	
IMUL	r64,m64,i	1	1		1		1			1	
DIV	r8	10	10		'		'		20-24	11-14	
DIV	r16	11	11						20-24	11-14	
DIV	r32	10	10						20-28	11-18	
DIV	r64	34-56	X						30-94	22-76	
IDIV	r8	10	10						21-24	11-14	
IDIV	r16	10	10						21-25	11-14	
IDIV	r32	9	9						20-27	11-18	
IDIV	r64	59-	Χ						40-103	25-84	
OD)A/		138								0.5	
CBW		1	1						1	0.5	
CWDE		1	1			1			1	1	
CDQE		1	1						1	0.5	
CWD		2	2						1	1	
CDQ		1	1						1	1	
CQO		1	1						1	0.5	
POPCNT	r,r	1	1		1				3	1	SSE4.2
POPCNT	r,m	1	1		1		1			1	SSE4.2
CRC32	r,r	1	1		1				3	1	SSE4.2
CRC32	r,m	1	1		1		1			1	SSE4.2
Logic instructions											
AND OR XOR	r,r/i	1	1	X	Х	Х			1		
AND OR XOR	r,m	1	1	X	Х	Х	1			0.5	
AND OR XOR	m,r/i	2	1	X	Х	Х	2	1	6	1	
XOR	r,same	1	0						0	0.25	
TEST	r,r/i	1	1	X	Х	Х			1		
TEST	m,r/i	1	1	X	Х	Х	1			0.5	
SHR SHL SAR	r,i	1	1	х		Х			1	0.5	
SHR SHL SAR	m,i	3	1				2	1		2	
SHR SHL SAR	r,cl	3	3						2	2	
SHR SHL SAR	m,cl	5	3				2	1		4	
ROR ROL	r,i	1	1						1	1	
ROR ROL	m,i	4	3				2	1		2	
ROR ROL	r,cl	3	3						2	2	
ROR ROL	m,cl	5	3				2	1		4	
RCR	r8,1	high	-						high	high	
RCR	r16/32/64,1	3	3						2	2	
RCR	r,i	8	8						5	5	
1 =	.,,	_	-	I	l	1	l	I	, ,	, ,	

RCR	m,i	11	7				х	Х		6	
RCR	r,cl	8	8						5	5	
RCR	m,cl	11	7				х	x		6	
RCL	r,1	3	3						2	2	
RCL	r,i	8	8						6	6	
RCL	1		7				.,			6	
	m,i	11					Х	Х		1	
RCL	r,cl	8	8						6	6	
RCL	m,cl	11	7				Х	Х		6	
SHRD SHLD	r,r,i	1	1							0.5	
SHRD SHLD	m,r,i	3					2	1		2	
SHRD SHLD	r,r,cl	4	4						2	2	
SHRD SHLD	m,r,cl	5	3				2	1	_	4	
BT	r,r/i	1	1				_	'	1	0.5	
BT	1		-				.,		'		
	m,r	10	8				X			5	
BT	m,i	2	1				1			0.5	
BTR BTS BTC	r,r/i	1	1						1	0.5	
BTR BTS BTC	m,r	11	7				Х	x		5	
BTR BTS BTC	m,i	3	1				2	1		2	
BSF BSR	r,r	1	1						3	1	
BSF BSR	r,m	1 1	1		1		1		_	1	
SETcc	r	1 1	1	x	'	х	'		1	0.5	
SETCC							1	4	'		
	m	2	1	Х		Х	1	1		1	
CLC		1	0							0.25	
STC CMC		1	1	X	Х	Х			1		
CLD STD		3	3							4	
Control transfer instruc	 tions										
JMP	short/near	1	1			1			0	2	
JMP	r	1	1			1			0	2	
JMP	m	1 1	1			1	1		0	2	
Conditional jump	short/near			1	1			1			1
		1 1	1			1			Λ	12	fact if not
, ,	Short/hear	1	1			1			0	1-2	fast if not
, .	snort/near	1	1			1			0	1-2	fast if not jumping
	Short/near										
Fused arithmetic and	SHOTTHEAL	1	1			1			0	1-2	
Fused arithmetic and branch		1	1			1				1-2	
Fused arithmetic and branch J(E/R)CXZ	short	1 2	1 2	x	x					1-2 2-4	
Fused arithmetic and branch J(E/R)CXZ LOOP	short short	1 2 7	1 2 7	x	x	1				1-2 2-4 5	
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E	short short short	1 2 7 11	1 2 7 11	x	x	1				1-2 2-4 5 5	
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL	short short	1 2 7 11 3	1 2 7	x	x	1	1	1		1-2 2-4 5 5 2	
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL	short short short	1 2 7 11	1 2 7 11	x	x	1	1 1	1 1		1-2 2-4 5 5 2	
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL	short short short near	1 2 7 11 3	1 2 7 11 2	x	x	1 1 1				1-2 2-4 5 5 2	
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL	short short short near r	1 2 7 11 3 2 3	1 2 7 11 2 1 2	x	x	1 1 1 1 1 1	1 2	1		1-2 2-4 5 5 2 2	
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL CALL	short short short near r m	1 2 7 11 3 2 3 2	1 2 7 11 2 1 2 2	x	x	1 1 1 1 1 1	1 2 1	1		1-2 2-4 5 5 2 2 2	
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL CALL RET RET	short short short near r m	1 2 7 11 3 2 3 2 3	1 2 7 11 2 1 2 2 2	х	x	1 1 1 1 1 1	1 2	1		1-2 2-4 5 5 2 2 2 2	jumping
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND	short short short near r m	1 2 7 11 3 2 3 2 3 15	1 2 7 11 2 1 2 2 2 13	х	X	1 1 1 1 1 1	1 2 1	1		1-2 2-4 5 5 2 2 2 2 2 7	jumping
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND	short short short near r m	1 2 7 11 3 2 3 2 3	1 2 7 11 2 1 2 2 2	x	х	1 1 1 1 1 1	1 2 1	1		1-2 2-4 5 5 2 2 2 2	jumping
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO	short short short near r m	1 2 7 11 3 2 3 2 3 15	1 2 7 11 2 1 2 2 2 13	х	X	1 1 1 1 1 1	1 2 1	1		1-2 2-4 5 5 2 2 2 2 2 7	jumping
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO	short short short near r m	1 2 7 11 3 2 3 2 3 15	1 2 7 11 2 1 2 2 2 13	х	x	1 1 1 1 1 1	1 2 1	1		1-2 2-4 5 5 2 2 2 2 2 7	jumping
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO String instructions LODS	short short short near r m	1 2 7 11 3 2 3 2 3 15 4 3	1 2 7 11 2 1 2 2 2 13 4	х	x	1 1 1 1 1 1	1 2 1 1	1	0	1-2 2-4 5 5 2 2 2 2 2 7 6	jumping
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO String instructions LODS REP LODS	short short short near r m	1 2 7 11 3 2 3 2 3 15 4	1 2 7 11 2 1 2 2 2 13 4	х	х	1 1 1 1 1 1	1 2 1 1	1 1		1-2 2-4 5 5 2 2 2 2 2 7 6	jumping
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO String instructions LODS REP LODS STOS	short short short near r m	1 2 7 11 3 2 3 2 3 15 4 3 5n+12 3	1 2 7 11 2 1 2 2 2 13 4	х	x	1 1 1 1 1 1	1 2 1 1	1	0 ~2n	1-2 2-4 5 5 2 2 2 2 2 7 6	not 64 bit
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO String instructions LODS REP LODS STOS	short short short near r m	1 2 7 11 3 2 3 2 3 15 4	1 2 7 11 2 1 2 2 2 13 4	x	x	1 1 1 1 1 1	1 2 1 1	1 1	0	1-2 2-4 5 5 2 2 2 2 2 7 6	not 64 bit not 64 bit worst
Fused arithmetic and branch J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO String instructions LODS REP LODS STOS REP STOS REP STOS	short short short near r m	1 2 7 11 3 2 3 2 3 15 4 3 5n+12 3	1 2 7 11 2 1 2 2 13 4	x	X	1 1 1 1 1 1	1 2 1 1	1 1	0 ~2n	1-2 2-4 5 5 2 2 2 2 2 7 6	not 64 bit

MOVS		5						4		ì
REP MOVS		2n					1.5 n		worst	ì
									case	ì
REP MOVS		3/16B					1/16B		best case	ì
CCAC								4		ì
SCAS		3					25.45	1		ı
REP SCAS CMPS		6n+47 5					2n+45	4		ı
REP CMPS		8n+80					2n+80	4		ı
REP CIVIPS		011700					211+00			ı
Other										ı
NOP (90)		1	0					0.25		ı
Long NOP (0F 1F)		1	0					0.25	decode	ı
									only 1	ı
									per clk	ı
PAUSE		7	7					11		ı
ENTER	a,0	12	10		2	1		8		ı
ENTER	a,b	49+6b					84+3b			ı
LEAVE		3	3		1			7		ı
CPUID		31-75					100-250			ì
RDTSC		21						28		ı
RDTSCP		23						36		ı
RDPMC		35						42		

Floating point x87 instructions

Instruction	Operands	μops	μορε	un	fus	ed c	loma	in	Latency	Reci-	Com-
	fused p0 do-main	p015	p0	p1	р5	p23	p4		procal through- put	ments	
Move instructions											
FLD	r	1	1	1					1	1	
FLD	m32/64	1	1				1		3	1	
FLD	m80	4	2	1	1		2		4	2	
FBLD	m80	43	40				3		45	21	
FST(P)	r	1	1	1					1	1	
FST(P)	m32/m64	1					1	1	4	1	
FSTP	m80	7	3				2	2	5	5	
FBSTP	m80	246								252	
FXCH	r	1	0						0	0.5	
FILD	m	1	1		1		1		6	1	
FIST(P)	m	3	1		1		1	1	7	2	
FISTTP	m	3	1		1		1	1	7	2	SSE3
FLDZ		1	1	1						2	
FLD1		2	2	1	1					2	
FLDPI FLDL2E etc.		2	2		2					2	
FCMOVcc	r	3	3						3	2	
FNSTSW	AX	2	2						2	1	
FNSTSW	m16	2	1				1	1		1	
FLDCW	m16	3	2				1		8		
FNSTCW	m16	2	1	1			1	1	5	1	
FINCSTP FDECSTP		1	1	1					1	1	
FFREE(P)	r	1	1							1	
FNSAVE	m	143								166	

FRSTOR	m	90							165	
Arithmetic instructions										
FADD(P) FSUB(R)(P)	r	1	1		1			3	1	
FADD(P) FSUB(R)(P)	m	2	2		1		1		1	
FMUL(P)	r	1	1	1				5	1	
FMUL(P)	m	1	1	1			1		1	
FDIV(R)(P)	r	1	1	1			-	10-24	10-24	
FDIV(R)(P)	m	1	1	1			1		10-24	
FABS		1	1	1			-	1	1	
FCHS		1	1	1				1	1	
FCOM(P) FUCOM	r	1	1		1			3	1	
FCOM(P) FUCOM	m .	1	1		1		1		1	
FCOMPP FUCOMPP		2	2	1	1				1	
FCOMI(P) FUCOMI(P)	r	3	3	1	1	1		4	1	
FIADD FISUB(R)	m '	2	2	'			1		1	
FIMUL	m	2	2	1	1		1		1	
FIDIV(R)	m m	2	2	1	1		1		'	
FICOM(P)	m	2	2	'	2		1		2	
FTST	'''	1	1		1		'		1	
FXAM		2	2		1				2	
FPREM		28	28		'			21	21	
FPREM1		41-87	20					26-50	26-50	
FRNDINT		17	17					20-30	20-50	
FRINDINI		17	17					22		
Math										
FSCALE		27	27					12		
FXTRACT		17	17					10		
FSQRT		1	1	1				10-24		
FSIN		64-100	х					47-100		
FCOS		20-110	Х					47-115		
FSINCOS		20-110	Х					43-123		
F2XM1		53-118	Х					61-69		
FYL2X										
FYL2XP1										
FPTAN		102	102					130		
FPATAN		28-91	X					93-146		
Other									_	
FNOP		1	1	1					1	
WAIT		2	2						1	
FNCLEX		5	5						22	
FNINIT		26	26						81	

Integer MMX and XMM instructions

Instruction	Operands	μops	μops	un	fuse	ed d	oma	in	Latency	Reci-	Com-
		fused do- main	p015	p0	p1	р5	p23	p4		procal through- put	ments
Move instructions											
MOVD	r32/64,mm/x	1	1	Х	Х	Х			1		
MOVD	m32/64,mm/x	1					1	1	3	1	

		- Juli	iay Di	luge							
MOVD	mm/x,r32/64	1	1	Х	Х	Х			1		
MOVD	mm/x,m32/64	1					1		3	0.5	
MOVQ	mm/x,mm/x	1	1	Х	Х	х			1		
MOVQ	mm/x,m64	1					1		3	0.5	
MOVQ	m64, mm/x	1					1	1	3	1	
MOVDQA	X,X	1	1	х	Х	х			1		
MOVDQA	x, m128	1					1		3	0.5	
MOVDQA	m128, x	1					1	1	3	1	
MOVDQU	x, m128	1	1				1		3	0.5	
MOVDQU	m128, x	1	1				1	1	3	1	
LDDQU	x, m128	1	1				1		3	0.5	SSE3
MOVDQ2Q	mm, x	2	2						1	1	
MOVQ2DQ	x,mm	1	1						1		
MOVNTQ	m64,mm	1					1	1	~300	1	
MOVNTDQ	m128,x	1					1	1	~300		
MOVNTDQA	x, m128	1					1	•		0.5	SSE4.1
PACKSSWB/DW	7,						•			0.0	552
PACKUSWB	mm,mm	1	1	1					1	1	
PACKSSWB/DW	,										
PACKUSWB	mm,m64	1	1	1			1				
PACKSSWB/DW											
PACKUSWB	X,X	1	1		Х	Х			1	0.5	
PACKSSWB/DW											
PACKUSWB	x,m128	1	1		Х	Х	1			0.5	
PACKUSDW	X,X	1	1		Х	Х			1	0.5	SSE4.1
PACKUSDW	x,m	1	1		Х	Х	1			0.5	SSE4.1
PUNPCKH/LBW/WD/DQ	mm/x,mm/x	1	1		Х	Х			1	0.5	
PUNPCKH/LBW/WD/DQ	mm/x,m	1	1		Х	Х	1			0.5	
PUNPCKH/LQDQ	X,X	1	1		Х	Х			1	0.5	
PUNPCKH/LQDQ	x, m128	2	1		Х	Х	1			0.5	
PMOVSX/ZXBW	X,X	1	1		Х	Х			1	0.5	SSE4.1
PMOVSX/ZXBW	x,m64	1	1		Х	х	1			0.5	SSE4.1
PMOVSX/ZXBD	x,x	1	1		Х	х			1	0.5	SSE4.1
PMOVSX/ZXBD	x,m32	1	1		Х	х	1			0.5	SSE4.1
PMOVSX/ZXBQ	x,x	1	1		Х	х			1	0.5	SSE4.1
PMOVSX/ZXBQ	x,m16	1	1		Х	х	1			0.5	SSE4.1
PMOVSX/ZXWD	X,X	1	1		Х	х			1	0.5	SSE4.1
PMOVSX/ZXWD	x,m64	1	1		Х	х	1			0.5	SSE4.1
PMOVSX/ZXWQ	X,X	1	1		Х	х			1	0.5	SSE4.1
PMOVSX/ZXWQ	x,m32	1	1		Х	х	1			0.5	SSE4.1
PMOVSX/ZXDQ	x,x	1	1		х	х			1	0.5	SSE4.1
PMOVSX/ZXDQ	x,m64	1	1		х	х	1			0.5	SSE4.1
PSHUFB	mm/x,mm/x	1	1		х	х			1	0.5	SSSE3
PSHUFB	mm/x,m	2	1		х	х	1			0.5	SSSE3
PSHUFW	mm,mm,i	1	1		х	х			1	0.5	
PSHUFW	mm,m64,i	2	1		Х	х	1			0.5	
PSHUFD	x,x,i	1	1		Х	Х			1	0.5	
PSHUFD	x,m128,i	2	1		Х	х	1			0.5	
PSHUFL/HW	, x,x,i	1	1		Х	Х			1	0.5	
PSHUFL/HW	x, m128,i	2	1		Х	Х	1			0.5	
PALIGNR	mm/x,mm/x,i	1	1		X	X			1	0.5	SSSE3
PALIGNR	mm/x,m,i	2	1		X	X	1		•	0.5	SSSE3
PBLENDVB	x,x,xmm0	2	2		1	1			2	1	SSE4.1
PBLENDVB	x,m,xmm0	3	2		1	1	1		_	1	SSE4.1
	, ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		_	1	1 .	١.'		1		•	00= 1.1

	1	i.				ı	1		1	ı	
PBLENDW	x,x,i	1	1		Х	Х			1	0.5	SSE4.1
PBLENDW	x,m,i	2	1		Х	Х	1			0.5	SSE4.1
MASKMOVQ	mm,mm	4	1	1			2	1		1	
MASKMOVDQU	X,X	10	4				4	х		6	
PMOVMSKB	r32,mm/x	1	1	1					2	1	
PEXTRB	r32,x,i	2	2	1	х	х			2	1	SSE4.1
PEXTRB	m8,x,i	2	1		х	х	1	1		1	SSE4.1
PEXTRW	r32,mm/x,i	2	2	1	х	х			2	1	
PEXTRW	m16,mm/x,i	2	1		х	х	1	1		2	SSE4.1
PEXTRD	r32,x,i	2	2	1	Х	х			2	1	SSE4.1
PEXTRD	m32,x,i	3	2	1	x	х	1	1		1	SSE4.1
PEXTRQ	r64,x,i	2	2	1	x	Х			2	1	SSE4.1,
PEXTRQ	m64,x,i	3	2	1	X	Х	1	1	_	1	64b
PINSRB	x,r32,i	2	2	.	X	Х			2	1	SSE4.1
PINSRB	x,m8,i	2	1		X	X	1		_	0.5	SSE4.1
PINSRW	mm/x,r32,i	2	2		X	X			2	1	332
PINSRW	mm/x,m16,i	2	1		X	X	1		_	0.5	
PINSRD	x,r32,i	2	2		x	X	'		2	1	SSE4.1
PINSRD	x,m32,i	2	1		x	X	1			0.5	SSE4.1
PINSRQ	x,r64,i	2	2		x	X	'		2	1	SSE4.1,
PINSRQ	x,104,1 x,m64,i	2	1		X		1			0.5	64 b
FINSKQ	X,11104,1	2	'		X	Х	1			0.5	04.0
A with we atic in atmosphic ma											
Arithmetic instructions		4	4			.,			4	0.5	
PADD/SUB(U,S)B/W/D/Q	mm/x, mm/x	1	1		X	X	,		1	0.5	
PADD/SUB(U,S)B/W/D/Q	mm/x,m	1	1		Х	Х	1			0.5	00050
PHADD/SUB(S)W/D	mm/x, mm/x	3	3		Х	Х			2	1.5	SSSE3
PHADD/SUB(S)W/D	mm/x,m64	4	3		Х	Х	1			1.5	SSSE3
PCMPEQ/GTB/W/D	mm/x,mm/x	1	1		Х	Х			1	0.5	
PCMPEQ/GTB/W/D	mm/x,m	1	1		Х	Х	1			0.5	
PCMPEQQ	X,X	1	1		X	Х			1	0.5	SSE4.1
PCMPEQQ	x,m128	1	1		Х	Х	1			0.5	SSE4.1
PCMPGTQ	X,X	1	1	1					5	1	SSE4.2
PCMPGTQ	x,m128	1	1	1			1			1	SSE4.2
PSUBxx, PCMPGTx	x,same	1	0						0	0.25	
PCMPEQx	x,same	1	1						0	0.5	
PMULL/HW PMULHUW	mm/x,mm/x	1	1	1					5	1	
PMULL/HW PMULHUW	mm/x,m	1	1	1			1			1	
PMULHRSW	mm/x,mm/x	1	1	1					5	1	SSSE3
PMULHRSW	mm/x,m	1	1	1			1			1	SSSE3
PMULLD	X,X	1	1	1					5	1	SSE4.1
PMULLD	x,m128	2	1	1			1			1	SSE4.1
PMULDQ	X,X	1	1	1					5	1	SSE4.1
PMULDQ	x,m128	1	1	1			1			1	SSE4.1
PMULUDQ	mm/x,mm/x	1	1	1					5	1	
PMULUDQ	mm/x,m	1	1	1			1			1	
PMADDWD	mm/x,mm/x	1	1	1					5	1	
PMADDWD	mm/x,m	1	1	1			1			1	
PMADDUBSW	mm/x,mm/x	1	1	1					5	1	SSSE3
PMADDUBSW	mm/x,m	1	1	1			1			1	SSSE3
PAVGB/W	mm/x,mm/x	1	1		x	Х			1	0.5	3-4
PAVGB/W	mm/x,m	1	1		X	X	1		·	0.5	
PMIN/MAXSB	X,X	1	1		X	X			1	0.5	SSE4.1
PMIN/MAXSB	x,m128	1	1		X	X	1		'	0.5	SSE4.1
	7,111120		1 '	1		^		1	I	1 5.5	JUL

PMIN/MAXUB PMIN/MAXUB PMIN/MAXSW PMIN/MAXSW PMIN/MAXUW PMIN/MAXUW PMIN/MAXU/SD PMIN/MAXU/SD PHMINPOSUW PHMINPOSUW PABSB/W/D PABSB/W/D PSIGNB/W/D PSIGNB/W/D PSADBW PSADBW	mm/x,mm/x mm/x,m mm/x,m mm/x,m/x mm/x,m x,x x,m x,x x,m128 x,x x,m128 mm/x,mm/x mm/x,m mm/x,m mm/x,m	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1 1 1 1	1 1 1 1 1	x x x x x x x x x x x x x x x x x x x	x x x x x x x x x x x x x x x x x x x	1 1 1 1 1 1	1 1 1 1 5 1 1 5	0.5 0.5 0.5 0.5 0.5 0.5 0.5 1 0.5 0.5 0.5 1	SSE4.1 SSE4.1 SSE4.1 SSE4.1 SSE4.1 SSE3 SSSE3 SSSE3 SSSE3
MPSADBW	· ·	3	3		1	1	'	6	1	SSE4.1
MPSADBW	x,x,i x,m,i	4	3	1	1	1	1	0	1	SSE4.1
Logic instructions PAND(N) POR PXOR PAND(N) POR PXOR PXOR PTEST PTEST PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q	mm/x,mm/x mm/x,m x,same x,x x,m128 mm,mm/i mm,m64 x,i x,x x,m128 x,i	1 1 1 1 1 1 1 2 3 1	1 1 0 2 2 1 1 1 2 2	X X 1 1 1 1 1 1 1 1 1 1	x x x x	x x x x	1 1 1	1 0 1 1 2	0.5 0.25 1 1 2 1 1 1	SSE4.1 SSE4.1
String instructions										
PCMPESTRI PCMPESTRI PCMPESTRM PCMPESTRM PCMPISTRI PCMPISTRI PCMPISTRI PCMPISTRM PCMPISTRM	x,x,i x,m128,i x,x,i x,m128,i x,x,i x,m128,i x,x,i x,m128,i	8 8 8 8 3 4 3 4	8 7 8 7 3 3 3				1 1 1 1	4 11-12 3 11	4 4 4 3 3 3 3	SSE4.2 SSE4.2 SSE4.2 SSE4.2 SSE4.2 SSE4.2 SSE4.2 SSE4.2
Encryption instructions										
PCLMULQDQ AESDEC, AESDECLAST, AESENC, AESENCLAST	x,x,i	18	18					14	8	CLMUL
	X,X	2	2					8	4	AES
AESIMC	x,x	2	2						2	AES
AESKEYGENASSIST	x,x,i	11	11					8	8	AES
Other EMMS		31	31						18	

Floating point XMM and YMM instructions

Instruction	Operands	μops	μops	un	fus	ed d	oma	in	Latency	Reci-	Com-
	•	fused do- main	p015							procal through- put	ments
Move instructions											
MOVAPS/D	X,X	1	1			1			1	1	
VMOVAPS/D	y,y	1	1			1			1	1	AVX
MOVAPS/D MOVUPS/D	x,m128	1					1		3	0.5	
VMOVAPS/D VMOVUPS/											
D	y,m256	1					1+		4	1	AVX
MOVAPS/D MOVUPS/D	m128,x	1					1	1	3	1	
VMOVAPS/D VMOVUPS/											
D	m256,y	1					1	1+	3	1	AVX
MOVSS/D	X,X	1	1			1			1	1	
MOVSS/D	x,m32/64	1					1		3	0.5	
MOVSS/D	m32/64,x	1					1	1	3	1	
MOVHPS/D MOVLPS/D	x,m64	1	1			1	1		3	1	
MOVH/LPS/D	m64,x	1					1	1	3	1	
MOVLHPS MOVHLPS	X,X	1	1			1			1	1	
MOVMSKPS/D	r32,x	1	1	1					2	1	
VMOVMSKPS/D	r32,y	1	1	1					2	1	
MOVNTPS/D	m128,x	1					1	1	~300	1	
VMOVNTPS/D	m256,y	1					1	4	~300	25	AVX
SHUFPS/D	x,x,i	1	1			1			1	1	
SHUFPS/D	x,m128,i	2	1			1	1			1	
VSHUFPS/D	y,y,y,i	1	1			1			1	1	AVX
VSHUFPS/D	y, y,m256,i	2	1			1	1+			1	AVX
VPERMILPS/PD	x,x,x/i	1	1			1	-		1	1	AVX
VPERMILPS/PD	y,y,y/i	1	1			1			1	1	AVX
VPERMILPS/PD	x,x,m	2	1			1	1			1	AVX
VPERMILPS/PD	y,y,m	2				1	1+			1	AVX
VPERMILPS/PD	x,m,i	2	1			1	1			1	AVX
VPERMILPS/PD	y,m,i	2	1			1	1+			1	AVX
VPERM2F128	1	1				1	١.,		2	1 1	AVX
VPERM2F128	y,y,y,i	2				1	1+			1	AVX
BLENDPS/PD	y,y,m,i	1	1			-	1 +		1	0.5	SSE4.
BLENDPS/PD	x,x,i x,m128,i	2		X		X	1		'	0.5	SSE4.
VBLENDPS/PD			1	X		X	'		1		AVX
	y,y,i	1 2	1	X		X	4.		1	1	
VBLENDPS/PD	y,m256,i	2	1	X		X	1+			1	AVX
BLENDVPS/PD	x,x,xmm0		2	X		X	4		2	1	SSE4.
BLENDVPS/PD	x,m,xmm0	3	2	X		X	1			1	SSE4.
VBLENDVPS/PD	y,y,y,y	2	2	X		X			2	1	AVX
VBLENDVPS/PD	y,y,m,y	3	2	Х		X	1+			1	AVX
MOVDDUP	X,X	1	1			1			1	1	SSE3
MOVDDUP	x,m64	1					1		3	0.5	SSE3
VMOVDDUP	y,y	1	1			1			1	1	AVX
VMOVDDUP	y,m256	1					1+		3	1	AVX
VBROADCASTSS	x,m32	1					1			1	AVX
VBROADCASTSS	y,m32	2	1			1	1			1	AVX
VBROADCASTSD	y,m64	2	1			1	1			1	AVX

			,	9							
VBROADCASTF128	y,m128	2	1			1	1			1	AVX
MOVSH/LDUP	X,X	1	1			1			1	1	SSE3
MOVSH/LDUP	x,m128	1					1		3	0.5	SSE3
VMOVSH/LDUP	y,y	1	1			1			1	1	AVX
VMOVSH/LDUP	y,m256	1					1+		4	1	AVX
UNPCKH/LPS/D	x,x	1	1			1			1	1	SSE3
UNPCKH/LPS/D	x,m128	1	1			1	1		•	1	SSE3
VUNPCKH/LPS/D	y,y,y	1				1	'		1	1 1	AVX
VUNPCKH/LPS/D	y,y,y y,y,m256	1	1			1	1+		1		AVX
EXTRACTPS		2	2	1		-	١.,		2	1	SSE4.1
	r32,x,i			'		1	4	,	2	1	
EXTRACTPS	m32,x,i	3	2			1	1	1	0	1	SSE4.1
VEXTRACTF128	x,y,i	1	1			1		,	2	1	AVX
VEXTRACTF128	m128,y,i	2	1				1	1		1	AVX
INSERTPS	x,x,i	1	1			1			1	1	SSE4.1
INSERTPS	x,m32,i	2	1			1	1			1	SSE4.1
VINSERTF128	y,y,x,i	1	1			1			2	1	AVX
VINSERTF128	y,y,m128,i	2	1			1	1			1	AVX
VMASKMOVPS/D	x,x,m128	3	2				1			1	AVX
VMASKMOVPS/D	y,y,m256	3	2				1+			1	AVX
VMASKMOVPS/D	m128,x,x	4	2				1	1		1	AVX
VMASKMOVPS/D	m256,y,y	4	2				1	1+		2	AVX
Conversion											
CVTPD2PS	X,X	2	2		1	1			4	1	
CVTPD2PS	x,m128	2	2		1	1	1		4	1 1	
VCVTPD2PS		2	2		1	1	'		4	1	AVX
VCVTPD2PS	x,y x,m256	2	2		1	1	1+		4	1	AVX
	· ·					1	ĮΤ		2	1 1	AVA
CVTSD2SS	X,X	2	2		1	1	_		3		
CVTSD2SS	x,m64	2	2		1	1	1		0		
CVTPS2PD	X,X	2	2	1		1			3	1	
CVTPS2PD	x,m64	2	1	1			1			1	A
VCVTPS2PD	y,x	2	2	1		1			4	1	AVX
VCVTPS2PD	y,m128	3	2	1		1	1		_	1	AVX
CVTSS2SD	X,X	2	2	1		1			3	1	
CVTSS2SD	x,m32	2	1	1			1			1	
CVTDQ2PS	X,X	1	1		1				3	1	
CVTDQ2PS	x,m128	1	1		1		1			1	
VCVTDQ2PS	y,y	1	1		1				3	1	AVX
VCVTDQ2PS	y,m256	1	1		1		1+			1	AVX
CVT(T) PS2DQ	X,X	1	1		1				3	1	
CVT(T) PS2DQ	x,m128	1	1		1		1			1	
VCVT(T) PS2DQ	y,y	1	1		1				3	1	AVX
VCVT(T) PS2DQ	y,m256	1	1		1		1+			1	AVX
CVTDQ2PD	X,X	2	2		1	1			4	1	
CVTDQ2PD	x,m64	2	2		1	1	1			1	
VCVTDQ2PD	y,x	2	2		1	1			5	1	AVX
VCVTDQ2PD	y,m128	3	2		1	1	1			1	AVX
CVT(T)PD2DQ	X,X	2	2		1	1			4	1	
CVT(T)PD2DQ	x,m128	2	2		1	1	1			1	
VCVT(T)PD2DQ	x,y	2	2		1	1			5	1	AVX
VCVT(T)PD2DQ	x,m256	2	2		1	1	1+			1	AVX
CVTPI2PS	x,mm	1	1		1				4	2	
CVTPI2PS	x,m64	1	1		1		1			2	
	-					*		. '		*	

CVT(T)PS2PI	mm,x	2	2		1	1		4	1	
CVT(T)PS2PI	mm,m128	2	1		1		1		1	
CVTPI2PD	x,mm	2	2		1	1		4	1	
CVTPI2PD	x,m64	2	2		1	1	1		1	
CVT(T) PD2PI	mm,x	2	2		ļ '	'		4	1	
, ,		2	2				1	4	1	
CVT(T) PD2PI	mm,m128						'			
CVTSI2SS	x,r32	2	2		1	1		4	1.5	
CVTSI2SS	x,m32	1	1		1		1		1.5	
CVT(T)SS2SI	r32,x	2	2	1	1			4	1	
CVT(T)SS2SI	r32,m32	2	2		1		1		1	
CVTSI2SD	x,r32	2	2	1	1			4	1.5	
CVTSI2SD	x,m32	1	1		1		1		1.5	
CVT(T)SD2SI	r32,x	2	2	1	1			4	1	
CVT(T)SD2SI	r32,m64	2	2	1	1		1	•	1	
0 1 (1)00201	102,11104	_	_	'	ļ '				'	
Arithmetic										
ADDSS/D SUBSS/D		1	1		1			3	1	
	X,X				-			٥		
ADDSS/D SUBSS/D	x,m32/64	1	1		1		1		1	
ADDPS/D SUBPS/D	X,X	1	1		1			3	1	
ADDPS/D SUBPS/D	x,m128	1	1		1		1		1	
VADDPS/D VSUBPS/D	y,y,y	1	1		1			3	1	AVX
VADDPS/D VSUBPS/D	y,y,m256	1	1		1		1+		1	AVX
ADDSUBPS/D	x,x	1	1		1			3	1	SSE3
ADDSUBPS/D	x,m128	1	1		1		1		1	SSE3
VADDSUBPS/D	y,y,y	1	1		1			3	1	AVX
VADDSUBPS/D	y,y,m256	1	1		1		1+		1	AVX
HADDPS/D HSUBPS/D	x,x	3	3		1	2	'	5	2	SSE3
HADDPS/D HSUBPS/D		4	3		1	2	1	3	2	SSE3
	x,m128	4	٥		'	_	'			SSES
VHADDPS/D VHSUBPS/	V V V	3	3		1	2		5	2	AVX
D	y,y,y	3	٥		'	_		5		AVA
VHADDPS/D VHSUBPS/	V V m256	4	9		1	2	4.		2	AVX
D	y,y,m256	4	3	,	ı	2	1+	_	2	AVA
MULSS MULPS	X,X	1	1	1				5	1	
MULSS MULPS	x,m	1	1	1			1		1	
VMULPS	y,y,y	1	1	1				5	1	AVX
VMULPS	y,y,m256	1	1	1			1+		1	AVX
MULSD MULPD	X,X	1	1	1				5	1	
MULSD MULPD	x,m	1	1	1			1		1	
VMULPD	y,y,y	1	1	1				5	1	AVX
VMULPD	y,y,m256	1	1	1			1+		1	AVX
DIVSS DIVPS	x,x	1	1	1				10-14	10-14	
DIVSS DIVPS	x,m	1	1	1			1		10-14	
VDIVPS		3	3	2		1		21-29	20-28	AVX
VDIVPS	y,y,y		3	2		1	1+	21-29	20-28	AVX
	y,y,m256	4				'	'	40.00		AVA
DIVSD DIVPD	x,x	1	1	1				10-22	10-22	
DIVSD DIVPD	x,m	1	1	1			1		10-22	
VDIVPD	y,y,y	3	3	2		1		21-45	20-44	AVX
VDIVPD	y,y,m256	4	3	2		1	1+		20-44	AVX
RCPSS/PS	x,x	1	1	1				5	1	
RCPSS/PS	x,m128	1	1	1			1		1	
VRCPPS	y,y	3	3	2		1		7	2	AVX
VRCPPS	y,m256	4	3				1+		2	AVX
1	,,====	•	1	I	l	1	1 1	ı	_	

			,	J						
CMPccSS/D CMPccPS/D	V V	1	1		1			3	1	
CMPccSS/D CMPccPS/D	X,X	ı	'		1			S	I	
	x,m128	2	1		1		1		1	
VCMPccPS/D	y,y,y	1	1		1			3	1	AVX
VCMPccPS/D	y,y,m256	2	1		1		1+		1	AVX
COMISS/D UCOMISS/D	X,X	2	2	1	1			2	1	
COMISS/D UCOMISS/D	x,m32/64	2	2	1	1		1		1	
MAXSS/D MINSS/D	x,x	1	1		1			3	1	
MAXSS/D MINSS/D	x,m32/64	1	1		1		1		1	
MAXPS/D MINPS/D	x,x	1	1		1			3	1	
MAXPS/D MINPS/D	x,m128	1	1		1		1		1	
VMAXPS/D VMINPS/D	y,y,y	1	1		1			3	1	AVX
VMAXPS/D VMINPS/D	y,y,m256	1	1		1		1+		1	AVX
ROUNDSS/SD/PS/PD	x,x,i	1	1		1			3	1	SSE4.1
ROUNDSS/SD/PS/PD	x,m128,i	2	1		1		1		1	SSE4.1
VROUNDSS/SD/PS/PD	y,y,i	1	1		1			3	1	AVX
VROUNDSS/SD/PS/PD	y,m256,i	2	1		1		1+		1	AVX
DPPS	x,x,i	4	4	1	2	1		12	2	SSE4.1
DPPS	x,m128,i	6	5				1		4	SSE4.1
VDPPS	y,y,y,i	4	4	1	2	1		12	2	AVX
VDPPS	y,m256,i	6	5				1+		4	AVX
DPPD	x,x,i	3	3	1	1	1		9	2	SSE4.1
DPPD	x,m128,i	4	3				1		2	SSE4.1
Math										
SQRTSS/PS	X,X	1	1	1				10-14	10-14	
SQRTSS/PS	x,m128	1	1	1			1		10-14	
VSQRTPS	у,у	3	3						21-28	AVX
VSQRTPS	y,m256	4	3				1+		21-28	AVX
SQRTSD/PD	X,X	1	1	1				10-21	10-21	
SQRTSD/PD	x,m128	2	1	1			1		10-21	
VSQRTPD	у,у	3	3					21-43	21-43	AVX
VSQRTPD	y,m256	4	3				1+		21-43	AVX
RSQRTSS/PS	X,X	1	1	1				5	1	
RSQRTSS/PS	x,m128	1	1	1			1		1	
VRSQRTPS	y,y	3	3					7	2	AVX
VRSQRTPS	y,m256	4	3				1+		2	AVX
Logic										
AND/ANDN/OR/XORPS/PD	X,X	1	1			1		1	1	
AND/ANDN/OR/XORPS/PD	x,m128	1	1			1	1		1	
VAND/ANDN/OR/XORPS/		,						,		A > 0.4
PD	y,y,y	1	1			1		1	1	AVX
VAND/ANDN/OR/XORPS/	v v m256	1	4			1	1_		4	A\/\
PD (V)XORPS/PD	y,y,m256 x/y,x/y,same	1 1	0			1	1+	0	0.25	AVX
(V)XORF3/FD	x/y,x/y,same	'	0					U	0.23	
Other										
VZEROUPPER		4						2	1	AVX
VZEROALL		12							11	AVX, 32 bit
		12							''	AVX,
VZEROALL		20							9	64 bit

LDMXCSR	m32	3	3			1			3		
STMXCSR	m32	3	3	1	1	1	1		1		
VSTMXCSR	m32	3	3	1	1	1	1		1	AVX	
FXSAVE	m4096	130							68		
FXRSTOR	m4096	116							72		
XSAVEOPT	m	100-16	1					60-500			

Intel Ivy Bridge

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm

register, mm/x = mmx or xmm register, y = 256 bit ymm register, same = same register for both operands. m = memory operand, m32 = 32-bit memory oper-

and, etc.

μops fused domain: The number of μops at the decode, rename, allocate and retirement stages in

the pipeline. Fused µops count as one.

μορs unfused domain: The number of μορs for each execution port. Fused μορs count as two. Fused

macro-ops count as one. The instruction has μ op fusion if the sum of the numbers listed under p015 + p23 + p4 exceeds the number listed under μ ops fused domain. A number indicated as 1+ under a read or write port means a 256-bit read or write operation using two clock cycles for handling 128 bits each cycle. The port cannot receive another read or write μ op in the second clock cycle, but a read port can receive an address-calculation μ op in the second clock cycle. An x under p0, p1 or p5 means that at least one of the μ ops listed under p015 can optionally go to this port. For example, a 1 under p015 and an x under p0 and p5 means one μ op which can go to either port 0 or port 5, whichever is vacant first. A value listed under p015 but nothing under p0, p1 and p5 means that it is not known which of the three ports these μ ops go to.

p015: The total number of μops going to port 0, 1 and 5.
p0: The number of μops going to port 0 (execution units).
p1: The number of μops going to port 1 (execution units).
p5: The number of μops going to port 5 (execution units).

p23: The number of μops going to port 2 or 3 (memory read or address calculation).

p4: The number of μops going to port 4 (memory write data).

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Where hyperthreading is enabled, the use of the same execution units in the other thread leads to inferior performance. Denormal numbers, NAN's and infinity do not increase the latency. The time unit used is core clock cycles, not the reference clock cycles given by

the time stamp counter.

Reciprocal throughput: The average number of core clock cycles per instruction for a series of inde-

pendent instructions of the same kind in the same thread.

The latencies and throughputs listed below for addition and multiplication using full size YMM registers are obtained only after a warm-up period of a thousand instructions or more. The latencies may be one or two clock cycles longer and the reciprocal throughputs double the values for shorter sequences of code.

There is no warm-up effect when vectors are 128 bits wide or less.

Integer instructions

Instruction	Operands	μοps fused	μops	μορs unfused domain				in	Latency	Reci- procal	Com- ments
		do- main	p015	p0	p1	р5	p23	p4	1	through- put	
Move instructions											
MOV	r,i	1	1	Х	х	Х			1	0.33	
MOV	r8/16,r8/16	1	1	X	Х	X			1	0.33	

			,	0							
MOV	r32/64,r32/64	1	1	x	х	х			0-1	0.25	may be elimin.
MOV	r8/16,m8/16	1	1	x	x	х	1		2	0.5	ellitilit.
MOV	r32/64,m32/64	1		^	^	^	1		2	0.5	
MOV	r,m	1					1		2	1	64 b abs
	,,,,,	•					-		_	•	address
MOV	m,r	1					1	1	3	1	
MOV	m,i	1					1	1		1	
MOVNTI	m,r	2					1	1	~340	1	
MOVSX MOVSXD	r,r	1	1	x	Х	х			1	0.33	
MOVZX	r16,r8	1	1	Х	Х	Х			1	0.33	
MOVZX	r32/64,r8	1	1	х	Х	Х			0-1	0.25	may be
MOVZV	"20/C4 "40	4	,						_	0.00	elimin.
MOVZX	r32/64,r16	1	1	Х	Х	Х			1	0.33	
MOVSX MOVZX	r16,m8	2	1	Х	Х	Х	1		3	0.5	
MOVSX MOVZX MOVSXD	r32/64,m	1					1		2	0.5	
CMOVcc	rr	2	2	x	x	х			2	0.67	
CMOVCC	r,r r,m	2	2	X	X	X	1			~0.8	
XCHG	r,r	3	3	x	X	X	'		2	1	
XCHG	r,m	7	X	^	^	^	2	3	25	'	implicit
XONO	,,,,,	,	_ ^				_		20		lock
XLAT		3	2				1		7	1	
PUSH	r	1					1	1	3	1	
PUSH	i	1					1	1		1	
PUSH	m	2					2	1		1	
PUSH	(E/R)SP	2	1	х	Х	Х	1	1	3	1	
PUSHF(D/Q)		3	2	х	Х	Х	1	1		1	
PUSHA(D)		19	3	Х	Х	Х	8	8		8	not 64 bit
POP	r	1					1		2	0.5	
POP	(E/R)SP	3	2	х	Х	Х	1			0.5	
POP	m	2					2	1		1	
POPF(D/Q)		9	8	Х	Х	Х	1			18	
POPA(D)		18	10	Х	Х	Х	8			9	not 64 bit
LAHF SAHF		1	1	Х		Х			1	1	
SALC		3	3	Х	Х	Х			1	1	not 64 bit
LEA	r16,m	2	2	Х	1	Х			2-4	1	
LEA	r32/64,m	1	1	X	Х				1	0.5	1-2 com- ponents
LEA	r32/64,m	1	1		1				3	1	3 com-
	102/04,111		'		'					'	ponents
											or RIP
BSWAP	r32	1	1		1				1	1	
BSWAP	r64	2	2	х	1	Х			2	1	
PREFETCHNTA	m	1					1			43	
PREFETCHT0/1/2	m	1					1			43	
LFENCE		2								4	
MFENCE		3					1	1		36	
SFENCE		2					1	1		6	
Arithmetic instructions											
ADD SUB	r,r/i	1	1	x	x	х			1	0.33	
ADD SUB	r,m	1	1	X	X	X	1		'	0.55	
ADD SUB	m,r/i	2	1	x	X	X	2	1	6	1	
,	,	_		^	1 ^	· ^	ı -		, ,	'	1

ADC SBB	
ADC SBB CMP CMP CMP CMP M,r/i I 1 1 X X X X I 1 0.33 CMP INC DEC NEG NOT INC DEC NEG NOT M 3 1 X X X X 2 1 6 1 AAA AAS DAA DAS AAD AAM MUL IMUL MUL IMUL MUL IMUL T16 MUL IMUL T16,r16,i Z 2 Z X 1 X X 2 1 6 1 T 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	I
CMP r,r/i 1 1 x </td <td></td>	
CMP r,r/i 1 1 x </td <td></td>	
CMP m,r/i 1 1 x x x x 1 1 0.5 INC DEC NEG NOT r 1 1 x x x x 1 0.33 INC DEC NEG NOT m 3 1 x x x 2 1 6 1 AAA AAS DAA DAS 3 3 3 4 4 not 64 AAD 3 3 3 2 2 not 64 AAM 8 8 8 20 8 not 64 MUL IMUL r16 4 4 4 2 4 2 MUL IMUL r64 2 2 3 3 4 2 MUL IMUL r64 2 2 3 1 3 1 MUL IMUL r64 2 2 3 1 3 1 IMUL r76 1 1 1	
INC DEC NEG NOT	
INC DEC NEG NOT AAA AAS DAA DAS AAD AAM MUL IMUL M	
AAA AAS 2 2 x 1 x 4 not 64 DAA DAS 3 3 3 3 4 not 64 AAD 8 8 8 20 8 not 64 AAM 8 8 8 20 8 not 64 MUL IMUL r8 1 1 1 3 1 MUL IMUL r16 4 4 4 2 MUL IMUL r32 3 3 4 2 MUL IMUL r64 2 2 3 1 IMUL r,r 1 1 3 1 IMUL r16,r16,i 2 2 4 1	
DAA DAS AAD AAM MUL IMUL REA MUL IMUL REA MUL IMUL REA	bit
AAD AAM MUL IMUL r8 1 1 1 3 3 1 4 2 8 8 not 64 NUL IMUL r16 4 4 4 7 MUL IMUL r32 3 3 4 2 MUL IMUL r32 3 3 1 MUL IMUL r64 2 2 3 1 IMUL r16,r16,i 2 2 1 1 1 1 3 1 1 3 1 1 1 1 1 1 1 1 1 1	bit
AAM r8 1 1 1 3 1 1 1 3 1 1 1 1 3 1 1 1 4 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 3 1 1 1 3 1 1 1 3 1 1 1 1 3 1	bit
MUL IMUL r8 1 1 1 3 1 MUL IMUL r16 4 4 4 2 MUL IMUL r32 3 3 4 2 MUL IMUL r64 2 2 3 1 IMUL r,r 1 1 1 3 1 IMUL r16,r16,i 2 2 4 1	
MUL IMUL r16 4 4 4 2 MUL IMUL r32 3 3 4 2 MUL IMUL r64 2 2 3 1 IMUL r,r 1 1 1 3 1 IMUL r16,r16,i 2 2 4 1	
MUL IMUL r32 3 3 4 2 MUL IMUL r64 2 2 3 1 IMUL r,r 1 1 1 3 1 IMUL r16,r16,i 2 2 4 1	
MUL IMUL r64 2 2 3 1 IMUL r,r 1 1 1 3 1 IMUL r16,r16,i 2 2 4 1	
IMUL r,r 1 1 1 3 1 IMUL r16,r16,i 2 2 4 1	
IMUL r16,r16,i 2 2 4 1	
IMUL r64,r64,i 1 1 1 3 1	
MUL IMUL m8 1 1 1 1 3 1	
MUL IMUL m16 4 3 1 1 2	
MUL IMUL m32 3 2 1 2	
MUL IMUL	
MUL r64,m64,i 1 1 1 1 1 1 1 1 1	
DIV r64 35-57 x 29-94 22-76	
IDIV r8 11 11 20-23 8	
IDIV r16 11 11 20-24 8	
IDIV r32 9 9 1 19-26 8-11	
IDIV r64 59- x 28-103 26-88	
CWD	
CQO	
	2
CRC32 r,m 1 1 1 1 SSE4	.2
Logic instructions	
AND OR XOR r,r/i 1 1 x x x 1 1 0.33	
AND OR XOR r,m 1 1 x x x 1 0.5	
AND OR XOR m,r/i 2 1 x x x 2 1 6 1	- 1
TEST r,r/i 1 1 x x x x 1 0.33	
TEST m,r/i 1 1 x x x 1 0.5	

ROR ROL ROR ROL	r,i m,i	1 4	3	X		Х	2	1	1	0.5 2	
ROR ROL	m,ı r,cl	2	2	X		Х	2	1	1	1	
ROR ROL	m,cl	5	3				2	1	-	4	
RCL RCR	r,1	3	3	х	х	Х			2	2	
RCL RCR	r,i	8	8	х	х	Х			5	5	
RCL RCR	m,i	11	8	х	х	х	2	1		6	
RCL RCR	r,cl	8	8	X	Х	Х			5	5	
RCL RCR	m,cl	11	8	х	х	х	2	1		6	
SHRD SHLD	r,r,i	1	1	х		х			1	0.5	
SHRD SHLD	m,r,i	3	3	х	х	х	2	1		2	
SHRD SHLD	r,r,cl	4	4	х	1	х			2	2	
SHRD SHLD	m,r,cl	5	4	X	1	х	2	1		4	
ВТ	r,r/i	1	1	х		х			1	0.5	
ВТ	m,r	10	9	х	х	х	1			5	
ВТ	m,i	2	1	х		Х	1			0.5	
BTR BTS BTC	r,r/i	1	1	х		Х			1	0.5	
BTR BTS BTC	m,r	11	8	х	Х	Х	2	1		5	
BTR BTS BTC	m,i	3	2	Х		Х	1	1		2	
BSF BSR	r,r	1	1		1				3	1	
BSF BSR	r,m	1	1		1		1			1	
SETcc	r	1	1	х		х			1	0.5	
SETcc	m	2	1	Х		Х	1	1		1	
CLC		1	0							0.25	
STC CMC		1	1	Х	Х	Х			1	0.33	
CLD STD		3	3	Х	Х	Х				4	
Control transfer instructi											
JMP	short/near	1	1			1			0	2	
JMP	r	1	1			1			0	2	
JMP	m	1	1			1	1		0	2	
Conditional jump	short/near	1	1			1			0	1-2	fast if no jump
Fused arithmetic and branch		1	1			1			0	1-2	fast if no
J(E/R)CXZ	short	2	2		v	1				1-2	jump
LOOP	short	7	7	X	Х					4-5	
LOOP LOOP(N)E	short	11	11							6	
CALL	near	2	1			1	1	1		2	
CALL	r	2	1			1	1	1		2	
CALL	m	3	1			1	2	1		2	
RET	'''	2	1			1	1	'		2	
RET	i	3	2	x	x	1	1			2	
BOUND	r,m	15	13	^		'	2			7	not 64 bit
INTO	1,111	4	4	x	x	х	_			6	not 64 bit
		"	•	^	^	^					
String instructions											
LODS		3	2	X	Х	Х	1			1	

REP LODS STOS REP STOS		~5n 3 many	1	x	х	x	1	1	~2n n	1	worst case
REP STOS		many							1/16B		best case
MOVS REP MOVS		5 2n	2	x	x	x	2	1	n	4	worst case
REP MOVS		4/16B							1/16B		best case
SCAS REP SCAS CMPS REP CMPS		3 ~6n 5 ~8n	3	x	x x	x x	1 2		~2n ~2n	1 4	odoc
Synchronization instruct	ions										
XADD	m,r	4	3	Х	Х	Х	1	1	7		
LOCK XADD	m,r	8	5	Х	Х	Х	2	1	22		
LOCK ADD	m,r	7	5	Х	Χ	Х	1	1	22		
CMPXCHG	m,r	5	3	Х	Х	Х	2	1	7		
LOCK CMPXCHG	m,r	9	6	Х	Χ	Х	2	1	22		
CMPXCHG8B	m,r	14	11	Х	Х	Х	2	1	7		
LOCK CMPXCHG8B	m,r	18	15	Х	Х	Х	2	1	22		
CMPXCHG16B	m,r	22	19	Х	Х	Х	2	1	16		
LOCK CMPXCHG16B	m,r	24	21	х	Х	Х	2	1	27		
Other											
NOP / Long NOP		1	0							0.25	
PAUSE		7	7							10	
ENTER	a,0	12	9	Х	Χ	Χ	2	1		8	
ENTER	a,b	45+7b							84+3b		
LEAVE		3	2	Х	Χ	Х	1			6	
XGETBV		8								9	XGETBV
CPUID		37-82							100-340		
RDTSC		21								27	
RDPMC		35								39	
RDRAND	r	13	12	Х	Х	X	1			104-117	RDRAND

Floating point x87 instructions

Instruction	Operands	μοps fused	µops	un	fus	ed d	loma	Latency		Com- ments	
		do- main	p015	p0	p1	р5	p23	p4		through- put	
Move instructions											
FLD	r	1	1			1			1	1	
FLD	m32/64	1					1		3	1	
FLD	m80	4	2		1	1	2		5	2	
FBLD	m80	43	40				3		45	21	
FST(P)	r	1	1			1			1	1	
FST(P)	m32/m64	1					1	1	4	1	

				_							
FSTP	m80	7	3				2	2	5	5	
FBSTP	m80	243								252	
FXCH	r	1	0						0	0.5	
FILD	m	1	1		1		1		6	1	
FIST(P)	m	3	1		1		1	1	7	1	
FISTTP	m	3	1		1		1	1	7	2	SSE3
FLDZ	""	1			'	4		'	'		JOLJ
			1			1					
FLD1		2	2	١,	1	1					
FLDPI FLDL2E etc.		2	2	1	1	_			_	2	
FCMOVcc	r	3	3	1		2			2	2	
FNSTSW	AX	2	2	1	Х	Х			4	1	
FNSTSW	m16	2	1				1	1		1	
FLDCW	m16	3	2			2	1			3	
FNSTCW	m16	2	1			1	1	1		1	
FINCSTP FDECSTP		1	1			1			1	1	
FFREE(P)	r	1	1			1				1	
FNSAVE	m	143				·				167	
FRSTOR		90								162	
FRSTOR	m	90								102	
Arithmetic instructions											
FADD(P) FSUB(R)(P)	r	1	1		1				3	1	
FADD(P) FSUB(R)(P)	m	2	1		1		1			1	
FMUL(P)	r '''	1	1	1	'		'		5	1	
` '		2					4		3	1	
FMUL(P)	m		1	1			1		40.04	-	
FDIV(R)(P)	r	1	1	1					10-24	8-18	
FDIV(R)(P)	m	2	1	1			1		_	8-18	
FABS		1	1			1			1	1	
FCHS		1	1			1			1	1	
FCOM(P) FUCOM	r	1	1		1				3	1	
FCOM(P) FUCOM	m	1	1		1		1			1	
FCOMPP FUCOMPP		2	2		1	1			4	1	
FCOMI(P) FUCOMI(P)	r	3	3	1	1	1			5	1	
FIADD FISUB(R)	m	2	2		2		1			2	
FIMUL	m	2	2	1	1		1			2	
FIDIV(R)	m	2	2	1	1		1			_	
FICOM(P)	m	2	2	'	2		1			2	
FTST	""	1	1		1		'			1	
		2	2								
FXAM		1			2				04.00	2	
FPREM		28	28						21-26	12	
FPREM1		41							27-50	19	
FRNDINT		17	17						22	11	
Math											
FSCALE		25	25	x	x	х			49	49	
FXTRACT		17	17	x	X	X			10	10	
FSQRT		1 1	17		^	^			10-23	8-17	
			'	1							
FSIN		21-78		X	X	X			47-106	47-106	
FCOS		23-100		X	Х	Х			48-115	48-115	
FSINCOS		20-110		Х	Х	Х			50-123	50-123	
F2XM1		16-23		Х	Х	Х			~68	~68	
FYL2X		42	42	Х	Х	Χ			90-106		
FYL2XP1		56	56	Х	Х	Х			82		
FPTAN		102	102	Х	Х	Х			130		
	-						•	-			'

FPATAN	28-	72	х	х	х	94-150		
Other								
FNOP	1	1			1		1	
WAIT	2	2	х	х	1		1	
FNCLEX	5	5 5	х	х	Х		22	
FNINIT	2	6 26	x	х	Х		80	

Instruction	Operands	μορs fused	μops	un	fuse	ed d	loma	in	Latency		Com- ments
		do- main	p015	p0	p1	р5	p23	p4			
Move instructions											
MOVD	r32/64,mm/x	1	1	1					1	1	
MOVD	m32/64,mm/x	1					1	1	3	1	
MOVD	mm/x,r32/64	1	1			1			1	1	
MOVD	mm/x,m32/64	1					1		3	0.5	
MOVQ	mm/x,mm/x	1	1	Х	х	х			1	0.33	
MOVQ	mm/x,m64	1					1		3	0.5	
MOVQ	m64, mm/x	1					1	1	3	1	
MOVDQA MOVDQU	x,x	1	1	X	Х	х			0-1	0.25	eliminat.
MOVDQA MOVDQU	x, m128	1					1		3	0.5	
MOVDQA MOVDQU	m128, x	1					1	1	3	1	
LDDQU	x, m128	1	1				1		3	0.5	SSE3
MOVDQ2Q	mm, x	2	2	X	х	1			1	1	
MOVQ2DQ	x,mm	1	1						1	0.33	
MOVNTQ	m64,mm	1					1	1	~360	1	
MOVNTDQ	m128,x	1					1	1	~360	1	
MOVNTDQA	x, m128	1					1	-	3	0.5	SSE4.1
PACKSSWB/DW	,,,,,,,										
PACKUSWB	mm,mm	1	1	1					1	1	
PACKSSWB/DW											
PACKUSWB	mm,m64	1	1	1			1			1	
PACKSSWB/DW											
PACKUSWB	X,X	1	1		Х	Х			1	0.5	
PACKSSWB/DW							١.				
PACKUSWB	x,m128	1	1		Х	Х	1		1	0.5	
PACKUSDW	X,X	1	1		Х	Х			1	0.5	SSE4.1
PACKUSDW	x,m	1	1		Х	Х	1			0.5	SSE4.1
PUNPCKH/LBW/WD/DQ	mm/x,mm/x	1	1		Х	Х	١.		1	0.5	
PUNPCKH/LBW/WD/DQ	mm/x,m	1	1		Х	Х	1			0.5	
PUNPCKH/LQDQ	X,X	1	1		Х	X			1	0.5	
PUNPCKH/LQDQ	x, m128	2	1		Х	X	1			0.5	
PMOVSX/ZXBW	X,X	1	1		Х	X			1	0.5	SSE4.1
PMOVSX/ZXBW	x,m64	1	1		Х	X	1			0.5	SSE4.1
PMOVSX/ZXBD	X,X	1	1		Х	X			1	0.5	SSE4.1
PMOVSX/ZXBD	x,m32	1	1		Х	Х	1			0.5	SSE4.1
PMOVSX/ZXBQ	X,X	1	1		Х	Х			1	0.5	SSE4.1
PMOVSX/ZXBQ	x,m16	1	1		Х	Х	1			0.5	SSE4.1
PMOVSX/ZXWD	X,X	1	1		Х	X			1	0.5	SSE4.1
PMOVSX/ZXWD	x,m64	1	1		Х	Х	1			0.5	SSE4.1

PMOVSX/ZXWQ x,x 1 1 x x 1 0.5 SSE4 PMOVSX/ZXWQ x,m32 1 1 x x 1 0.5 SSE4 PMOVSX/ZXDQ x,m64 1 1 x x 1 0.5 SSE4 PSHUFB mm/x,mm/x 1 1 x x 1 0.5 SSSE4 PSHUFB mm/x,mm/x 1 1 x x 1 0.5 SSSE4 PSHUFW mm/x,m 2 1 x x 1 0.5 SSSE4 PSHUFW mm,mm,i 1 1 x x 1 0.5 SSSE4 PSHUFW mm,mm,i 1 1 x x 1 0.5 SSSE4 PSHUFD xmm,x,i 1 1 x x 1 0.5 SSSE4
PMOVSX/ZXDQ x,x 1 1 x x 1 0.5 SSE4 PMOVSX/ZXDQ x,m64 1 1 x x 1 0.5 SSE4 PSHUFB mm/x,mm/x 1 1 x x 1 0.5 SSSE4 PSHUFB mm/x,m 2 1 x x 1 0.5 SSSE4 PSHUFW mm,mm,i 1 1 x x 1 0.5 SSSE4 PSHUFW mm,mm,i 1 1 x x 1 0.5 SSSE4 PSHUFW mm,m64,i 2 1 x x 1 0.5 O.5
PMOVSX/ZXDQ x,m64 1 1 x x 1 0.5 SSE4 PSHUFB mm/x,mm/x 1 1 x x 1 0.5 SSSE PSHUFB mm/x,m 2 1 x x 1 0.5 SSSE PSHUFW mm,mm,i 1 1 x x 1 0.5 SSSE PSHUFW mm,m64,i 2 1 x x 1 0.5 0.5
PSHUFB mm/x,mm/x 1 1 x x 1 0.5 SSSE PSHUFB mm/x,m 2 1 x x 1 0.5 SSSE PSHUFW mm,mm,i 1 1 x x 1 0.5 SSSE PSHUFW mm,m64,i 2 1 x x 1 0.5 0.5
PSHUFB mm/x,m 2 1 x x 1 0.5 SSSI PSHUFW mm,mm,i 1 1 x x 1 0.5 0.5 PSHUFW mm,m64,i 2 1 x x 1 0.5
PSHUFW mm,mm,i 1 1 1 x x 1 1 0.5 PSHUFW mm,m64,i 2 1 x x 1 0.5
PSHUFW mm,m64,i 2 1 x x 1 0.5
PSHUFW mm,m64,i 2 1 x x 1 0.5
ן אוווווא ן אוווווא ן אווווווא ן אווווווא ן ער אווווווא ן ער אוווווווא ן ער אוווווווא ו
PSHUFD
PSHUFL/HW x,x,i 1 1 x x 1 0.5
PSHUFL/HW x, m128,i 2 1 x x 1 0.5
PALIGNR mm/x,mm/x,i 1 1 x x 1 0.5 SSSI
PALIGNR mm/x,m,i 2 1 x x 1 0.5 SSSI
PBLENDVB
PBLENDVB
PBLENDW x,x,i
PBLENDW x,m,i 2 1 x x 1 0.5 SSE4
MASKMOVQ mm,mm 4 1 1 2 1 1 1
MASKMOVDQU x,x 10 4 x 1 x 4 2 6
PMOVMSKB r32,mm/x 1 1 1 2 1
PEXTRB r32,x,i 2 2 1 x x 2 1 SSE4
PEXTRB m8,x,i 2 1 x x 1 1 SSE4
PEXTRW r32,mm/x,i 2 1 1 x x 2 1
PEXTRW 132,11111/x,1 2 1 1 x x 1 1 SSE4
PEXTRD r32,x,i 2 2 1 x x 2 1 SSE4
PEXTRD 132,x,i 2 2 1 x x 1 1 SSE4
PEXTRQ r64,x,i 2 2 1 x x 2 1 SSE4
PINSRB
PINSRB
PINSRW x,riio,i 2 1 x x 1 0.3 33E4
PINSRW mm/x,m16,i 2 1 x x 1 0.5
PINSRQ x,m64,i 2 1 x x 1 0.5 SSE4
Arithmetic instructions
PCMPEQQ
PCMPGTQ
PCMPGTQ
PMULL/HW PMULHUW mm/x,mm/x 1 1 1 5 1
PMULL/HW PMULHUW mm/x,m 1 1 1 1 1 1 1 1 1
PMULHRSW mm/x,mm/x 1 1 1 5 1 SSSI
PMULHRSW mm/x,m 1 1 1 1 1 1 SSSE

PMULLD	X,X	1	1	1				5	1	SSE4.1
PMULLD	x,m128	2	1	1			1		1	SSE4.1
PMULDQ	x,x	1	1	1				5	1	SSE4.1
PMULDQ	x,m128	1	1	1			1		1	SSE4.1
PMULUDQ	mm/x,mm/x	1	1	1				5	1	
PMULUDQ	mm/x,m	1	1	1			1		1	
PMADDWD	mm/x,mm/x	1	1	1				5	1	
PMADDWD	mm/x,m	1	1	1			1		1	
PMADDUBSW	mm/x,mm/x	1	1	1				5	1	SSSE3
PMADDUBSW	mm/x,m	1	1	1			1		1	SSSE3
PAVGB/W	mm/x,mm/x	1	1		Х	х		1	0.5	
PAVGB/W	mm/x,m	1	1		х	х	1		0.5	
PMIN/MAXSB	x,x	1	1		Х	х		1	0.5	SSE4.1
PMIN/MAXSB	x,m128	1	1		х	х	1		0.5	SSE4.1
PMIN/MAXUB	mm/x,mm/x	1	1		Х	х		1	0.5	
PMIN/MAXUB	mm/x,m	1	1		x	х	1		0.5	
PMIN/MAXSW	mm/x,mm/x	1	1		Х	х		1	0.5	
PMIN/MAXSW	mm/x,m	1	1		Х	X	1		0.5	
PMIN/MAXUW	x,x	1	1		X	X		1	0.5	SSE4.1
PMIN/MAXUW	x,m	1	1		X	X	1	·	0.5	SSE4.1
PMIN/MAXU/SD	x,x	1	1		X	X		1	0.5	SSE4.1
PMIN/MAXU/SD	x,m128	1	1		X	X	1	•	0.5	SSE4.1
PHMINPOSUW	X,X	1	1	1			.	5	1	SSE4.1
PHMINPOSUW	x,m128	1	1	1			1		1	SSE4.1
PABSB/W/D	mm/x,mm/x	1	1	'	х	Х	.	1	0.5	SSSE3
PABSB/W/D	mm/x,m	1	1		X	X	1		0.5	SSSE3
PSIGNB/W/D	mm/x,mm/x	1	1		X	X	.	1	0.5	SSSE3
PSIGNB/W/D	mm/x,m	1	1		X	X	1		0.5	SSSE3
PSADBW	mm/x,mm/x	1	1	1	^	^	'	5	1	OCOLO
PSADBW	mm/x,m	1	1	1			1		1	
MPSADBW	x,x,i	3	3	1	1	1	'	6	1	SSE4.1
MPSADBW	x,m,i	4	3	1	1	1	1		1	SSE4.1
WII GABBY	Α,ιιι,ι	-	"	'	'	'	'		'	OOL4.1
Logic instructions										
PAND(N) POR PXOR	mm/x,mm/x	1	1	X	х	Х		1	0.33	
PAND(N) POR PXOR	mm/x,m	1	1	X	X	X	1	•	0.5	
PTEST	x,x	2	2	1	X	X	.	1	1	SSE4.1
PTEST	x,m128	3	2	1	X	X	1		1	SSE4.1
PSLL/RL/RAW/D/Q	mm,mm/i	1	1	1	^	^	'	1	1	OOL+.1
PSLL/RL/RAW/D/Q	mm,m64	1	1	1			1	'	1	
PSLL/RL/RAW/D/Q	xmm,i	1	1	1				1	1	
PSLL/RL/RAW/D/Q	X,X	2	2	1	х	Х		2	1	
PSLL/RL/RAW/D/Q	x,m128	3	2	1	X	X	1	_	1	
PSLL/RLDQ	x,iii 120	1	1	'	X	X	'	1	0.5	
I GEE/TEBQ	Λ,ι		'						0.0	
String instructions										
PCMPESTRI	x,x,i	8	8	3	1	4		4	4	SSE4.2
PCMPESTRI	x,m128,i	8	7	3	1	3	1		4	SSE4.2
PCMPESTRM	x,x,i	8	8	3	1	4		12	4	SSE4.2
PCMPESTRM	x,m128,i	8	7	3	1	3	1		4	SSE4.2
PCMPISTRI	x,x,i	3	3	3				3		SSE4.2
PCMPISTRI	x,m128,i	4	3	3			1		3	SSE4.2
PCMPISTRM	x,x,i	3	3	3				11		SSE4.2
1	1		1					'	•	. '

PCMPISTRM	x,m128,i	4	3	3			1		3	SSE4.2
Encryption instructions										
PCLMULQDQ	x,x,i	18	18	Х	Х	х		14	8	CLMUL
PCLMULQDQ	x,m,i	18	17	Х	Х	х	1		8	CLMUL
AESDEC, AESDECLAST, AESENC, AESENCLAST										
	X,X	2	2	Х	Х	1		4	1	AES
AESDEC, AESDECLAST, AESENC, AESENCLAST										
,	x,m	3	2	x	Х	1	1		1	AES
AESIMC	X,X	2	2			2		14	2	AES
AESIMC	x,m	3	2			2	1		2	AES
AESKEYGENASSIST	x,x,i	11	11	х	Х	х		10	8	AES
AESKEYGENASSIST	x,m,i	11	10	х	х	Х	1		7	AES
Other										
EMMS		31	31						18	

Floating point XMM and YMM instructions

Instruction	Operands	μορs fused	μops	un	fus	ed d	loma	in	Latency	procal	Com- ments
		do- main	p015	p0	p1	р5	p23	p4		through- put	
Move instructions											
MOVAPS/D	X,X	1	1			1			0-1	≤1	elimin.
VMOVAPS/D	y,y	1	1			1			0-1	≤1	elimin.
MOVAPS/D MOVUPS/D	x,m128	1					1		3	0.5	
VMOVAPS/D VMOVUPS/											
D	y,m256	1					1+		4	1	AVX
MOVAPS/D MOVUPS/D	m128,x	1					1	1	3	1	
VMOVAPS/D VMOVUPS/							١.				
D	m256,y	1					1	1+	4	2	AVX
MOVSS/D	X,X	1	1			1	١.		1	1	
MOVSS/D	x,m32/64	1					1		3	0.5	
MOVSS/D	m32/64,x	1					1	1	3	1	
MOVHPS/D MOVLPS/D	x,m64	2	1			1	1		4	1	
MOVH/LPS/D	m64,x	2					1	1	3	1	
MOVLHPS MOVHLPS	X,X	1	1			1			1	1	
MOVMSKPS/D	r32,x	1	1	1					2	1	
VMOVMSKPS/D	r32,y	1	1	1					2	1	
MOVNTPS/D	m128,x	1					1	1	~380	1	
VMOVNTPS/D	m256,y	1					1	1+	~380	2	AVX
SHUFPS/D	x,x,i	1	1			1			1	1	
SHUFPS/D	x,m128,i	2	1			1	1			1	
VSHUFPS/D	y,y,y,i	1	1			1			1	1	AVX
VSHUFPS/D	y, y,m256,i	2	1			1	1+			1	AVX
VPERMILPS/PD	x,x,x/i	1	1			1			1	1	AVX
VPERMILPS/PD	y,y,y/i	1	1			1			1	1	AVX
VPERMILPS/PD	x,x,m	2	1			1	1			1	AVX
VPERMILPS/PD	y,y,m	2	1			1	1+			1	AVX
VPERMILPS/PD	x,m,i	2	1			1	1			1	AVX

VPERM2F128 y,y,m,i 1 0.5 SSE4.1 BLENDVPS/PD y,m2566,i 2 1 1 1 1 3 2 X 1 1 SSE4.1 BLENDVPS/PD y,y,y,y,y 2 2 x x 1 1 SSE4.2 VBLENDVPS/PD y,y,y,m,y 3 2 x x 1 1 AVX <th>- 1</th>	- 1
BLENDPS/PD	
BLENDPS/PD	
VBLENDPS/PD y,y,i 1 1 x x 1 0.5 AVX VBLENDPS/PD y,m256,i 2 1 x x 1+ 1 AVX BLENDVPS/PD x,x,xmm0 2 2 x x 2 1 SSE4.* VBLENDVPS/PD y,y,y,y 2 2 x x 1 1 SSE4.* VBLENDVPS/PD y,y,y,my 3 2 x x 1 1 AVX VBLENDVPS/PD y,y,my 3 2 x x 1 1 AVX VBLENDVPS/PD y,y,my 3 2 x x 1 1 AVX VBLENDVPS/PD y,y,y,my 3 2 x x 1 1 AVX MOVDUP x,m64 1 1 1 1 1 AVX VBROADCASTSS x,m32 1 1 1 4 0.5 AVX	
VBLENDPS/PD y,m256,i 2 1 x x 1+ 2 1 AVX BLENDVPS/PD x,x,xmm0 2 2 x x 1 SSE4.1 VBLENDVPS/PD y,y,y,y 2 2 x x 1 1 SSE4.1 VBLENDVPS/PD y,y,m,y 3 2 x x 1 1 AVX MOVDDUP x,x 1 2 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 3 0.5 SSE3	
BLENDVPS/PD x,x,xmm0 2 2 x x 1 SSE4.1 BLENDVPS/PD x,m,xmm0 3 2 x x 1 1 SSE4.1 VBLENDVPS/PD y,y,y,y 2 2 x x 1 1 AVX VBLENDVPS/PD y,y,m,y 3 2 x x 1 1 AVX VBLENDVPS/PD y,y,m,y 3 2 x x 1 1 AVX VBLENDVPS/PD y,y,m,y 3 2 x x 1 1 AVX VBLENDVPS/PD y,y,m,m,y 3 2 x x 1 1 AVX MOVDUP x,m64 1 1 1 1 1 1 AVX VMOVDDUP y,m256 1 1 1 1 4 0.5 AVX VBROADCASTSS y,m32 2 1 1 1 5 1 <	
BLENDVPS/PD x,m,xmm0 3 2 x 1 1 SSE4.1 VBLENDVPS/PD y,y,y,y 2 2 x x 1 AVX VBLENDVPS/PD y,y,m,y 3 2 x x 1+ 1 AVX MOVDDUP x,x 1 1 1 1 1 1 AVX VMOVDDUP y,m64 1 1 1 1 1 AVX VMOVDDUP y,m256 1 1+ 3 1 AVX VMOVDDUP y,m32 1 1+ 3 1 AVX VBROADCASTSS y,m32 2 1 1 1 4 0.5 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 1 1 AVX VBROADLASTF128 y,m256	
VBLENDVPS/PD y,y,y,y 2 2 x x 1 AVX VBLENDVPS/PD y,y,m,y 3 2 x x 1+ 1 AVX MOVDDUP x,x 1 1 1 1 1 1 1 SSE3 VMOVDDUP y,y 1 1 1 1 1 1 AVX VMOVDDUP y,m256 1 1+ 3 1 AVX VBROADCASTSS y,m32 1 1+ 3 1 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 5 1 AVX VBROADCASTF128 y,m128 1 1 1 1 1 1 1 1	
VBLENDVPS/PD y,y,m,y 3 2 x x 1+ 1 AVX MOVDDUP x,x 1	
MOVDDUP x,x 1 1 1 1 1 1 1 1 1 3 0.5 SSE3 VMOVDDUP y,y,y 1 1 1 1 1 1 1 AVX VMOVDDUP y,m256 1 1 1 1 1 1 AVX VBROADCASTSS x,m32 1 1 1 4 0.5 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 1 1 1 1 1 1	
MOVDDUP x,m64 1 1 1 1 3 0.5 SSE3 VMOVDDUP y,y 1 1 1 1 1 1 AVX VBROADCASTSS x,m32 1 1 4 0.5 AVX VBROADCASTSS y,m32 2 1 1 1 5 1 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 AVX WOVSH/LDUP x,m128 1 1 1 1 1 1 1 1	
VMOVDDUP y,y 1 1 1 1 1 1 AVX VMOVDDUP y,m256 1 1 1+ 3 1 AVX VBROADCASTSS x,m32 1 1 4 0.5 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 1 1 1 1 1 3 0.5 SSE3 VXX 1 1 1	
VMOVDDUP y,m256 1 1+ 3 1 AVX VBROADCASTSS x,m32 1 1 4 0.5 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 AVX VBROADCASTF128 y,m128 2 1 1 1 1 1 AVX WONSH/LDUP x,m128 1 1 1 1 1 1 AVX VMOVSH/LDUP y,m256 1 1 1 1 1 1 1 AVX	
VBROADCASTSS x,m32 1 1 4 0.5 AVX VBROADCASTSS y,m32 2 1 1 1 5 1 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTSD y,m64 2 1 1 1 1 5 1 AVX VBROADCASTSD y,m64 2 1 1 1 1 AVX VBROADCASTSD y,m128 1<	
VBROADCASTSS y,m32 2 1 1 1 5 1 AVX VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX VBROADCASTSD y,m64 2 1 1 1 1 1 AVX VBROADCASTSD y,m128 2 1 1 1 1 1 AVX VBROADCASTSD y,m128 1 1 1 1 1 AVX MOVSH/LDUP x,x 1 1 1 1 3 0.5 SSE3 VMOVSH/LDUP y,m256 1 1 1 1 1 1 AVX UNPCKH/LPS/D x,m128 1 1 1 1 1 1 SSE3 VUNPCKH/LPS/D y,y,m256 1 1 1 1 1 1 AV	
VBROADCASTSD y,m64 2 1 1 1 5 1 AVX VBROADCASTF128 y,m128 2 1 1 1 5 1 AVX MOVSH/LDUP x,x 1 1 1 1 1 1 SSE3 VMOVSH/LDUP y,y 1 1 1 1 1 1 AVX VMOVSH/LDUP y,m256 1 1+ 1 1 AVX VMOVSH/LDUP y,m256 1 1+ 1 1 AVX UNPCKH/LPS/D x,x 1 1 1 1 1 SSE3 VUNPCKH/LPS/D x,m128 1 1 1 1 1 1 AVX VUNPCKH/LPS/D y,y,m256 1 1 1 1 1 AVX VUNPCKH/LPS/D y,y,m256 1 1 1 1 1 AVX EXTRACTPS m32,x,i 3 2	
VBROADCASTF128 y,m128 2 1	
MOVSH/LDUP x,x 1 1 1 1 1 1 SSE3 MOVSH/LDUP x,m128 1 1 1 1 3 0.5 SSE3 VMOVSH/LDUP y,y 1 1 1 1 1 1 AVX VMOVSH/LDUP y,m256 1 1+ 1 1 AVX UNPCKH/LPS/D x,x 1 1 1 1 1 1 SSE3 VUNPCKH/LPS/D x,m128 1 1 1 1 1 1 1 SSE3 VUNPCKH/LPS/D x,y,y 1 1 1 1 1 1 AVX VUNPCKH/LPS/D y,y,m256 1 1 1 1 1 1 AVX EXTRACTPS r32,x,i 2 2 x x 2 1 SSE4.7 VEXTRACTF128 m128,y,i 2 0 1 1 4 1 AVX	
MOVSH/LDUP x,m128 1 1 1 3 0.5 SSE3 VMOVSH/LDUP y,y 1 1 1 1 1 1 AVX VMOVSH/LDUP y,m256 1 1 1 1 1 1 AVX UNPCKH/LPS/D x,x 1 1 1 1 1 1 SSE3 VUNPCKH/LPS/D x,m128 1 1 1 1 1 1 SSE3 VUNPCKH/LPS/D x,y,y 1 1 1 1 1 1 AVX VUNPCKH/LPS/D y,y,m256 1 1 1 1 1 1 AVX VUNPCKH/LPS/D y,y,m256 1 1 1 1 1 AVX EXTRACTPS r32,x,i 2 2 x x 2 1 SSE4.1 VEXTRACTF128 m128,y,i 2 0 1 1 4 1 AVX <td></td>	
VMOVSH/LDUP y,y 1 1 1 1 1 AVX VMOVSH/LDUP y,m256 1 1 1+ 1 1 AVX UNPCKH/LPS/D x,x 1 1 1 1 1 1 SSE3 VUNPCKH/LPS/D x,m128 1 1 1 1 1 1 SSE3 VUNPCKH/LPS/D y,y,y 1 1 1 1 1 1 AVX VUNPCKH/LPS/D y,y,m256 1 1 1 1 1 1 AVX EXTRACTPS r32,x,i 2 2 x x 2 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 1 1 2 1 AVX VEXTRACTF128 m128,y,i 2 0 1 1 1 AVX VEXTRACTF128 x,x,i 1 1 1 1 1 1 1 AVX	
VMOVSH/LDUP y,m256 1 1 1+ 1 AVX UNPCKH/LPS/D x,x 1 <	
UNPCKH/LPS/D x,x 1 2 1 1 1 1 1 1 1 1 1 1 1	
UNPCKH/LPS/D x,m128 1	
VUNPCKH/LPS/D y,y,y 1 1 1 1 1 AVX VUNPCKH/LPS/D y,y,m256 1 1 1 1+ 1 1 AVX EXTRACTPS r32,x,i 2 2 x x 2 1 SSE4.7 EXTRACTPS m32,x,i 3 2 x x 1 1 1 SSE4.7 VEXTRACTF128 x,y,i 1 1 1 2 1 AVX VEXTRACTF128 m128,y,i 2 0 1 1 4 1 AVX INSERTPS x,x,i 1 1 1 1 1 1 1 SSE4.7 INSERTPS x,m32,i 2 1 1 1 1 1 1 SSE4.7	
VUNPCKH/LPS/D y,y,m256 1 1 1 1+ 1 AVX EXTRACTPS r32,x,i 2 2 x x 2 1 SSE4.7 EXTRACTPS m32,x,i 3 2 x x 1 1 1 SSE4.7 VEXTRACTF128 x,y,i 1 1 1 2 1 AVX VEXTRACTF128 m128,y,i 2 0 1 1 4 1 AVX INSERTPS x,x,i 1 1 1 1 1 1 1 SSE4.7 INSERTPS x,m32,i 2 1 1 1 1 1 SSE4.7	
EXTRACTPS r32,x,i 2 2 x x 2 1 SSE4.1 EXTRACTPS m32,x,i 3 2 x x 1 1 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 1 2 1 AVX VEXTRACTF128 m128,y,i 2 0 1 1 4 1 AVX INSERTPS x,x,i 1 1 1 1 1 1 1 SSE4.1 INSERTPS x,m32,i 2 1 1 1 1 1 SSE4.1	
EXTRACTPS m32,x,i 3 2 x 1 1 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 1 2 1 AVX VEXTRACTF128 m128,y,i 2 0 1 1 4 1 AVX INSERTPS x,x,i 1 1 1 1 1 1 1 1 SSE4.1 INSERTPS x,m32,i 2 1 1 1 1 1 SSE4.1	
VEXTRACTF128 x,y,i 1 1 1 2 1 AVX VEXTRACTF128 m128,y,i 2 0 1 1 4 1 AVX INSERTPS x,x,i 1 1 1 1 1 1 1 SSE4.1 INSERTPS x,m32,i 2 1 1 1 1 1 SSE4.1	
VEXTRACTF128 m128,y,i 2 0 1 1 4 1 AVX INSERTPS x,x,i 1 1 1 1 1 1 1 1 SSE4.1 INSERTPS x,m32,i 2 1 1 1 1 1 SSE4.1	
INSERTPS	
INSERTPS x,m32,i 2 1 1 1 1 SSE4.1	
VINSERTF128 y,y,x,i 1 1 1 2 1 AVX	
VINSERTF128 y,y,m128,i 2 1 x 1 4 1 AVX	
VMASKMOVPS/D x,x,m128 3 2 x 1 4 1 AVX	
VMASKMOVPS/D y,y,m256 3 2 1+ 5 1 AVX	
VMASKMOVPS/D m128,x,x 4 2 x x 1 1 1 AVX	
VMASKMOVPS/D m256,y,y 4 2 x x 1 1+ 2 AVX	
Conversion	
CVTPD2PS x,x 2 2 1 1 1 4 1	
CVTPD2PS x,m128 2 2 1 1 1 1 1	
VCVTPD2PS x,y 2 2 1 1 4 1 AVX	
VCVTPD2PS	
CVTSD2SS	
CVTSD2SS x,m64 2 2 1 1 1 1 1	
CVTPS2PD x,x 2 2 1 1 1 1 1 1 1	
CVTPS2PD x,m64 2 1 1 1 1 1	
VCVTPS2PD y,x 2 2 1 1 4 1 AVX	
VCVTPS2PD	
CVTSS2SD x,x 2 2 1 1 2 1	
CVTSS2SD x,m32 2 1 1 1 1 1	

CVTDQ2PS	x,x	1	1		1				3	1	
CVTDQ2PS	x,m128	1	1		1		1			1	
VCVTDQ2PS	y,y	1	1		1				3	1	AVX
VCVTDQ2PS	y,m256	1	1		1		1+			1	AVX
CVT(T) PS2DQ	x,x	1	1		1		•		3	1	
CVT(T) PS2DQ	x,m128	1	1		1		1			1	
1							'		_		A) ()(
VCVT(T) PS2DQ	у,у	1	1		1				3	1	AVX
VCVT(T) PS2DQ	y,m256	1	1		1		1+		_	1	AVX
CVTDQ2PD	X,X	2	2		1	1			4	1	
CVTDQ2PD	x,m64	2	2		1	1	1			1	
VCVTDQ2PD	y,x	2	2		1	1			5	1	AVX
VCVTDQ2PD	y,m128	2	2		1	1	1			1	AVX
CVT(T)PD2DQ	X,X	2	2		1	1			4	1	
CVT(T)PD2DQ	x,m128	2	2		1	1	1			1	
VCVT(T)PD2DQ	x,y	2	2		1	1			5	1	AVX
VCVT(T)PD2DQ	x,m256	2	2		1	1	1+			1	AVX
CVTPI2PS	x,mm	1	1		1	•			4	'	/\\
CVTPI2PS	x,m64	1	1		1		1		,	3	
	· ·					4	'		4		
CVT(T)PS2PI	mm,x	2	2		1	1			4	1	
CVT(T)PS2PI	mm,m128	2	1		1		1		_	1	
CVTPI2PD	x,mm	2	2		1	1			4	1	
CVTPI2PD	x,m64	2	2		1	1	1			1	
CVT(T) PD2PI	mm,x	2	2		1	1			4	1	
CVT(T) PD2PI	mm,m128	2	2		1	1	1			1	
CVTSI2SS	x,r32	2	2		1	1			4	3	
CVTSI2SS	x,m32	1	1		1		1			3	
CVT(T)SS2SI	r32,x	2	2	1	1				4	1	
CVT(T)SS2SI	r32,m32	2	2	1	1		1			1	
CVTSI2SD	x,r32	2	2		1	1			4	3	
CVTSI2SD	x,m32	2	1		1	-	1		-	3	
CVT(T)SD2SI	r32,x	2	2	1	1		-		4	1	
CVT(T)SD2SI	r32,m64	2	2	1	1		1		•	1	
VCVTPS2PH	x,v,i	3	3	1	1	1	'		10	1	F16C
VCVTPS2PH		3	2	1	1	'	1	4	10		l I
I .	m,v,i			1	ļ '	4	ı	1		1	F16C
VCVTPH2PS	V,X	2	2	1		1			6	1	F16C
VCVTPH2PS	v,m	2	1		1		1			1	F16C
Arithmetic											
ADDSS/D SUBSS/D	X,X	1	1		1				3	1	
ADDSS/D SUBSS/D	x,m32/64	1	1		1		1			1	
ADDPS/D SUBPS/D	x,x	1	1		1				3	1	
ADDPS/D SUBPS/D	x,m128	1	1		1		1			1	
VADDPS/D VSUBPS/D	y,y,y	1	1		1				3	1	AVX
VADDPS/D VSUBPS/D	y,y,m256	1	1		1		1+			1	AVX
ADDSUBPS/D	x,x	1	1		1				3	1	SSE3
ADDSUBPS/D	x,m128	1	1		1		1		•	1	SSE3
VADDSUBPS/D		1	1		1				3	1	AVX
VADDSUBPS/D	y,y,y y,y,m256	1	1		1		1+			1	AVX
HADDPS/D HSUBPS/D						2	1 +		F		SSE3
	X,X	3	3		1	2			5	2	l I
HADDPS/D HSUBPS/D	x,m128	4	3		1	2	1			2	SSE3
VHADDPS/D VHSUBPS/			2			_				_	A1/3/
D	у,у,у	3	3		1	2			5	2	AVX

VHVDDD6/D VH6HBD6/		I	-	ı	l	l	1 1	1	1	
VHADDPS/D VHSUBPS/ D	y,y,m256	4	3		1	2	1+		2	AVX
MULSS MULPS	x,x	1	1	1		_	'	5	1	/(//
MULSS MULPS	x,m	1	1	1			1		1	
VMULPS	у,у,у	1	1	1				5	1	AVX
VMULPS	y,y,m256	1	1	1			1+		1	AVX
MULSD MULPD	x,x	1	1	1				5	1	,, .
MULSD MULPD	x,m	1	1	1			1		1	
VMULPD	у,у,у	1	1	1				5	1	AVX
VMULPD	y,y,m256	1	1	1			1+		1	AVX
DIVSS DIVPS	x,x	1	1	1				10-13	7	,, .
DIVSS DIVPS	x,m	1	1	1			1		7	
VDIVPS	у,у,у	3	3	2		1		19-21	14	AVX
VDIVPS	y,y,m256	4	3	2		1	1+	102.	14	AVX
DIVSD DIVPD	x,x	1	1	1				10-20	8-14	,, .
DIVSD DIVPD	x,m	1	1	1			1	10 20	8-14	
VDIVPD	у,у,у	3	3	2		1	'	20-35		AVX
VDIVPD	y,y,m256	4	3	2		1	1+	20 00	16-28	AVX
RCPSS/PS	x,x	1	1	1			'	5	1	/ (/ /
RCPSS/PS	x,m128	1	1	1			1		1	
VRCPPS	у,у	3	3	2		1	'	7	2	AVX
VRCPPS	y,m256	4	3	2		1	1+		2	AVX
CMPccSS/D CMPccPS/D	y ,									
	X,X	1	1		1			3	1	
CMPccSS/D CMPccPS/D										
	x,m128	2	1		1		1		1	
VCMPccPS/D	y,y,y	1	1		1			3	1	AVX
VCMPccPS/D	y,y,m256	2	1		1		1+		1	AVX
COMISS/D UCOMISS/D	x,x	2	2	1	1				1	
COMISS/D UCOMISS/D	x,m32/64	2	2	1	1		1		1	
MAXSS/D MINSS/D	X,X	1	1		1			3	1	
MAXSS/D MINSS/D	x,m32/64	1	1		1		1		1	
MAXPS/D MINPS/D	X,X	1	1		1			3	1	
MAXPS/D MINPS/D	x,m128	1	1		1		1		1	
VMAXPS/D VMINPS/D	у,у,у	1	1		1			3	1	AVX
VMAXPS/D VMINPS/D	y,y,m256	1	1		1		1+		1	AVX
ROUNDSS/SD/PS/PD	x,x,i	1	1		1			3	1	SSE4.1
ROUNDSS/SD/PS/PD	x,m128,i	2	1		1		1		1	SSE4.1
VROUNDSS/SD/PS/PD	y,y,i	1	1		1			3	1	AVX
VROUNDSS/SD/PS/PD	y,m256,i	2	1	١.	1		1+		1	AVX
DPPS	x,x,i	4	4	1	2	1		12	2	SSE4.1
DPPS	x,m128,i	6	5	1	2	2	1		4	SSE4.1
VDPPS	y,y,y,i	4	4	1	2	1		12	2	AVX
VDPPS	y,m256,i	6	5	1	2	2	1+		4	AVX
DPPD	x,x,i	3	3	1	1	1		9	1	SSE4.1
DPPD	x,m128,i	4	3	1	1	1	1		1	SSE4.1
Math										
SQRTSS/PS	x,x	1	1	1				11	7	
SQRTSS/PS	x,m128	1	1	1			1		7	
VSQRTPS	у,у у,у	3	3	2		1		19	14	AVX
VSQRTPS	y,m256	4	3	2		1	1+		14	AVX
SQRTSD/PD	x,x	1	1	1				16	8-14	
	,	1 -	1 -	1 -	I	I	1 1	1	1	1 1

SQRTSD/PD	x,m128	1	1	1		1			8-14	
VSQRTPD	у,у	3	3	2	1			28	16-28	AVX
VSQRTPD	y,m256	4	3	2	1	1+			16-28	AVX
RSQRTSS/PS	x,x	1	1	1				5	1	
RSQRTSS/PS	x,m128	1	1	1		1			1	
VRSQRTPS	y,y	3	3	2	1			7	2	AVX
VRSQRTPS	y,m256	4	3	2	1	1+			2	AVX
Logic										
AND/ANDN/OR/XORPS/PD	x,x	1	1		1			1	1	
AND/ANDN/OR/XORPS/PD	x,m128	1	1		1	1			1	
VAND/ANDN/OR/XORPS/										
PD	y,y,y	1	1		1			1	1	AVX
VAND/ANDN/OR/XORPS/									_	
PD	y,y,m256	1	1		1	1+			1	AVX
Other										
VZEROUPPER		4	0						1	AVX
VZEROALL		12	2						11	32 bit
VZEROALL		20	2						9	64 bit
LDMXCSR	m32	3	2	1	1	1		6	3	
STMXCSR	m32	3	2	1	1	1	1	7	1	
FXSAVE	m4096	130							66	
FXRSTOR	m4096	116							68	
XSAVEOPT	m	100-16	1					60-500		

Intel Haswell

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the same data.

Instructions with or without V name prefix behave the same unless otherwise noted.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm register, mm/

x = mmx or xmm register, y = 256 bit ymm register, v = any vector register (mmx, xmm, ymm). same = same register for both operands. m = memory operand, m32 = 32-bit

memory operand, etc.

µops fused

The number of µops at the decode, rename and allocate stages in the pipeline. Fused

domain: µops count as one.

μops unfused domain:

The total number of µops for all execution port. Fused µops count as two. Fused macroops count as one. The instruction has µop fusion if this number is higher than the num-

ber under fused domain. Some operations are not counted here if they do not go to any

execution port or if the counters are inaccurate.

μορs each port: The number of μορs for each execution port. p0 means a μορ to execution port 0.

p01means a μop that can go to either port 0 or port 1. p0 p1 means two μops going to

port 0 and 1, respectively.

Port 0: Integer, f.p. and vector ALU, mul, div, branch

Port 1: Integer, f.p. and vector ALU

Port 2: Load Port 3: Load Port 4: Store

Port 5: Integer and vector ALU Port 6: Integer ALU, branch

Port 7: Store address

Latency: This is the delay that the instruction generates in a dependency chain. The numbers are

minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Where hyperthreading is enabled, the use of the same execution units in the other thread leads to inferior performance. Denormal numbers, NAN's and infinity do not increase the latency. The time unit used is core clock cycles, not the refer-

ence clock cycles given by the time stamp counter.

Reciprocal throughput:

The average number of core clock cycles per instruction for a series of independent in-

structions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instruc- tions							
MOV	r,i	1	1	p0156		0.25	
MOV	r8/16,r8/16	1	1	p0156	1	0.25	
MOV	r32/64,r32/64	1	1	p0156	0-1	0.25	may be elim.
MOV	r8l,m	1	2	p23 p0156		0.5	
MOV	r8h,m	1	1	p23		0.5	
MOV	r16,m	1	2	p23 p0156		0.5	
MOV	r32/64,m	1	1	p23	2	0.5	all addressing modes
MOV	m,r	1	2	p237 p4	3	1	
MOV	m,i	1	2	p237 p4		1	
MOVNTI	m,r	2	2	p23 p4	~400	1	

MOVSX MOVZX MOVSXD	r,r	1	1	p0156	1	0.25	
MOVSX MOVZX	r16,m8	1	2	p23 p0156		0.5	
MOVSX MOVZX MOVSXD	r,m	1	1	p23		0.5	all other combinations
CMOVcc	r,r	2	2	2p0156	2	0.5	
CMOVcc	r,m	3	3	2p0156 p23	_	1	
XCHG	r,r	3	3	3p0156	2	1	
XCHG	r,m	8	8	000.00	21		implicit lock
XLAT	',	3	3		7	2	miphon room
PUSH	r	1	2	p237 p4	3	1	
PUSH		1	2	p237 p4	3		
PUSH	m l	2	3	p4 2p237			
PUSH	stack pointer	2	3	p0156 p237 p4			
PUSHF(D/Q)	Stack politici	3	4	p1 p4 p237 p06		1	
PUSHA(D)		3 11	19	p i p4 p237 p00		8	not 64 bit
POP	_			200	2		HOL 04 DIL
	r	1	1	p23	2	0.5	
POP	stack pointer	3	3	p23 2p0156		4	
POP	m	2	3	2p237 p4		1	
POPF(D/Q)		9	9			18	
POPA(D)		18	18			9	not 64 bit
LAHF SAHF		1	1	p06	1	1	
SALC		3	3	3p0156	1	1	not 64 bit
LEA	r16,m	2	2	p1 p0156	4	1	16 or 32 bit
			_				address size
LEA	r32/64,m	1	1	p15	1	0.5	1 or 2 compo- nents in address
LEA	r32/64,m	1	1	p1	3	1	3 components in address
LEA	r32/64,m	1	1	p1		1	rip relative address
BSWAP	r32	1	1	p15	1	0.5	
BSWAP	r64	2	2	p06 p15	2	1	
MOVBE	r16,m16	3	3	2p0156 p23		0.5	MOVBE
MOVBE	r32,m32	2	2	p15 p23		0.5	MOVBE
MOVBE	r64,m64	3	3	2p0156 p23		0.5	MOVBE
MOVBE	m16,r16	2	3	p06 p237 p4		1	MOVBE
MOVBE	m32,r32	2	3	p15 p237 p4		1	MOVBE
MOVBE	m64,r64	3	4	p06 p15 p237 p4		1	MOVBE
PREFETCHNTA/ 0/1/2	m	1	1	p23		0.5	
LFENCE		2		none counted		4	
MFENCE		3	2	p23 p4		33	
SFENCE		2	2	p23 p4		5	
0. 2.102		_	_	μ=σ μ .			
Arithmetic in- structions							
ADD SUB	r,r/i	1	1	p0156	1	0.25	
ADD SUB	r,m	1	2	p0156 p23		0.5	
ADD SUB	m,r/i	2	4	2p0156 2p237 p4	6	1	
ADC SBB	r,r/i	2	2	2p0156	2	1	
ADC SBB	r,m	2	3	2p0156 p23		1	

ADC SBB	m,r/i	4	6	3p0156 2p237 p4	7	2	I
CMP	r,r/i	4 1	1	p0156	1	0.25	
CMP	m,r/i	1	2	p0156 p23	1	0.23	
INC DEC NEG		1	1	p0136 p23	1	0.3	
NOT	r	ı	I	p0130	ı	0.23	
INC DEC NOT	m	3	4	p0156 2p237 p4	6	1	
NEG	m	2	4	p0156 2p237 p4	6	1 1	
AAA		2	2	p1 p0156	4		not 64 bit
AAS		2	2	p1 p56	6		not 64 bit
DAA DAS		3	3	p1 2p0156	4		not 64 bit
AAD		3	3	p1 2p0156	4		not 64 bit
AAM		8	8	p0 p1 p5 p6	21	8	not 64 bit
MUL IMUL	r8	1	1	p1	3	1	HOLOT DIE
MUL IMUL	r16	4	4	p1 p0156	4	2	
MUL IMUL	r32	3	3	p1 p0156	4	2	
MUL IMUL	r64	2	2	p1 p6	3	1	
MUL IMUL	m8	1	2	p1 p23		1	
MUL IMUL	m16	4	5	p1 3p0156 p23		2	
MUL IMUL	m32	3	4	p1 2p0156 p23		2	
MUL IMUL	m64	2	3	p1 p6 p23		1	
IMUL	r,r	1	1	p1 p0 p23	3		
IMUL	r,m	1	2	p1 p23			
IMUL	r16,r16,i	2	2	p1 p0156	4		
IMUL	r32,r32,i	1	1	p1	3		
IMUL	r64,r64,i	1	1	p1	3		
IMUL	r16,m16,i	2	3	p1 p0156 p23			
IMUL	r32,m32,i	1	2	p1 p23			
IMUL	r64,m64,i	1	2	p1 p23			
MULX	r32,r32,r32	3	3	p1 2p056	4		BMI2
MULX	r32,r32,m32	3	4	p1 2p056 p23		1	BMI2
MULX	r64,r64,r64	2	2	p1 p6	4	1	BMI2
MULX	r64,r64,m64	2	3	p1 p6 p23	•	1	BMI2
DIV	r8	9	9	p0 p1 p5 p6	22-25	9	Biviiz
DIV	r16	11	11	p0 p1 p5 p6	23-26	9	
DIV	r32	10	10	p0 p1 p5 p6	22-29	9-11	
DIV	r64	36	36	p0 p1 p5 p6	32-96	21-74	
IDIV	r8	9	9	p0 p1 p5 p6	23-26	8	
IDIV	r16	10	10	p0 p1 p5 p6	23-26	8	
IDIV	r32	9	9	p0 p1 p5 p6	22-29	8-11	
IDIV	r64	59	59	p0 p1 p5 p6	39-103	24-81	
CBW		1	1	p0156	1		
CWDE		1	1	p0156	1		
CDQE		1	1	p0156	1		
CWD		2	2	p0156	1		
CDQ		1	1	p06	1		
CQO		1	1	p06	1		
POPCNT	r,r	1	1	p1	3	1	SSE4.2
POPCNT	r,m	1	2	p1 p23		1	SSE4.2
CRC32	r,r	1	1	p1	3	1	SSE4.2
CRC32	r,m	1	2	p1 p23	_	1	SSE4.2
		•	_	F - F		,	 · · -
Logic instruc-							
tions							

AND OR XOR	r,r/i	1	1	p0156	1	0.25	
AND OR XOR	r,m	1	2	p0156 p23		0.5	
AND OR XOR	m,r/i	2	4	2p0156 2p237 p4	6	1	
TEST	r,r/i	1	1	p0156	1	0.25	
TEST	m,r/i	1	2	p0156 p23		0.5	
SHR SHL SAR	r,i	1	1	p06	1	0.5	
SHR SHL SAR	m,i	3	4	2p06 p237 p4		2	
SHR SHL SAR	r,cl	3	3	3p06	2	2	
SHR SHL SAR	m,cl	5	6	3p06 2p23 p4		4	
ROR ROL	r,1	2	2	2p06	1	1	short form
ROR ROL	r,i	1	1	p06	1	0.5	
ROR ROL	m,i	4	5	2p06 2p237 p4		2	
ROR ROL	r,cl	3	3	3p06	2	2	
ROR ROL	m,cl	5	6	opec	_	4	
RCR RCL	r,1	3	3	2p06 p0156	2	2	
RCR RCL	m,1	4	6	2p00 p0100	_	3	
RCR RCL	r,i	8	8	p0156	6	6	
RCR RCL		11	11	p0130	U	6	
RCR RCL	m,i	8	8	n0156	6	6	
RCR RCL	r,cl			p0156	6		
	m,cl	11	11	4		6	
SHRD SHLD	r,r,i	1	1 5	p1	3	1	
SHRD SHLD	m,r,i	3	5	0450		2	
SHLD	r,r,cl	4	4	p0156	3	2	
SHRD	r,r,cl	4	4	p0156	4	2	
SHRD SHLD	m,r,cl	5	7			4	5.416
SHLX SHRX SARX	r,r,r	1	1	p06	1	0.5	BMI2
SHLX SHRX SARX	r,m,r	2	2	p06 p23		0.5	BMI2
RORX	r,r,i	1	1	p06	1	0.5	BMI2
RORX	r,m,i	2	2	p06 p23		0.5	BMI2
BT	r,r/i	1	1	p06	1	0.5	
BT	m,r	10	10			5	
BT	m,i	2	2	p06 p23		0.5	
BTR BTS BTC	r,r/i	1	1	p06	1	0.5	
BTR BTS BTC	m,r	10	11			5	
BTR BTS BTC	m,i	3	4	2p06 p23 p4		2	
BSF BSR	r,r	1	1	p1	3	1	
BSF BSR	r,m	1	2	p1 p23		1	
SETcc	r	1	1	p06	1	0.5	
SETcc	m	2	3	p06 p237 p4		1	
CLC		1	0	none		0.25	
STC		1	1	p0156		0.25	
CMC		1	1	p0156	1		
CLD STD		3	3	p15 p6		4	
LZCNT	r,r	1	1	p1	3	1	LZCNT
LZCNT	r,m	1	2	p1 p23		1	LZCNT
TZCNT	r,r	1	1	p1	3	1	BMI1
TZCNT	r,m	1	2	p1 p23		1	BMI1
ANDN	r,r,r	1	1	p15	1	0.5	BMI1
ANDN	r,r,m	1	2	p15 p23	1	0.5	BMI1
BLSI BLSMSK	r,r	1	1	p15 p25	1	0.5	BMI1
BLSR	1,1	'	'		'	0.0	JIVIII
BLSI BLSMSK BLSR	r,m	1	2	p15 p23		0.5	BMI1

1		1	ı	1		ı	1
BEXTR	r,r,r	2	2	2p0156	2	0.5	BMI1
BEXTR	r,m,r	3	3	2p0156 p23		1	BMI1
BZHI	r,r,r	1	1	p15	1	0.5	BMI2
BZHI	r,m,r	1	2	p15 p23		0.5	BMI2
PDEP	r,r,r	1	1	p1	3	1	BMI2
PDEP	r,r,m	1	2	p1 p23		1	BMI2
PEXT	r,r,r	1	1	p1	3	1	BMI2
PEXT		1	2	p1 p23	0	1	BMI2
LXI	r,r,m	'		ρι ρ23		'	DIVIIZ
Control transfer i	instructions						
JMP	short/near	1	1	p6		1-2	
JMP	r	1	1	p6		2	
JMP	m	1	2	p23 p6		2	
Conditional jump	short/near	1	1	p6		1-2	predicted
			-				taken
Conditional jump	short/near	1	1	p06		0.5-1	predicted not taken
Fused arithmetic and branch		1	1	р6		1-2	predicted taken
Fused arithmetic and branch		1	1	p06		0.5-1	predicted not
	ob ort	2	2	n01F6 n6		0.5.2	taken
J(E/R)CXZ	short		2	p0156 p6		0.5-2	
LOOP	short	7	7			5	
LOOP(N)E	short	11	11			6	
CALL	near	2	3	p237 p4 p6		2	
CALL	r	2	3	p237 p4 p6		2	
CALL	m	3	4	2p237 p4 p6		3	
RET		1	2	p237 p6		1	
RET	i	3	4	p23 2p6 p015		2	
BOUND	r,m	15	15	p20 2p0 p0 10		8	not 64 bit
INTO	.,	4	4			5	not 64 bit
String instruc-							
tions							
LODSB/W		3	3	2p0156 p23		1	
LODSD/Q		2	2	p0156 p23		1	
REP LODS		5n+12		' '		~2n	
STOS		3	3	p23 p0156 p4		1	
REP STOS		<2n		P20 P0 100 P1		~0.5n	worst case
REP STOS		2.6/32B				1/32B	best case
KEP 3103		2.0/320				1/320	aligned by 32
MOVS		5	5	2p23 p4 2p0156		4	3.15.15G by 62
REP MOVS		~2n				~1.5 n	worst case
REP MOVS		4/32B				1/32B	best case
1.121 1010 00		7,020				1,020	aligned by 32
SCAS		3	3	p23 2p0156		1	
REP SCAS		≥6n		, ,		≥2n	
CMPS		5	5	2p23 3p0156		4	
REP CMPS		≥8n	_			≥2n	
Synchronization	instructions						
XADD	m,r	4	5			7	
LOCK XADD	m,r	9	9			19	
LOCK ADD	m,r	8	8			19	
LOOK ADD	111,1	0	ı			19	

CMPXCHG	m,r	5	6			8	
LOCK CMPXCHG	m,r	10	10			19	
CMPXCHG8B	m,r	15	15			9	
LOCK CMPXCHG8B	m,r	19	19			19	
CMPXCHG16B	m,r	22	22			15	
LOCK CMPXCHG16B	m,r	24	24			25	
Other							
NOP (90)		1	0	none		0.25	
Long NOP (0F		1	0	none		0.25	
1F)							
PAUSE		5	5	p05 3p6		9	
ENTER	a,0	12	12			8	
ENTER	a,b	~14+7b	~45+7b		~87+2b		
LEAVE		3	3	2p0156 p23		6	
XGETBV		8	8			9	XGETBV
RDTSC		15	15			24	
RDPMC		34	34			37	
RDRAND	r	17	17	p23 16p0156		~320	RDRAND

Floating point x87 instructions

Instruction	Operands	μορs fused domain	µops unfused domain	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc- tions	o por unuo			proposition por		Por	
FLD	r	1	1	p01	1	0.5	
FLD	m32/64	1	1	p23	3	0.5	
FLD	m80	4	4	2p01 2p23	4	2	
FBLD	m80	43	43		47	22	
FST(P)	r	1	1	p01	1	0.5	
FST(P)	m32/m64	1	2	p4 p237	4	1	
FSTP	m80	7	7	3p0156 2p23 2p4	1	5	
FBSTP	m80	238	226			265	
FXCH	r	2	0	none	0	0.5	
FILD	m	1	2	p01 p23	6	1	
FIST(P)	m	3	3	p1 p23 p4	7	1	
FISTTP	m	3	3	p1 p23 p4	7	2	SSE3
FLDZ		1	1	p01		1	
FLD1		2	2	2p01		2	
FLDPI FLDL2E et	tc.	2	2	2p01		2	
FCMOVcc	r	3	3	2p0 p5	2	2	
FNSTSW	AX	2	2	p0 p0156		1	
FNSTSW	m16	2	3	p0 p4 p237	6	1	
FLDCW	m16	3	3	p01 p23 p6	7	2	
FNSTCW	m16	2	3	p237 p4 p6		1	
FINCSTP FDECS	TP	1	1	p01	0	0.5	
FFREE(P)	r	1	1	p01		0.5	
FNSAVE	m	147	147			150	
FRSTOR	m	90	90			164	

Arithmatic in	1	1	1		1	
Arithmetic in- structions						
FADD(P)						
FSUB(R)(P)	r	1	1	p1	3	1
FADD(P)				Pi		
FSUB(R)(P)	m	1	2	p1 p23		1
FMUL(P)	r	1	1	p0	5	1
FMUL(P)		1	2		3	1
	m		1	p0 p23	10.24	8-18
FDIV(R)(P)	r	1		p0	10-24	
FDIV(R)(P)	m	1	2	p0 p23		8-18
FABS		1	1	p0	1	1
FCHS		1	1	p0	1	1
FCOM(P) FUCOM	r	1	1	p1		1
FCOM(P) FUCOM	m	1	2	p1 p23		1
FCOMPP FUCOM		2	2	2p01		1
FCOMI(P) FUCO	r	3	3	3p01		1.5
FIADD FISUB(R)	m	2	3	2p1 p23		2
FIMUL	m	2	3	p0 p1 p23		2
FIDIV(R)	m	2	3	p0 p1 p23		
FICOM(P)	m	2	3	2p1 p23		2
FTST		1	1	p1		1
FXAM		2	2	2p1		2
FPREM		28	28		19	13
FPREM1		41	41		27	17
FRNDINT		17	17		11	23
Math						
FSCALE	1	25-75			49-125	
FXTRACT		17	17		15	11
FSQRT		1	1	р0	10-23	8-17
FSIN		71-100			47-106	
FCOS		110			112	
FSINCOS		70-120			52-123	
F2XM1		58-89			63-68	
FYL2X		55-417			58-680	
FYL2XP1		55-228			58-360	
FPTAN		110-121			130	
FPATAN		78-160			96-156	
FAIAN		70-100			90-100	
Other						
FNOP	1	1	1	p01		0.5
WAIT		2	2	p01		1
FNCLEX		5	5	p0156		22
FNINIT		26	26			83

Integer vector instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μops each port	1	Recipro- cal through put	Comments
Move instruc-							
tions							
MOVD	r32/64,mm/x	1	1	p0	1	1	

MOVD	MOV (D	00/04	l 4		007.4			l I
MOVD	MOVD	m32/64,mm/x	1	2	p237 p4	3	1	
MOVQ		· ·					•	
MOVQ mm/x,r64 MOVQ 1 1 p5 b 1 1 1 MOVQ mm/x,m64 MOVQ 1 1 p15 b 1 1 0.33 may be elim. MOVQ m64, mm/x 1 1 p23 may be elim. 3 1. MOVDQA/U x, m128 movDQA/Q x, m128 movDQA/Q x, m128 movDQA/Q	_		1	1	p23	3	0.5	
MOVQ	MOVQ	r64,mm/x	1	1	p0	1	1	
MOVQ	MOVQ	mm/x,r64	1	1	p5	1	1	
MOVDQA/U	MOVQ	mm/x,mm/x	1		p015	1	0.33	
MOVDQA/U	MOVQ	mm/x.m64	1	1	p23	3	0.5	
MOVDQA/U	MOVQ	· ·	1	2	•	3		
MOVDQA/U x, m128 m128, x 1 1 p23 p237 p4 3 0.5 m23 m23 m23 p237 p4 VMOVDQA/U yy 1 1 p015 p23 p23 p4 3 1 AVX VMOVDQA/U yym256 p25 p23 p4 1 1 p23 p23 p4 4 1 AVX VMOVDQA/U m256,y 1 2 p237 p4 4 1 AVX VMOVDQQU m, m128 1 1 p23 3 0.5 AVX MOVDQ2Q mm, x 2 2 p01 p5 1 1 AVX MOVDQ2Q mm, x 2 2 p01 p5 1 1 AVX MOVNTQ m64, mm 1 2 p237 p4 ~400 1 AVX2 MOVNTDQ m28, x 1 2 p237 p4 ~400 1 AVX2 MOVNTDQ x, m128 1 1 p23 3 0.5 SSE4.1 YMOVNTDQ x, m128 1 1	· ·	· ·	1			_	0.33	may be elim.
MOVDQA/U		· ·	1		•			
VMOVDQA/U VMOVNTQ MC64,mm 1 1 2 p237 p4 4 4 1 1 0.33 May be elim. AVX VMOVDQA/U VMOVDQA/U VMOVNTQ MC64,mm 1 1 1 p015 1 1 1 0.33 May be elim. AVX VMOVDQA/U VMOVDQA/U VMOVDQA/U VMOVNTQ MC64,mm 1 1 2 p237 p4 -400 1 1 MAYA VMOVNTDQ MC256,y 1 2 p237 p4 -400 1 1 MAYA VMOVNTDQA VMOVNTQA VMOVNTDQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VMOVNTQA VX/y,y,y 1 1 1 p5 1 1 SSE4.1 VMVVD/VQA VX/y,y,y 1 1 1 p5 1 1 1 VMVVD/VQA VX/Y,V,V,V 1 1 1 p5 1 1 1 VMVVD/VQA VX/Y,V,V,V 1 1 1 p5 1 1 1 VMVVD/VQA VMOVNTQA VX/Y,V,V,V 1 1 1 p5 3 1 1 SSE4.1 VMVVD/VQA VMVVXZX BW BD BQ DW DQ VMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSE4.1 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSE4.1 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSE4.1 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSE4.1 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSSE3 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSSE3 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSSE3 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSSE3 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSSE3 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSSE3 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSSE3 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5 1 1 SSSE3 VMMOVSX/ZX BW BD BQ DW DQ VX/V,V,V,V 1 1 1 p5		i i						
VMOVDQA/U	MOVDQAVO	111120, X	'		p201 p4		'	Δ\/Υ
VMOVDQA/U y,n256 1 1 p23 3 0.5 AVX VMOVDQA/U m256,y 1 2 p237 p4 4 1 AVX LDDQU x, m128 1 1 p23 3 0.5 SSE3 MOVDQ2Q mm, x 2 2 p01 p5 1 1 MOVDTQ m64,mm 1 2 p237 p4 ~400 1 MOVNTDQ m64,mm 1 2 p237 p4 ~400 1 MOVNTDQA m256,y 1 2 p237 p4 ~400 1 MOVNTDQA x,m128 1 1 p23 3 0.5 SSE4.1 MOVNTDQA x,m256 1 1 p23 3 0.5 SSE4.1 MOVNTDQA x,m256 1 1 p23 3 0.5 SSE4.1 MOVNTDQA x,m258 1 1 p23 3 0.5 SSE4.1 PA	VMOVDQA/U	V V	1	1	p015	0-1	0.33	
VMOVDQA/U m256,y 1 2 p237 p4 4 1 AVX LDDQU x, m128 1 1 p23 3 0.5 SSE3 MOVDQ2Q mm, x 2 2 p01 p5 1 1 MOVDTQ m64,mm 1 2 p237 p4 ~400 1 MOVNTDQ m64,mm 1 2 p237 p4 ~400 1 MOVNTDQ m256,y 1 2 p237 p4 ~400 1 VMOVNTDQA x,m128 1 1 p23 3 0.5 SSE4.1 VMOVNTDQA y,m256 1 1 p23 3 0.5 SSE4.1 VMOVNTDQA y,m256 1 1 p23 3 0.5 SSE4.1 PACKSSWB/DW PACKSSWB/DW mm,m64 3 3 p23 2p5 2 2 PACKUSWB x,x/y,y,y 1 1 p5 1 1 SSE4.1					•			
LDDQU		_			•			
MOVDQ2Q mm, x 2 2 p01 p5 1 1 MOVQZDQ x,mm 1 1 p015 1 0.33 MOVNTQ m64,mm 1 2 p237 p4 ~400 1 MOVNTDQ m128,x 1 2 p237 p4 ~400 1 VMOVNTDQ m256,y 1 2 p237 p4 ~400 1 VMOVNTDQA x, m128 1 1 p23 3 0.5 SSE4.1 VMOVNTDQA x, m128 1 1 p23 3 0.5 SSE4.1 VMOVNTDQA y,m256 1 1 p23 3 0.5 SSE4.1 VMOVNTDQA y,m256 1 1 p23 3 0.5 SSE4.1 VMOVNTDQA y,m256 1 1 p23 3 0.5 SSE4.1 PACKUSWB mm,mm 3 3 p5 2 2 PACKUSWB x,m/y,y,m		_	· ·					
MOVQ2DQ								SSES
MOVNTQ		=						
MOVNTDQ		· ·			•			
VMOVNTDQ m256,y 1 2 p237 p4 ~400 1 AVX2 MOVNTDQA x, m128 1 1 p23 3 0.5 SSE4.1 VMOVNTDQA y,m256 1 1 p23 3 0.5 AVX2 PACKSSWB/DW PACKUSWB mm,mm 3 3 p5 2 2 PACKSSWB/DW PACKUSWB mm,m64 3 3 p23 2p5 2 2 PACKSSWB/DW PACKUSWB x,x/y,y,y 1 1 p5 1 1 PACKSSWB/DW PACKUSWB x,m/y,y,m 1 2 p23 p5 1 2 PACKUSWB x,x/y,y,y 1 1 p5 1 1 SSE4.1 PACKUSWB x,x/y,y,y 1 1 p5 1 1 SSE4.1 PUNPCKH/L BW/WD/DQ v,v/v,v,v 1 1 p5 1 1 PUNPCKH/L QDQ x,x/y,y,y 1 1 p5 1	· ·							
MOVNTDQA		· ·					1	
VMOVNTDQA PACKSSWB/DW PACKUSWB PACKSSWB/DW PACKUSWB PACKSSWB/DW PACKUSWB PACKSSWB/DW PACKUSWB PACKSSWB/DW PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB VAMPY,y,y 1 1 p5 1 1 1 2 <td></td> <td></td> <td>1</td> <td></td> <td></td> <td>~400</td> <td></td> <td></td>			1			~400		
PACKSSWB/DW	MOVNTDQA	x, m128	1	1	p23		0.5	SSE4.1
PACKUSWB	VMOVNTDQA	y,m256	1	1	p23	3	0.5	AVX2
PACKSSWB/DW	PACKSSWB/DW							
PACKUSWB	PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSDW								
PACKUSWB PACKSSWB/DW PACKSSWB/DW PACKUSDW x,x/y,y,y 1 1 p5 1 1 PACKUSDW PACKUSDW x,m/y,y,m 1 2 p23 p5 1 1 SSE4.1 PACKUSDW x,m/y,y,m 1 2 p23 p5 1 1 SSE4.1 PUNPCKH/L BW/WD/DQ v,v/v,v,v 1 1 p5 1 1 PUNPCKH/L QDQ v,m/v,v,m 1 2 p23 p5 1 1 PUNPCKH/L QDQ x,x/y,y,y 1 1 p5 1 1 PUNPCKH/L QDQ x,x/y,y,m 2 2 p23 p5 1 PMOVSX/ZX BW BD BQ DW DQ x,m/y,y,m 2 2 p23 p5 1 PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 VPSHUFB v,v/v,v,v 1 1 p5 1 1 SSSE3 PSH	PACKUSWB	mm,m64	3	3	p23 2p5		2	
PACKSSWB/DW PACKUSWB								
PACKUSWB x,m/y,y,m 1 2 p23 p5 1 1 SSE4.1 PACKUSDW x,x/y,y,y 1 1 p5 1 1 SSE4.1 PACKUSDW x,m/y,y,m 1 2 p23 p5 1 SSE4.1 PUNPCKH/L BW/WD/DQ v,v/v,v,v 1 1 p5 1 1 PUNPCKH/L QDQ v,m/v,v,m 1 2 p23 p5 1 PUNPCKH/L QDQ x,x/y,y,y 1 1 p5 1 1 QDQ x,x/y,y,y 1 1 p5 1 1 1 PUNPCKH/L QDQ x,x/y,y,y 1 1 p5 1 1 1 QDQ x,x/y,y,y 1 1 p5 1 1 1 1 1 1 2 1 2 1 2 1 2 1 3 1 3 1 3 1 3 1<		x,x / y,y,y	1	1	p5	1	1	
PACKUSDW x,x/y,y,y 1 1 p5 1 1 SSE4.1 PACKUSDW x,m/y,y,m 1 2 p23 p5 1 SSE4.1 PUNPCKH/L BW/WD/DQ v,v/v,v,v 1 1 p5 1 1 PUNPCKH/L BW/WD/DQ v,m/v,v,m 1 2 p23 p5 1 1 PUNPCKH/L QDQ x,x/y,y,y 1 1 p5 1 1 PUNPCKH/L QDQ x,x/y,y,y,m 2 2 p23 p5 1 1 PUNPCKH/L QDQ x,x/y,y,y,m 1 1 p5 1 1 PUNPCKH/L QDQ x,x/y,y,y,m 1 1 p5 1 1 PMOVSX/ZX BW BD BQ DW DQ x,m y,x 1 1 p5 1 SSE4.1 PSHUFB v,v/v,v,v 1 1 p5 1 AVX2 PSHUFB v,m/v,v,m 2 2 p23 p5			_	_				
PACKUSDW								
PUNPCKH/L BW/WD/DQ						1		
BW/WD/DQ v,v / v,v,v 1 1 p5 1 1 PUNPCKH/L BW/WD/DQ v,m / v,v,m 1 2 p23 p5 1 PUNPCKH/L QDQ x,x / y,y,y 1 1 p5 1 1 PUNPCKH/L QDQ x,m / y,y,m 2 2 p23 p5 1 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 1 p5 1 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 VPSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW T 1 1 p23 p5<		x,m / y,y,m	1	2	p23 p5		1	SSE4.1
PUNPCKH/L BW/WD/DQ v,m / v,v,m 1 2 p23 p5 1 PUNPCKH/L QDQ x,x / y,y,y 1 1 p5 1 1 PUNPCKH/L QDQ x,m / y,y,m 2 2 p23 p5 1 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 1 p5 1 1 SSE4.1 PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW PSHUFD v,v,i 1 1 p5 1 1 1					_			
BW/WD/DQ v,m / v,v,m 1 2 p23 p5 1 PUNPCKH/L QDQ x,x / y,y,y 1 1 p5 1 1 PUNPCKH/L QDQ x,m / y,y,m 2 2 p23 p5 1 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 1 p5 1 1 SSE4.1 PMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 SSSE3 PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 p5 1 1 p5 1 p5 1 p5 1		V,V / V,V,V	1	1	p5	1	1	
PUNPCKH/L QDQ x,x/y,y,y 1 1 p5 1 1 PUNPCKH/L QDQ x,m/y,y,m 2 2 p23 p5 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 1 p5 1 1 SSE4.1 PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v/v,v,v 1 1 p5 1 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 SSSE3 PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 1		,						
QDQ x,x/y,y,y 1 1 p5 1 1 PUNPCKH/L QDQ x,m/y,y,m 2 2 p23 p5 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 1 p5 1 1 PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 VPSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD p,v,i 1 1 1 p5 1 1		v,m / v,v,m	1	2	p23 p5		1	
PUNPCKH/L QDQ x,m / y,y,m 2 2 p23 p5 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 1 p5 1 1 SSE4.1 PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 SSSE3 PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFW PSHUFD 1 1 1 p5 1 <			4	4	F	4	4	
QDQ x,m / y,y,m 2 2 p23 p5 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 1 p5 1 1 SSE4.1 PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFW PSHUFD 1 1 1 p5 1		x,x / y,y,y	1	1	р5	1	1	
PMOVSX/ZX BW BD BQ DW DQ x,x 1 1 p5 1 1 SSE4.1 PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1		, m / , , , , m	2	2	n02 nE		4	
BD BQ DW DQ x,x 1 1 p5 1 1 SSE4.1 PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1		X,III / Y,Y,III			pzა ps		1	
PMOVSX/ZX BW BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 PSHUFD PSHUFD v,v,i 1 1 p5 1 1		V V	1	1	n5	1	1	SSF4 1
BD BQ DW DQ x,m 1 2 p23 p5 1 SSE4.1 VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1		^,^	1	Į.	ρō	1	'	00L4.1
VPMOVSX/ZX BW BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1		x.m	1	2	p23 p5		1	SSE4.1
BD BQ DW DQ y,x 1 1 p5 3 1 AVX2 VPMOVSX/ZX BW BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,v 1 1 p5 1 1 SSSE3 PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 PSHUFD v,v,i 1 1 p5 1 1		,	-	_	F=- F			
BD BQ DW DQ y,m 2 2 p5 p23 1 AVX2 PSHUFB v,v / v,v,w 1 1 p5 1 1 SSSE3 PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 PSHUFD v,v,i 1 1 p5 1 1		y,x	1	1	p5	3	1	AVX2
PSHUFB v,v/v,v,v 1 1 p5 1 1 SSSE3 PSHUFB v,m/v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 PSHUFD v,v,i 1 1 p5 1 1		_			·			
PSHUFB v,m / v,v,m 2 2 p23 p5 1 SSSE3 PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1		y,m	2		p5 p23		1	
PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 PSHUFD v,v,i 1 1 p5 1 1	PSHUFB	v,v / v,v,v	1	1	p5	1	1	SSSE3
PSHUFW mm,mm,i 1 1 p5 1 1 PSHUFW mm,m64,i 2 2 p23 p5 1 PSHUFD v,v,i 1 1 p5 1 1	PSHUFB	v,m / v,v,m	2	2	p23 p5		1	SSSE3
PSHUFW mm,m64,i 2 2 p23 p5 1 PSHUFD v,v,i 1 1 p5 1 1	PSHUFW	mm,mm,i	1	1	p5	1	1	
PSHUFD v,v,i 1 1 p5 1 1	PSHUFW	mm,m64,i	2	2	p23 p5		1	
	PSHUFD		1	1		1	1	
	PSHUFD	v,m,i	2	2	p23 p5		1	

PSHUFL/HW	v,v,i	1	1	p5	1	1	
PSHUFL/HW	v,m,i	2	2	p23 p5		1	
PALIGNR	v,v,i / v,v,v,i	1	1	p5	1	1	SSSE3
PALIGNR	v,m,i / v,v,m,i	2	2	p23 p5		1	SSSE3
PBLENDVB	x,x,xmm0	2	2	2p5	2	2	SSE4.1
PBLENDVB	x,m,xmm0	3	3	2p5 p23		2	SSE4.1
VPBLENDVB	V,V,V,V	2	2	2p5	2	2	AVX2
VPBLENDVB	v,v,m,v	3	3	2p5 p23		2	AVX2
PBLENDW	x,x,i / v,v,v,i	1	1	p5	1	1	SSE4.1
PBLENDW	x,m,i / v,v,m,i	2	2	p23 p5		1	SSE4.1
VPBLENDD	v,v,v,i	1	1	p015	1	0.33	AVX2
VPBLENDD	v,v,m,i	2	2	p015 p23		0.5	AVX2
VPERMD	y,y,y	1	1	p5	3	1	AVX2
VPERMD	y,y,m	1	2	p5 p23		1	AVX2
VPERMQ	y,y,i	1	1	p5	3	1	AVX2
VPERMQ	y,m,i	2	2	p5 p23		1	AVX2
VPERM2I128	y,y,y,i	1	1	p5	3	1	AVX2
VPERM2I128	y,y,m,i	2	2	p5 p23	-	1	AVX2
MASKMOVQ	mm,mm	4	4	p0 p4 2p23	13-413	1	/ / / /
MASKMOVDQU	x,x	10	10	4p04 2p56 4p23	14-438	6	
VPMASKMOVD/Q	v,v,m	3	3	p23 2p5	4	2	AVX2
VPMASKMOVD/Q	m,v,v	4	4	p0 p1 p4 p23	13-14	1	AVX2
PMOVMSKB	r,v	1	1	p0	3	1	/(//\2
PEXTRB/W/D/Q	r32,x,i	2	2	p0 p5	2	1	SSE4.1
PEXTRB/W/D/Q	m8,x,i	2	3	p23 p4 p5	_	1	SSE4.1
VEXTRACTI128	x,y,i	1	1	p5	3	1	AVX2
VEXTRACTI128	m,y,i	2	2	p23 p4	4	1	AVX2
PINSRB	x,r32,i	2	2	p5	2	2	SSE4.1
PINSRB	x,n8,i	2	2	p23 p5	2	1	SSE4.1
PINSRW	mm/x,r32,i	2	2	p23 p3	2	2	33L4.1
PINSRW	mm/x,m16,i	2	2	p23 p5	2	1	
PINSRD/Q	x,r32,i	2	2	p23 p3	2	2	SSE4.1
PINSRD/Q	x,n32,i x,m32,i	2	2	p23 p5	2	1	SSE4.1
VINSERTI128		1	1		3	1	AVX2
VINSERTI128	y,y,x,i	_		p5		0.5	AVX2 AVX2
VPBROADCAST	y,y,m,i	2	2	p015 p23	4	0.5	AVA2
B/W/D/Q	x,x	1	1	p5	1	1	AVX2
VPBROADCAST	,		-			-	
B/W	x,m8/16	3	3	p01 p23 p5	5	1	AVX2
VPBROADCAST					_		
D/Q	x,m32/64	1	1	p23	4	0.5	AVX2
VPBROADCAST		_	_		2	4	A) ()/O
B/W/D/Q	y,x	1	1	p5	3	1	AVX2
VPBROADCAST B/W	y,m8/16	3	3	p01 p23 p5	7	1	AVX2
VPBROADCAST	y,1110/10			po 1 p20 p0	•		/(///2
D/Q	y,m32/64	1	1	p23	5	0.5	AVX2
VBROADCASTI128	y,m128	1	1	p23	3	0.5	AVX2
VPGATHERDD	x,[r+s*x],x	20	20	·		9	AVX2
VPGATHERDD	y,[r+s*y],y	34	34			12	AVX2
VPGATHERQD	x,[r+s*x],x	15	15			8	AVX2
VPGATHERQD	x,[r+s*y],x	22	22			7	AVX2
VPGATHERDQ	x,[r+s*x],x	12	12			7	AVX2
VPGATHERDQ	y,[r+s*x],y	20	20			9	AVX2
VPGATHERQQ	x,[r+s*x],x	14	14			7	AVX2
1	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ı	I .	1		1	ı I

VPGATHERQQ	y,[r+s*y],y	22	22			9	AVX2
Arithmetic in-							
structions							
PADD/SUB(S,US)	,	_		4.5		0.5	
B/W/D/Q	v,v / v,v,v	1	1	p15	1	0.5	
PADD/SUB(S,US) B/W/D/Q	v,m / v,v,m	1	2	p15 p23		0.5	
PHADD(S)W/D	V,1117 V,V,111	'	_	p 10 p20		0.0	
PHSUB(S)W/D	v,v / v,v,v	3	3	p1 2p5	3	2	SSSE3
PHADD(S)W/D	, , ,			' '			
PHSUB(S)W/D	v,m / v,v,m	4	4	p1 2p5 p23		2	SSSE3
PCMPEQB/W/D							
PCMPGTB/W/D	v,v / v,v,v	1	1	p15	1	0.5	
PCMPEQB/W/D		_		45.00		0.5	
PCMPGTB/W/D	v,m / v,v,m	1	2	p15 p23		0.5	00544
PCMPEQQ	v,v / v,v,v	1	1	p15	1	0.5	SSE4.1
PCMPEQQ	v,m / v,v,m	1	2	p15 p23	_	0.5	SSE4.1
PCMPGTQ	v,v / v,v,v	1	1	p0	5	1	SSE4.2
PCMPGTQ	v,m / v,v,m	1	2	p0 p23		1	SSE4.2
PMULL/HW PMULHUW	v,v / v,v,v	1	1	p0	5	1	
PMULL/HW	V,V / V,V,V	l I	, I	po]	I	
PMULHUW	v,m / v,v,m	1	2	p0 p23		1	
PMULHRSW	v,v / v,v,v	1	1	p0	5	1	SSSE3
PMULHRSW	v,m / v,v,m	1	2	p0 p23		1	SSSE3
PMULLD	x,x / y,y,y	2	2	2p0	10	2	SSE4.1
PMULLD	x,m / y,y,m	3	3	2p0 p23		2	SSE4.1
PMULDQ	x,x / y,y,y	1	1	p0	5	1	SSE4.1
PMULDQ	x,m / y,y,m	1	2	p0 p23		1	SSE4.1
PMULUDQ	v,v / v,v,v	1	1	p0	5	1	
PMULUDQ	v,m / v,v,m	1	2	p0 p23		1	
PMADDWD	v,v / v,v,v	1	1	p0	5	1	
PMADDWD	v,m / v,v,m	1	2	p0 p23		1	
PMADDUBSW	v,v / v,v,v	1	1	p0	5	1	SSSE3
PMADDUBSW	v,m / v,v,m	1	2	p0 p23		1	SSSE3
PAVGB/W	v,v / v,v,v	1	1	p15	1	0.5	
PAVGB/W	v,m / v,v,m	1	2	p15 p23		0.5	
PMIN/PMAX							
SB/SW/SD							
UB/UW/UD	x,x / y,y,y	1	1	p15	1	0.5	SSE4.1
PMIN/PMAX							
SB/SW/SD UB/UW/UD	x,m / y,y,m	1	2	p15 p23		0.5	SSE4.1
PHMINPOSUW	X,111 / Y, Y, 1111	1	1	p13 p23	5	1	SSE4.1
PHMINPOSUW	x,m128	1	2	p0 p23		1	SSE4.1
PABSB/W/D	V,V	1	1	p15	1	0.5	SSSE3
PABSB/W/D	v,w v,m	1	2	p15 p23	'	0.5	SSSE3
PSIGNB/W/D	v,v / v,v,v	1	1	p15	1	0.5	SSSE3
PSIGNB/W/D	v,m / v,v,m	1	2	p15 p23		0.5	SSSE3
PSADBW	v,v / v,v,v	1	1	p10 p20	5	1	20000
PSADBW	v,m / v,v,m	1	2	p0 p23		1	
MPSADBW	x,x,i / v,v,v,i	3	3	p0 2p5	6	2	SSE4.1
MPSADBW	x,m,i / v,v,m,i	4	4	p0 2p5 p23		2	SSE4.1
				' ' '			
•	•			•			

Logic instruc-			I				
tions							
PAND PANDN							
POR PXOR	v,v / v,v,v	1	1	p015	1	0.33	
PAND PANDN							
POR PXOR	v,m / v,v,m	1	2	p015 p23		0.5	
PTEST	V,V	2	2	p0 p5	2	1	SSE4.1
PTEST	v,m	2	3	p0 p5 p23		1	SSE4.1
PSLLW/D/Q							
PSRLW/D/Q							
PSRAW/D/Q	mm,mm	1	1	p0	1	1	
PSLLW/D/Q							
PSRLW/D/Q	0.4	_		0.00		_	
PSRAW/D/Q	mm,m64	1	2	p0 p23		1	
PSLLW/D/Q							
PSRLW/D/Q PSRAW/D/Q	x,x / v,v,x	2	2	p0 p5	2	1	
PSLLW/D/Q	,,			ρυ ρυ	۷	ľ	
PSRLW/D/Q							
PSRAW/D/Q	x,m / v,v,m	2	2	p0 p23		1	
PSLLW/D/Q	7,, 1,1,	_	_	P 0 P 2			
PSRLW/D/Q							
PSRAW/D/Q	v,i / v,v,i	1	1	p0	1	1	
VPSLLVD/Q							
VPSRAVD							
VPSRLVD/Q	V,V,V	3	3	2p0 p5	2	2	AVX2
VPSLLVD/Q							
VPSRAVD		_				_	
VPSRLVD/Q	v,v,m	4	4	2p0 p5 p23		2	AVX2
PSLLDQ	., .	_		_	4	_	
PSRLDQ	x,i / v,v,i	1	1	p5	1	1	
String instruc							
String instruc- tions							
PCMPESTRI	x,x,i	8	8	6p05 2p16	11	4	SSE4.2
PCMPESTRI	x,m128,i	8	8	3p0 2p16 2p5 p23	11	4	SSE4.2
PCMPESTRM	x,111120,1 x,x,i	9	9	3p0 2p16 2p3 p23	10	5	SSE4.2
PCMPESTRM	x,m128,i	9	9	6p05 2p16 p23	10	5	SSE4.2
PCMPISTRI		3	3		11	3	SSE4.2 SSE4.2
	X,X,İ	3 4	4	3p0	11	3	SSE4.2 SSE4.2
PCMPISTRI	x,m128,i	3	3	3p0 p23	40	3	SSE4.2 SSE4.2
PCMPISTRM	X,X,İ	3 4	4	3p0	10	3	
PCMPISTRM	x,m128,i	4	4	3p0 p23		3	SSE4.2
Encryption instru	uctions						
PCLMULQDQ	x,x,i	3	3	2p0 p5	7	2	CLMUL
PCLMULQDQ	x,m,i	4	4	2p0 p5 p23	,	2	CLMUL
AESDEC,	Α,ΙΙΙ,Ι	7	•	2ρ0 ρ3 ρ23			CLIVIOL
AESDEC, AESDECLAST,							
AESENC,							
AESENCLAST	x,x	1	1	p5	7	1	AES
AESDEC,	, , , , , , , , , , , , , , , , , , ,			'			
AESDECLAST,							
AESENC,							
AESENCLAST	x,m	2	2	p5 p23		1.5	AES
AESIMC	x,x	2	2	2p5	14	2	AES
AESIMC	x,m	3	3	2p5 p23		2	AES

AESKEYGENAS SIST AESKEYGENAS SIST	x,x,i x,m,i	10 10	10	2p0 8p5 2p0 p23 7p5	10	9 8	AES AES	
Other EMMS		31	31			13		

Floating point	Operands	µops fused domain	μορs unfused	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc-	Operanus	uomam	uomam	pops each port	Latericy	put	Comments
tions							
MOVAPS/D	x,x	1	1	p5	0-1	1	may be elim.
VMOVAPS/D	y,y	1	1	p5	0-1	1	may be elim.
MOVAPS/D	3,73			'			
MOVUPS/D	x,m128	1	1	p23	3	0.5	
VMOVAPS/D							
VMOVUPS/D	y,m256	1	1	p23	3	0.5	AVX
MOVAPS/D			_		_		
MOVUPS/D	m128,x	1	2	p237 p4	3	1	
VMOVAPS/D		4		m227 m4	4	4	A) (V
VMOVUPS/D	m256,y	1	2	p237 p4	4	1	AVX
MOVSS/D	X,X	1	1	p5	1	1	
MOVSS/D	x,m32/64	1	1	p23	3	0.5	
MOVSS/D	m32/64,x	1	2	p237 p4	3	1	
MOVHPS/D	x,m64	1	2	p23 p5	4	1	
MOVHPS/D	m64,x	1	2	p4 p237	3	1	
MOVLPS/D	x,m64	1	2	p23 p5	4	1	
MOVLPS/D	m64,x	1	2	p4 p237	3	1	
MOVHLPS	X,X	1	1	p5_	1	1	
MOVLHPS	X,X	1	1	p5	1	1	
MOVMSKPS/D	r32,x	1	1	p0	3	1	
VMOVMSKPS/D	r32,y	1	1	p0	2	1	
MOVNTPS/D	m128,x	1	2	p4 p237	~400	1	
VMOVNTPS/D	m256,y	1	2	p4 p237	~400	1	AVX
SHUFPS/D	x,x,i / v,v,v,i	1	1	p5	1	1	
SHUFPS/D	x,m,i / v,v,m,i	2	2	p5 p23		1	
VPERMILPS/PD	v,v,i	1	1	p5	1	1	AVX
VPERMILPS/PD	v,m,i	2	2	p5 p23		1	AVX
VPERMILPS/PD	V,V,V	1	1	p5	1	1	AVX
VPERMILPS/PD	v,v,m	2	2	p5 p23		1	AVX
VPERM2F128	y,y,y,i	1	1	p5	3	1	AVX
VPERM2F128	y,y,m,i	2	2	p5 p23		1	AVX
VPERMPS	y,y,y	1	1	p5	3	1	AVX2
VPERMPS	y,y,m	1	2	p5 p23		1	AVX2
VPERMPD	y,y,i	1	1	p5	3	1	AVX2
VPERMPD	y,m,i	2	2	p5 p23		1	AVX2
BLENDPS/PD	x,x,i / v,v,v,i	1	1	p015	1	0.33	SSE4.1
BLENDPS/PD	x,m,i / v,v,m,i	2	2	p015 p23		0.5	SSE4.1
BLENDVPS/PD	x,x,xmm0	2	2	2p5	2	2	SSE4.1
BLENDVPS/PD	x,m,xmm0	3	3	2p5 p23		2	SSE4.1

VBLENDVPS/PD	V,V,V,V	2	2	2p5	2	2	AVX
VBLENDVPS/PD	v,v,m,v	3	3	2p5 p23		2	AVX
MOVDDUP	V,V	1	1	p5	1	1	SSE3
MOVDDUP	v,m	1	1	p23	3	0.5	SSE3
VBROADCASTSS	x,m32	1	1	p23	4	0.5	AVX
VBROADCASTSS	y,m32	1	1	p23	5	0.5	AVX
VBROADCASTSS	x,x	1	1	p5	1	1	AVX2
VBROADCASTSS	y,x	1	1	p5	3	1	AVX2
VBROADCASTSD	y,m64	1	1	p23	5	0.5	AVX
VBROADCASTSD	y,x	1	1	p5	3	1	AVX2
VBROADCASTF128	y,m128	1	1	p23	3	0.5	AVX
MOVSH/LDUP	V,V	1	1	p5	1	1	SSE3
MOVSH/LDUP	v,m	1	1	p23	3	0.5	SSE3
UNPCKH/LPS/D	x,x / v,v,v	1	1	p5	1	1	SSE3
UNPCKH/LPS/D	x,m / v,v,m	1	2	p5 p23		1	SSE3
EXTRACTPS	r32,x,i	2	2	p0 p5		1	SSE4.1
EXTRACTPS	m32,x,i	3	3	p0 p5 p23	4	1	SSE4.1
VEXTRACTF128	x,y,i	1	1	p5	3	1	AVX
VEXTRACTF128	m128,y,i	2	2	p23 p4	4	1	AVX
INSERTPS	x,x,i	1	1	p5	1	1	SSE4.1
INSERTPS	x,m32,i	2	2	p23 p5	4	1	SSE4.1
VINSERTF128	y,y,x,i	1	1	p5	3	1	AVX
VINSERTF128	y,y,m128,i	2	2	p015 p23	4	2	AVX
VMASKMOVPS/D	v,v,m	3	3	2p5 p23	4	2	AVX
VMASKMOVPS/D	m128,x,x	4	4	p0 p1 p4 p23	13	1	AVX
VMASKMOVPS/D	m256,y,y	4	4	p0 p1 p4 p23	14	2	AVX
VGATHERDPS	x,[r+s*x],x	20	20	ρο ρι ρτ ρ2ο	17	9	AVX2
VGATHERDIS	y,[r+s*y],y	34	34			12	AVX2
VGATHEROPS	x,[r+s*x],x	15	15			8	AVX2
VGATHERQPS	x,[r+s*y],x	22	22			7	AVX2
VGATHERQFO	x,[r+s*x],x	12	12			7	AVX2
VGATHERDPD	y,[r+s*x],y	20	20			9	AVX2
VGATHEROPD	x,[r+s*x],x	14	14			7	AVX2
VGATHERQPD	y,[r+s*y],y	22	22			9	AVX2
VOATTIERQI D	y,[i · 3 y],y						AVAZ
Conversion							
CVTPD2PS	x,x	2	2	p1 p5	4	1	
CVTPD2PS	x,m128	2	3	p1 p5 p23		1	
VCVTPD2PS	x,y	2	2	p1 p5	5	1	AVX
VCVTPD2PS	x,m256	2	3	p1 p5 p23		1	AVX
CVTSD2SS	x,x	2	2	p1 p5	4	1	
CVTSD2SS	x,m64	2	3	p1 p5 p23		1	
CVTPS2PD	x,x	2	2	p0 p5	2	1	
CVTPS2PD	x,m64	2	2	p0 p23		1	
VCVTPS2PD	y,x	2	2	p0 p5	5	1	AVX
VCVTPS2PD	y,m128	2	2	p0 p23	_	1	AVX
CVTSS2SD	x,x	2	2	p0 p5	2	1	
CVTSS2SD	x,m32	2	2	p0 p23	_	1	
CVTDQ2PS	x,me2 x,x	1	1	p1	3	1	
CVTDQ2PS	x,m128	1	2	p1 p23		1	
VCVTDQ2PS	y,y		1	p1	3		AVX
VCVTDQ2FS	y,y y,m256	1	2	p1 p23		1	AVX
CVT(T) PS2DQ	x,x	1	1	p1	3	1	'''
(.) . OZDQ	7,7			۲.			I

CVT(I) PSZDQ	O\ (T(T) DOOD O	400	1 4	1 0		I	1 4	
VCVTI(T) PSZDQ	CVT(T) PS2DQ	x,m128	1	2	p1 p23		1	4.00
CVTDQ2PD			1			3	· ·	
CVTDQ2PD VCVTDQ2PD VCVTDQ2PD VCVTDQ2PD VCVTDQ2PD VCVTQ1PD2DQ VCVTQ1PD2PQ VCVTQ1PQ1PQ1PQ1PQ1PQ1PQ1PQ1PQ1PQ1PQ1PQ1PQ1PQ		y,m256					1	AVX
\text{VCVTDQ2PD \text{VX} \text{ VX} \text{ V2}	CVTDQ2PD	X,X			p1 p5	4	1	
VCVTDQ2PD y,m128 2 2 p1p23 4 1 CVT(T)PD2DQ x,m128 2 2 p1p5 p23 1 VCVT(T)PD2DQ x,m256 2 3 p1p5p23 1 VCVT(T)PD2DQ x,m256 2 3 p1p5p23 1 VCVT(P1PSES x,mm 1 1 p1 4 4 CVT(P1PSEP) x,m64 1 2 p1p23 3 CVT(P1PSEP) mm, 2 2 2 p1p5 4 1 CVT(P1PSEP) mm, 2 2 2 p1p5 4 1 CVT(P1PSEP) mm, m128 2 2 p1p5 4 1 CVT(P1PDED x,m64 2 2 p1p5 4 1 CVT(P1PSEP) x,m64 2 2 p1p5 4 1 CVT(P1PSEP) x,m64 2 2 p1p5 4 1 CVT(P1PSES) mm, 128 2<	CVTDQ2PD	x,m64	2		p1 p23		1	
CVT(T)PD2DQ	VCVTDQ2PD	y,x	2	2	p1 p5	6	1	AVX
CVT(T)PD2DQ	VCVTDQ2PD	-	2	2			1	AVX
CVT(T)PD2DQ			2	2		4	1	
VCVT(T)PD2DQ x,y 2 2 p1p5 6 1 AVX VCVT(T)PD2DQ x,m256 2 3 p1p5p23 1 AVX CVTPI2PS x,m64 1 2 p1p23 3 3 CVT(T)PS2PI mm,x 2 2 p1p5 4 1 CVTPI2PD mm,x 2 2 p1p5 4 1 CVTPI2PD x,m64 2 2 p1p5 4 1 CVTPI2PD x,m64 2 2 p1p5 4 1 CVTT(PD2PI mm,x 2 2 p1p5 4 1 CVT(T) PD2PI mm,x 2 2 p1p5 4 1 CVT(T)PD2PI mm,x 2 2 p1p5 4 1 CVT(T)SD2SI x,x32 2 2 p1p5 4 3 CVTSI2SS x,32,2 2 2 p1p5 4 3 <tr< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td></tr<>							1	
VCVT(T)PD2DQ						6	1	ΔVX
CVTPI2PS		-						
CVTPI2PS						4		AVA
CVT(T)PS2PI						-		
CVT(T)PS2PI mm,m128 2 2 p1 p23 1 CVTPI2PD x,m6 2 2 p1 p5 4 1 CVT(T) PD2PI x,m64 2 2 p1 p5 4 1 CVT(T) PD2PI mm,x 2 2 p1 p5 4 1 CVT(T) PD2PI mm,x 2 2 p1 p5 4 1 CVT(T) PD2PI mm,m128 2 3 p1 p5 p23 1 1 CVT(S) S2SI x,x32 2 2 p1 p23 3 1 CVT(T)SS2SI r32,x 2 2 p0 p1 4 1 CVT(T)SS2SI r32,m32 2 2 p1 p5 4 3 CVTSI2SD x,r32/64,x 2 2 p1 p5 4 1 CVT(T)SD2SI r32,m64 2 2 p1 p5 4 1 F16C CVT(T)SP2PH x,v,i 2 2 p1 p5 4 1 <td></td> <td></td> <td></td> <td></td> <td></td> <td>4</td> <td></td> <td></td>						4		
CVTPI2PD x,m64 2 2 p1 p5 4 1 CVT(T) PD2PI x,m64 2 2 p1 p5 4 1 CVT(T) PD2PI mm,x 2 2 p1 p5 4 1 CVT(T) PD2PI mm,m128 2 3 p1 p5 p23 1 CVTSI2SS x,r32 2 2 p1 p5 4 3 CVT(T)S2SSI r32,x 2 2 p0 p1 4 1 CVT(T)S2SI r32,m32 2 2 p0 p1 p23 1 CVT(SI2SD x,r32/64 2 2 p1 p23 3 CVT(T)SD2SI r32/e64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32/e64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32/e64,x 2 2 p1 p5 4 1 F16C CVT(T)SD2SI r32/e64 2 3 p0 p1 p23 1 T6C						4		
CVTPI2PD							· .	
CVT(T) PD2PI mm,x 2 2 p1 p5 4 1 CVT(T) PD2PI mm,m128 2 3 p1 p5 p23 1 CVTSI2SS x,r32 2 2 p1 p5 4 3 CVT(T)SS2SI r32,x 2 2 p0 p1 4 1 CVT(T)SS2SI r32,m32 2 3 p0 p1 p23 1 1 CVT(S)S2SI r32,m32 2 2 p1 p5 4 3 CVT(S)S2SI r32,64 2 2 p1 p5 4 3 CVTSI2SD x,m32 2 2 p1 p23 3 1 CVT(S)S2SI r32,64 2 2 p0 p1 4 1 1 CVT(T)SD2SI r32,64 2 3 p0 p1 p23 1 1 CVT(T)SD2SI r32,644 2 3 p0 p1 p23 1 1 CVT(T)SD2SI r32,m64 2 3 p0 p1 p23 1						4	· ·	
CVT(T) PD2PI mm,m128 2 3 p1 p5 p23 1 CVTSI2SS x,r32 2 2 p1 p5 4 3 CVTSI2SS x,r32 1 2 p1 p23 3 3 CVT(T)SS2SI r32,xx 2 2 p0 p1 4 1 CVT(T)S2SD x,r32/64 2 2 p1 p5 4 3 CVT(T)SDSI r32/64,x 2 2 p1 p23 1 CVT(T)SDSI r32/64,x 2 2 p0 p1 4 1 CVT(TS2PBI x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH x,v,i 2 2 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p23 1 F16C ADDSUBPS/D x,x / v,v,v 1 1 p1 p23 1							· ·	
CVTSI2SS x,r32 2 2 p1 p5 4 3 CVTSI2SS x,m32 1 2 p1 p23 3 CVT(T)SS2SI r32,x 2 2 p0 p1 4 1 CVT(T)SS2SI r32,m32 2 3 p0 p1 p23 1 1 CVTSI2SD x,r32/64 2 2 p1 p5 4 3 3 CVTSI2SD x,m32 2 2 p1 p5 4 3 CVTSI2SD x,m32/64,x 2 2 p0 p1 4 1 1 P1 p23 3 1 VCVT(T)SD2SI r32/64,x 2 2 p0 p1 p23 1 1 P1 p6 4 1 F16C VCVT(T)S2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 <td></td> <td></td> <td></td> <td></td> <td></td> <td>4</td> <td>· ·</td> <td></td>						4	· ·	
CVTSI2SS x,m32 1 2 p1 p23 3 CVT(T)SS2SI r32,x 2 2 p0 p1 4 1 CVT(T)SS2SI r32,m32 2 3 p0 p1 p23 1 CVTSI2SD x,r32/64 2 2 p1 p5 4 3 CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32,m64 2 3 p0 p1 p23 1 CVT(T)SD2SI r32,m64 2 3 p0 p1 p23 1 VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH x,v,i 4 4 p1 p4 p5 p23 1 F16C VCVTPB2PS v,x 2 2 p1 p5 4 1 F16C VCVTPB2PS v,x 2 2 p1 p23 1 F16C Arithmetic ADDSS/D PS/D x,x / v,v,v 1 1 p1 p23 1		mm,m128			p1 p5 p23			
CVT(T)SS2SI r32,x 2 2 p0 p1 4 1 CVT(T)SS2SI r32,m32 2 3 p0 p1 p23 1 CVTSI2SD x,r32/64 2 2 p1 p5 4 3 CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32/m64 2 3 p0 p1 p23 1 1 VCVTPS2PH x,vi 2 2 p1 p5 4 1 F16C VCVTPB2PB x,vi 4 4 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C Arithmetic ADDSS/D P8/D x,x / v,v,v 1 1 p1 3 1 SSE3 ADDSUBPS/D x,m / v,v,m		x,r32	2			4		
CVT(T)SS2SI r32,m32 2 3 p0 p1 p23 1 CVTSI2SD x,r32/64 2 2 p1 p5 4 3 CVTSI2SD x,m32 2 2 p1 p23 3 3 CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32,m64 2 3 p0 p1 p23 1 VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH m,v,i 4 4 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,m 2 2 p1 p23 1 F16C Arithmetic ADDSUBPS/D x,x / v,v,v 1 1 p1 3 1 SSE3 HADDSBPS/D x,m / v,v,m 1 <td>CVTSI2SS</td> <td>x,m32</td> <td>1</td> <td>2</td> <td>p1 p23</td> <td></td> <td>3</td> <td></td>	CVTSI2SS	x,m32	1	2	p1 p23		3	
CVT(T)SS2SI r32,m32 2 3 p0 p1 p23 1 CVTSI2SD x,r32/64 2 2 p1 p5 4 3 CVTSI2SD x,m32 2 2 p1 p23 3 3 CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32,m64 2 3 p0 p1 p23 1 1 VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH m,v,i 4 4 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p23 1 F16C VCVTPH2PS v,m 2 2 p1 p23 1 F16C Arithmetic ADDSUBPS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D x,m / v,v,m 1	CVT(T)SS2SI	r32,x	2	2		4	1	
CVTSI2SD x,r32/64 2 2 p1 p5 4 3 CVTSI2SD x,m32 2 2 p1 p23 3 CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32/m64 2 3 p0 p1 p23 1 1 VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPH2PS x,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p23 1 F16C VCVTPH2PS v,m 2 2 p1 p23 1 F16C Arithmetic ADDSUBPS/D PS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D NG		r32,m32	2	3	p0 p1 p23		1	
CVTSI2SD x,m32 2 2 p1 p23 3 CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32,m64 2 3 p0 p1 p23 1 VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH m,v,i 4 4 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p23 1 F16C VCVTPH2PS v,m 2 2 p1 p23 1 1 F16C Arithmetic ADDSUBPS/D x,m / v,v,m 1 2 </td <td></td> <td></td> <td>2</td> <td>2</td> <td></td> <td>4</td> <td>3</td> <td></td>			2	2		4	3	
CVT(T)SD2SI r32/64,x 2 2 p0 p1 4 1 CVT(T)SD2SI r32,m64 2 3 p0 p1 p23 1 VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH m,v,i 4 4 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p23 1 F16C VCVTPH2PS v,m 2 2 p1 p23 1 F16C ADDSS/D PS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D x,m / v,v,m 4								
CVT(T)SD2SI r32,m64 2 3 p0 p1 p23 1 VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH m,v,i 4 4 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p23 1 F16C VCVTPH2PS v,m 2 2 p1 p23 1 F16C ACVTPH2PS x,x / v,v,v 1 1 p1 3 1 S2 S2 S2 S2						4		
VCVTPS2PH x,v,i 2 2 p1 p5 4 1 F16C VCVTPS2PH m,v,i 4 4 4 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p23 1 F16C ACVCVTPH2PS v,m 2 2 p1 p23 1 1 P16C ACVCVTPH2PS x,m / v,v,m 1 2 p1 p23 1 1 2 2 S2 S2 3 3 1 S2 S2 S2 3 3 P1 2p5 </td <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>								
VCVTPS2PH m,v,i 4 4 4 p1 p4 p5 p23 1 F16C VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,x 2 2 p1 p23 1 F16C ADDSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D SUBSS/D SUBSS/D SUBSS/D SUBSS/D SUBS/D SUBS/D SUBS/D						1		E16C
VCVTPH2PS v,x 2 2 p1 p5 4 1 F16C VCVTPH2PS v,m 2 2 p1 p23 4 1 F16C Arithmetic ADDSS/D PS/D SUBSS/								
VCVTPH2PS v,m 2 2 p1 p23 1 F16C Arithmetic ADDSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D X,m / v,v,m x,x / v,v,v 1 1 p1 3 1 SSE3 ADDSUBPS/D PS/D ADDUBPS/D ADDU						4		
Arithmetic ADDSS/D PS/D SUBS/D SUB						4		
ADDSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D X,x / v,v,v 1 1 1 p1 3 1 ADDSS/D PS/D SUBSS/D PS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D HSUBPS/D X,x / v,v,v 3 3 p1 2p5 5 2 SSE3 HADDPS/D HSUBPS/D X,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D X,m / v,v,m 1 2 p01 p23 0.5 MULSS/D PS/D X,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS X,m 1 2 p0 p23 7 DIVSD DIVPD X,m 1 2 p0 p23 7 DIVSD DIVPD X,m 1 2 p0 p23 8-14 DIVSD DIVPD X,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 p23 14 AVX VDIVPD y,y,m 256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX	VCV IPHZPS	v,m			p1 p23		I	FIBC
ADDSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D X,x / v,v,v 1 1 1 p1 3 1 ADDSS/D PS/D SUBSS/D PS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D HSUBPS/D X,x / v,v,v 3 3 p1 2p5 5 2 SSE3 HADDPS/D HSUBPS/D X,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D X,m / v,v,m 1 2 p01 5 0.5 MULSS/D PS/D X,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS X,m 1 1 p0 10-13 7 DIVSS DIVPS X,m 1 2 p0 p23 7 DIVSD DIVPD X,m 1 2 p0 p23 7 DIVSD DIVPD X,m 1 2 p0 p23 8-14 DIVSD DIVPD X,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX	Arithmatia							
SUBSS/D PS/D x,x / v,v,v 1 1 p1 3 1 ADDSS/D PS/D x,m / v,v,m 1 2 p1 p23 1 ADDSUBPS/D x,x / v,v,v 1 1 p1 3 1 ADDSUBPS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D x,x / v,v,v 3 3 p1 2p5 5 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 HADDPS/D x,m / v,v,m 1 1 p01 5 0.5 SSE3 MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 O.5 DIVSS DIVPS x,m 1 2 p0 p23 7 O.5 DIVSD DIVPD x,m 1 2								
ADDSS/D PS/D SUBSS/D PS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D X,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D HSUBPS/D X,m / v,v,m 1 2 p1 p25 5 2 SSE3 HADDPS/D HSUBPS/D X,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D X,x / v,v,v 1 1 p01 5 0.5 MULSS/D PS/D X,x / v,v,v 1 1 p01 5 0.5 MULSS/D PS/D X,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS X,x 1 1 p0 10-13 7 DIVSS DIVPS X,m 1 2 p0 p23 7 DIVSD DIVPD X,m 1 2 p0 p23 7 DIVSD DIVPD X,m 1 2 p0 p23 8-14 DIVSD DIVPD X,m 1 2 p0 p23 8-14 VDIVPS Y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS Y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD Y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD Y,y,m256 4 4 2p0 p15 p23 16-28 AVX		v v I v v v	4	4	n1	2	1	
SUBSS/D PS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 ADDSUBPS/D x,x / v,v,v 1 1 p1 3 1 SSE3 ADDSUBPS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D HSUBPS/D x,x / v,v,v 3 3 p1 2p5 5 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D x,x / v,v,v 1 1 p01 p23 0.5 0.5 DIVSS DIVPS x,x 1 1 p0 p23 7 7 DIVSD DIVPD x,m 1 2 p0 p23 8-14 0.5 VDIVPS y,y,m 256 4 4 2p0 p15 p23 14 AVX		X,X / V,V,V	1	ı	ρī	3	ı	
ADDSUBPS/D x,x / v,v,v 1 1 p1 3 1 SSE3 ADDSUBPS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D x,x / v,v,v 3 3 p1 2p5 5 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 0.5 DIVSS DIVPS x,m 1 2 p0 p23 7 0.5 DIVSD DIVPD x,m 1 2 p0 p23 8-14 0.5 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPD <td></td> <td>v m / v v m</td> <td>1</td> <td>2</td> <td>n1 n23</td> <td></td> <td>1</td> <td></td>		v m / v v m	1	2	n1 n23		1	
ADDSUBPS/D x,m / v,v,m 1 2 p1 p23 1 SSE3 HADDPS/D x,x / v,v,v 3 3 p1 2p5 5 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 HSUBPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D x,x / v,v,v 1 1 p01 5 0.5 MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS x,x 1 1 p0 10-13 7 DIVSS DIVPS x,m 1 2 p0 p23 7 DIVSD DIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 <td></td> <td></td> <td></td> <td></td> <td></td> <td>2</td> <td></td> <td>SSE3</td>						2		SSE3
HADDPS/D x,x / v,v,v 3 3 p1 2p5 5 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D x,x / v,v,v 1 1 p01 5 0.5 MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS x,x 1 1 p0 10-13 7 DIVSS DIVPS x,m 1 2 p0 p23 7 DIVSD DIVPD x,x 1 1 p0 10-20 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 19-35 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX						3		
HSUBPS/D x,x / v,v,v 3 3 p1 2p5 5 2 SSE3 HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D x,x / v,v,v 1 1 p01 5 0.5 MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS x,x 1 1 p0 10-13 7 DIVSS DIVPS x,m 1 2 p0 p23 7 DIVSD DIVPD x,x 1 1 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,ym256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX		X,111 / V,V,111	l I		p i p23		Į.	SSES
HADDPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D x,x / v,v,v 1 1 p01 5 0.5 MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS x,x 1 1 p0 10-13 7 DIVSS DIVPS x,m 1 2 p0 p23 7 DIVSD DIVPD x,x 1 1 p0 p23 8-14 VDIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX		y y I y y y	2	2	n1 2n5	5	2	SSE3
HSUBPS/D x,m / v,v,m 4 4 p1 2p5 p23 2 SSE3 MULSS/D PS/D x,x / v,v,v 1 1 p01 5 0.5 MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS x,x 1 1 p0 10-13 7 DIVSD DIVPS x,m 1 2 p0 p23 7 DIVSD DIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX		X,X / V,V,V	ر ا	3	p i zps	5		SSES
MULSS/D PS/D x,x / v,v,v 1 1 p01 5 0.5 MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS x,x 1 1 p0 10-13 7 DIVSD DIVPD x,x 1 1 p0 p23 7 DIVSD DIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 14-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX		v m / v v m	4	4	n1 2n5 n22		2	SSE3
MULSS/D PS/D x,m / v,v,m 1 2 p01 p23 0.5 DIVSS DIVPS x,x 1 1 p0 10-13 7 DIVSD DIVPS x,m 1 2 p0 p23 7 DIVSD DIVPD x,x 1 1 p0 10-20 8-14 DIVSD DIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 14-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX								SSES
DIVSS DIVPS x,x 1 1 p0 10-13 7 DIVSD DIVPD x,m 1 2 p0 p23 7 DIVSD DIVPD x,x 1 1 p0 10-20 8-14 DIVSD DIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 14-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX						5		
DIVSS DIVPS x,m 1 2 p0 p23 7 DIVSD DIVPD x,x 1 1 p0 10-20 8-14 DIVSD DIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 14-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX						40.40		
DIVSD DIVPD x,x 1 1 p0 10-20 8-14 DIVSD DIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 14 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 19-35 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX						10-13		
DIVSD DIVPD x,m 1 2 p0 p23 8-14 VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 14 AVX VDIVPD y,y,m256 4 4 2p0 p15 19-35 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX						45.55		
VDIVPS y,y,y 3 3 2p0 p15 18-21 14 AVX VDIVPS y,y,m256 4 4 2p0 p15 p23 14 AVX VDIVPD y,y,y 3 3 2p0 p15 19-35 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX		X,X	1			10-20		
VDIVPS y,y,m256 4 4 2p0 p15 p23 14 AVX VDIVPD y,y,y 3 3 2p0 p15 19-35 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX		x,m	_					
VDIVPD y,y,y 3 3 2p0 p15 19-35 16-28 AVX VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX		y,y,y				18-21		
VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX	VDIVPS	y,y,m256	4	4	2p0 p15 p23		14	AVX
VDIVPD y,y,m256 4 4 2p0 p15 p23 16-28 AVX	VDIVPD	y,y,y	3	3	2p0 p15	19-35	16-28	AVX
	VDIVPD		4	4			16-28	AVX
	RCPSS/PS	x,x	1	1	p0	5	1	

RCPSS/PS	x,m128	1	2	p0 p23		1	
VRCPPS	y,y	3	3	2p0 p15	7	2	AVX
VRCPPS	y,m256	4	4	2p0 p15 p23		2	AVX
CMPccSS/D	y,111200	7	_	2po p 10 p20		_	7,47
CMPccPS/D	x,x / v,v,v	1	1	p1	3	1	
	X,X / V,V,V	ı	l I	pı	3	ľ	
CMPccSS/D		2	_	m1 m00		4	
CMPccPS/D	x,m / v,v,m	2	2	p1 p23		1	
(U)COMISS/D	X,X	1	1	p1		1	
(U)COMISS/D	x,m32/64	2	2	p1 p23		1	
MAXSS/D PS/D							
MINSS/D PS/D	x,x / v,v,v	1	1	p1	3	1	
MAXSS/D PS/D							
MINSS/D PS/D	x,m / v,v,m	1	2	p1 p23		1	
	, , ,			' '			
ROUNDSS/D PS/D	v,v,i	2	2	2p1	6	2	SSE4.1
	, ,			'			
ROUNDSS/D PS/D	v,m,i	3	3	2p1 p23		2	SSE4.1
DPPS	x,x,i / v,v,v,i	4	4	2p0 p1 p5	14	2	SSE4.1
DPPS	x,m,i / v,v,m,i	6	6	2p0 p1 p5 p23 p6		4	SSE4.1
DPPD		3	3	p0 p1 p5	9	1	SSE4.1
	x,x,i				9		
DPPD	x,m128,i	4	4	p0 p1 p5 p23		1	SSE4.1
VFMADD					_		
(all FMA instr.)	V,V,V	1	1	p01	5	0.5	FMA
VFMADD							
(all FMA instr.)	v,v,m	1	2	p01 p23		0.5	FMA
Math							
SQRTSS/PS	x,x	1	1	p0	11	7	
SQRTSS/PS	x,m128	1	2	p0 p23	• •	7	
VSQRTPS		3	3		19	14	AVX
	у,у			2p0 p15	19		
VSQRTPS	y,m256	4	4	2p0 p15 p23		14	AVX
SQRTSD/PD	X,X	1	1	p0	16	8-14	
SQRTSD/PD	x,m128	1	2	p0 p23		8-14	
VSQRTPD	y,y	3	3	2p0 p15	28-29	16-28	AVX
VSQRTPD	y,m256	4	4	2p0 p15 p23		16-28	AVX
RSQRTSS/PS	x,x	1	1	p0	5	1	
RSQRTSS/PS	x,m128	1	2	p0 p23		1	
	·		3		7		A\/V
VRSQRTPS	у,у	3		2p0 p15	/	2	AVX
VRSQRTPS	y,m256	4	4	2p0 p15 p23		2	AVX
Logic							
AND/ANDN/OR/				_			
XORPS/PD	x,x / v,v,v	1	1	p5	1	1	
AND/ANDN/OR/	,		_				
XORPS/PD	x,m / v,v,m	1	2	p5 p23		1	
Other							
VZEROUPPER		4	4	none		1	AVX
							AVX,
VZEROALL		12	12	none		10	32 bit
							AVX,
VZEROALL		20	20	none		8	64 bit
LDMXCSR	m32	3	3	p0 p6 p23	6	3	
STMXCSR	m32	3	4	p0 p4 p6 p237	7	1	
		3	,	ρυ p-τ ρυ p201	'		A\/~
VSTMXCSR	m32	3	l			1	AVX

FXSAVE	m4096	130	68	
FXRSTOR	m4096	116	72	
XSAVE		224	84	
XRSTOR		173	111	
XSAVEOPT	m			

Intel Broadwell

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the same data.

Instructions with or without V name prefix behave the same unless otherwise noted.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm register, mm/

x = mmx or xmm register, y = 256 bit ymm register, v = any vector register (mmx, xmm, ymm). same = same register for both operands. m = memory operand, m32 = 32-bit

memory operand, etc.

µops fused

The number of µops at the decode, rename and allocate stages in the pipeline. Fused

domain: µops count as one.

μops unfused domain:

The total number of µops for all execution port. Fused µops count as two. Fused macro-

ops count as one. The instruction has µop fusion if this number is higher than the number under fused domain. Some operations are not counted here if they do not go to any

execution port or if the counters are inaccurate.

μορs each port: The number of μορs for each execution port. p0 means a μορ to execution port 0.

p01means a μop that can go to either port 0 or port 1. p0 p1 means two μops going to

port 0 and 1, respectively.

Port 0: Integer, f.p. and vector ALU, mul, div, branch

Port 1: Integer, f.p. and vector ALU

Port 2: Load Port 3: Load Port 4: Store

Port 5: Integer and vector ALU Port 6: Integer ALU, branch

Port 7: Store address

Latency: This is the delay that the instruction generates in a dependency chain. The numbers are

minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Where hyperthreading is enabled, the use of the same execution units in the other thread leads to inferior performance. Denormal numbers, NAN's and infinity do not increase the latency. The time unit used is core clock cycles, not the refer-

ence clock cycles given by the time stamp counter.

Reciprocal throughput:

The average number of core clock cycles per instruction for a series of independent in-

structions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instruc- tions							
MOV	r,i	1	1	p0156		0.25	
MOV	r8/16,r8/16	1	1	p0156	1	0.25	
MOV	r32/64,r32/64	1	1	p0156	0-1	0.25	may be elim.
MOV	r8l,m	1	2	p23 p0156		0.5	
MOV	r8h,m	1	1	p23		0.5	
MOV	r16,m	1	2	p23 p0156		0.5	
MOV	r32/64,m	1	1	p23	2	0.5	all addressing modes
MOV	m,r	1	2	p237 p4	3	1	
MOV	m,i	1	2	p237 p4		1	
MOVNTI	m,r	2	2	p23 p4	~400	1	

MOVSX MOVZX	r,r	1	1	p0156	1	0.25	
MOVSXD							
MOVSX MOVZX	r16,m8	1	2	p23 p0156		0.5	
MOVSX MOVZX MOVSXD	r,m	1	1	p23		0.5	all other combinations
CMOVcc	r,r	1	1	p06	1	0.5	
CMOVcc	r,m	2	2	p06 p23		0.5	
XCHG	r,r	3	3	3p0156	2	1	
XCHG	r,m	8	8		21		implicit lock
XLAT		3	3	p23 2p0156	7	2	
PUSH	r	1	2	p237 p4	3	1	
PUSH	i	1	2	p237 p4		1	
PUSH	m	2	3	p4 2p237		1	
PUSH	stack pointer	2	3	p0156 p237 p4		1	
PUSHF(D/Q)	Stack pointer	3	4	p1 p4 p237 p06		1	
PUSHA(D)		11	19	p 1 p 4 p207 p00		8	not 64 bit
POP	r	1	1	p23	2	0.5	HOL O4 DIL
POP	stack pointer	3	3	p23 2p0156	2	4	
POP		2	3	2p237 p4		1	
	m		9	2p237 p4			
POPF(D/Q)		9				18	
POPA(D)		18	18		4	8	not 64 bit
LAHF SAHF		1	1	p06	1	1	
SALC		3	3	3p0156	1	1	not 64 bit
LEA	r16,m	2	2	p1 p05	2-4	1	16 or 32 bit address size
LEA	r32/64,m	1	1	p15	1	0.5	1 or 2 compo- nents in address
LEA	r32/64,m	1	1	p1	3	1	3 components in address
LEA	r32/64,m	1	1	p1		1	rip relative address
BSWAP	r32	1	1	p15	1	0.5	addicoo
BSWAP	r64	2	2	p06 p15	2	1	
MOVBE	r16,m16	3	3	2p0156 p23	_	0.5-1	MOVBE
MOVBE	r32,m32	2	2	p15 p23		0.5	MOVBE
MOVBE	r64,m64	3	3	2p0156 p23		0.5	MOVBE
MOVBE	m16,r16	2	3	p06 p237 p4		1	MOVBE
MOVBE	m32,r32	2	3	p15 p237 p4			MOVBE
MOVBE	m64,r64	3	4	p06 p15 p237 p4			MOVBE
MOVBE	11104,104	3	4	poo p 13 p237 p4		'	MOVBE
PREFETCHNTA/ 0/1/2	m	1	1	p23		0.5	
PREFETCHW	m	1	1	p23		1	PREFETCHW
LFENCE		2	•	none counted		4	
MFENCE		3	3	p23 p4		33	
SFENCE		2	2	p23 p4		6	
Arithmetic in- structions							
ADD SUB	r,r/i	1	1	p0156	1	0.25	
ADD SUB	r,m	1	2	p0156 p23		0.5	
ADD SUB	m,r/i	2	4	2p0156 2p237 p4	6	1	
ADC SBB	r,r/i	1	1	p06	1	1	

ADC SBB	l	۱ ۵	۱ ۵	n06 n22	l	1	
ADC SBB	r,m	2 4	2 6	p06 p23 3p0156 2p237 p4	7	1 2	
	m,r/i	· ·			7		
CMP	r,r/i	1	1	p0156	1	0.25	
CMP	m,r/i	1	2	p0156 p23	1	0.5	
INC DEC NEG NOT	r	1	1	p0156	1	0.25	
INC DEC NOT	m	3	4	p0156 2p237 p4	6	1	
NEG	m	2	4	p0156 2p237 p4	6	1	
AAA		2	2	p1 p56	4		not 64 bit
AAS		2	2	p1 p056	6		not 64 bit
DAA DAS		3	3	p1 2p056	4		not 64 bit
AAD		3	3	p1 2p056	6		not 64 bit
AAM		8	8	p0 p1 p5 p6	21	7	not 64 bit
MUL IMUL	r8	1	1	p1	3	1	
MUL IMUL	r16	4	4	p1 p0156	4	2	
MUL IMUL	r32	3	3	p1 p0156	4	2	
MUL IMUL	r64	2	2	p1 p6	3	1	
MUL IMUL	m8	1	2	p1 p23		1	
MUL IMUL	m16	4	5	p1 3p0156 p23		2	
MUL IMUL	m32	3	4	p1 2p0156 p23		2	
MUL IMUL	m64	2	3	p1 p6 p23		1	
IMUL	r,r	1	1	p1 p0 p23	3	1	
IMUL	r,m	1	2	p1 p23	3	1	
IMUL	r16,r16,i	2	2		4	1	
IMUL		1	1	p1 p0156	4 3	1	
IMUL	r32,r32,i			p1	3	=	
	r64,r64,i	2	1 3	p1	<u>ى</u>	1	
IMUL	r16,m16,i			p1 p0156 p23		1	
IMUL	r32,m32,i	1	2	p1 p23		1	
IMUL	r64,m64,i	1	2	p1 p23		1	DMIO
MULX	r32,r32,r32	3	3	p1 2p056	4	1	BMI2
MULX	r32,r32,m32	3	4	p1 2p056 p23	_	1	BMI2
MULX	r64,r64,r64	2	2	p1 p5	4	1	BMI2
MULX	r64,r64,m64	2	3	p1 p6 p23	00.05	1	BMI2
DIV	r8	9	9	p0 p1 p5 p6	22-25	9	
DIV	r16	11	11	p0 p1 p5 p6	23-26	9	
DIV	r32	10	10	p0 p1 p5 p6	22-29	9	
DIV	r64	36	36	p0 p1 p5 p6	32-95	21-73	
IDIV	r8	9	9	p0 p1 p5 p6	23-26	6	
IDIV	r16	10	10	p0 p1 p5 p6	23-26	6	
IDIV	r32	9	9	p0 p1 p5 p6	22-29	6	
IDIV	r64	59	59	p0 p1 p5 p6	39-103	24-81	
CBW		1	1	p0156	1		
CWDE		1	1	p0156	1		
CDQE		1	1	p0156	1		
CWD		2	2	p0156	1		
CDQ		1	1	p06	1		
CQO		1	1	p06	1		
POPCNT	r,r	1	1	p1	3	1	SSE4.2
POPCNT	r,m	1	2	p1 p23		1	SSE4.2
CRC32	r,r	1	1	p1	3	1	SSE4.2
CRC32	r,m	1	2	p1 p23		1	SSE4.2

Logic instruc-							
AND OR XOR	r,r/i	1	1	p0156	1	0.25	
AND OR XOR	r,m	1	2	p0156 p23		0.5	
AND OR XOR	m,r/i	2	4	2p0156 2p237 p4	6	1	
TEST	r,r/i	1	1	p0156	1	0.25	
TEST	m,r/i	1	2	p0156 p23	1	0.5	
SHR SHL SAR	r,i	1	1	p06	1	0.5	
SHR SHL SAR	m,i	3	4	2p06 p237 p4	•	2	
SHR SHL SAR	r,cl	3	3	3p06	2	2	
SHR SHL SAR	m,cl	5	6	3p06 2p23 p4	_	4	
ROR ROL	r,1	2	2	2p06	1	1	short form
ROR ROL	r,i	1	1	p06	1	0.5	
ROR ROL	m,i	4	5	2p06 2p237 p4		2	
ROR ROL	r,cl	3	3	3p06	2	2	
ROR ROL	m,cl	5	6	3p06 p23 p4	_	4	
RCR RCL	r,1	3	3	2p06 p0156	2	2	
RCR RCL	m,1	4	6		_	3	
RCR RCL	r,i	8	8	p0156	6	6	
RCR RCL	m,i	11	11	p 0.00		6	
RCR RCL	r,cl	8	8	p0156	6	6	
RCR RCL	m,cl	11	11	p 0.00		6	
SHRD SHLD	r,r,i	1	1	p1	3	1	
SHRD SHLD	m,r,i	3	5	F .		2	
SHLD	r,r,cl	4	4	p0156	3	2	
SHRD	r,r,cl	4	4	p0156	4	2	
SHRD SHLD	m,r,cl	5	7	p 0.00	•	4	
SHLX SHRX SARX	r,r,r	1	1	p06	1	0.5	BMI2
SHLX SHRX SARX	r,m,r	2	2	p06 p23		0.5	BMI2
RORX	r,r,i	1	1	p06	1	0.5	BMI2
RORX	r,m,i	2	2	p06 p23		0.5	BMI2
BT	r,r/i	1	1	p06	1	0.5	
BT	m,r	10	10	Poo		5	
BT	m,i	2	2	p06 p23		0.5	
BTR BTS BTC	r,r/i	1	1	p06	1	0.5	
BTR BTS BTC	m,r	10	10		-	5	
BTR BTS BTC	m,i	2	2	p06 p23		0.5	
BSF BSR	r,r	1	1	p1	3	1	
BSF BSR	r,m	1	2	p1 p23	_	1	
SETcc	r	1	1	p06	1	0.5	
SETcc	m	2	3	p06 p237 p4		1	
CLC		1	0	none		0.25	
STC		1	1	p0156		0.25	
CMC		1	1	p0156	1	1	
CLD STD		3	3	p15 p6	-	4	
LZCNT	r,r	1	1	p1	3	1	LZCNT
LZCNT	r,m	1	2	p1 p23		1	LZCNT
TZCNT	r,r	1	1	p1	3	1	BMI1
TZCNT	r,m	1	2	p1 p23		1	BMI1
ANDN	r,r,r	1	1	p15	1	0.5	BMI1
ANDN	r,r,m	1	2	p15 p23	1	0.5	BMI1
BLSI BLSMSK	r,r	1	1	p15	1	0.5	BMI1
BLSR	,						

BEXTR	BLSI BLSMSK BLSR	r,m	1	2	p15 p23		0.5	BMI1
BEXTR BZHI r,m,r BZHI r,r,r BZHI r,r,r BZHI r,m,r BZHI r,m,r 1 1 2 p15 p23 PDEP r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,r 1 1 1 p6 2 1-2 JMP short/near r 1 1 p6 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2		rrr	2	2	2n0156	2	0.5	BMI1
BZHI r,r,r 1 1 1 p15 1 0.5 BMI2 BZHI r,m,r 1 1 2 p15 p23 0.5 BMI2 PDEP r,r,r 1 1 1 p1 p3 3 1 BMI2 PDEP r,r,r 1 1 1 p1 3 1 BMI2 PEXT r,r,m 1 2 p1 p23 1 BMI2 PEXT r,r,m 1 2 p1 p23 1 BMI2 PEXT r,r,m 1 2 p1 p23 1 BMI2 PEXT r,r,m 1 2 p1 p23 1 BMI2 PEXT r,r,m 1 2 p1 p23 1 BMI2 PEXT r,r,m 1 2 p1 p23 1 PMI2 PEXT r,r,m 1 2 p1 p23 1 PMI2 PEXT r,r,m 1 2 p1 p23 1 PMI2 PEXT r,r,m 1 2 p1 p23 1 PMI2 PEXT r,r,m 1 2 p1 p23 1 PMI2 PEXT r,r,m 1 2 p6 1-2 JMP r 1 1 p6 2 2 JMP m 1 2 p23 p6 2 Conditional jump short/near 1 1 p6 1-2 JMP m 1 2 p23 p6 2 Conditional jump short/near 1 1 p6 1-2 Fused arithmetic and branch Fused arithmetic and branch Fused arithmetic and branch Fused arithmetic and branch JGE/R/CXZ short 2 2 p0156 p6 0.5-1 LOOP(N)E short 11 11					1 '	2		
BZHI						1		
PDEP			1 1		1 '	1		
PDEP					1 ' '	0		
PEXT r,r,r 1 1 2 p1 p1 p23 3 1 BMI2 BMI2 BMI2 Control transfer instructions JMP short/near 1 1 p6 1-2 JMP JMP r 1 1 p6 2 JMP JMP m 1 2 p23 p6 2 2 predicted faken predicted fa					1 '	3		
PEXT r,r,m 1 2 p1 p23 1 BMI2 Control transfer instructions JMP short/near 1 1 p6 1-2 JMP short/near 1 1 p6 2 JMP m 1 2 p23 p6 2 Conditional jump short/near 1 1 p6 1-2 predicted taken predicted taken predicted not taken predicted and branch Fused arithmetic and branch 1 1 p6 1-2 predicted not taken predicted not taken predicted not taken predicted and branch Lucor Fused arithmetic and branch 3 1 1 p6 0.5-1 predicted not taken predicted not taken Fused arithmetic and branch 3 4 p06 0.5-2 0.5-2 0.5-1 predicted not taken Fused arithmetic and branch 4 1 1 p6 0.5-1 predicted not taken Loop (N)E short 2 2 p0156 p6 0.5-1 0.5-1 po16cted not taken LOOP (N)E <td></td> <td></td> <td></td> <td></td> <td>1 1</td> <td></td> <td></td> <td></td>					1 1			
Control transfer instructions JMP short/near 1 1 p6 1-2 JMP JMP r 1 1 p6 2 2 JMP JMP m 1 2 p23 p6 2 2 2 JMP JMP JMP JMP JMP JMP JMP JMP MMP MMP JMP		r,r,r			1	3		
JMP JMP JMP MP r short/near r 1 1 1 p6 p6 p6 p6 p6 p6 p6 p6	PEXT	r,r,m	1	2	p1 p23		1	BMI2
JMP JMP r m 1 m 1 m 2 p23 p6 2 p23 p6 2 predicted taken Conditional jump short/near 1 m 2 m 3 m 3 m 3 m 3 m 3 m 3 m 3 m 3 m 3 m 3	Control transfer i	nstructions						
JMP JMP r m 1 m 1 m 2 p23 p6 2 p23 p6 2 predicted taken Conditional jump short/near 1 m 2 m 3 m 3 m 3 m 3 m 3 m 3 m 3 m 3 m 3 m 3			1 1	1	8 q		1-2	
JMP Conditional jump m 1 2 p23 p6 2 predicted taken Conditional jump short/near 1 1 p6 1-2 predicted taken Fused arithmetic and branch 1 1 p6 1-2 predicted not taken Fused arithmetic and branch 1 1 p6 1-2 predicted taken Fused arithmetic and branch 1 1 p6 0.5-1 predicted taken Fused arithmetic and branch 1 1 p6 0.5-1 predicted taken Fused arithmetic and branch 1 1 p6 0.5-1 predicted taken Fused arithmetic and branch 1 1 p6 0.5-1 predicted taken Fused arithmetic and branch 1 1 p6 0.5-1 predicted taken J(E/R)CXZ short 2 2 p0156 p6 0.5-1 predicted taken J(E/R)CXZ short 1 1 1 0.5 0.5 0.5-2 0.5-2 0.5-2 <td></td> <td></td> <td></td> <td></td> <td>1</td> <td></td> <td></td> <td></td>					1			
Conditional jump short/near 1 1 p6 1-2 predicted taken predicted not taken predicted not taken Fused arithmetic and branch 1 1 p6 1-2 predicted not taken predicted not taken predicted not taken Fused arithmetic and branch 1 1 p6 0.5-1 predicted not taken Fused arithmetic and branch 1 1 p06 0.5-1 predicted not taken J(E/R)CXZ short 2 2 p0156 p6 0.5-2 0.5-2 LOOP (N)E short 7 7 5 0.5-2 <t< td=""><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td></t<>					1			
Taken Tak					1 1			predicted
Tused arithmetic and branch Fused arithmetic arithmetic and branch Fused arithmetic Fused arithmetic and branch Fused arithmetic and branch Fus								taken
and branch Fused arithmetic and branch Fused arithmetic and branch J(E/R)CXZ	Conditional jump	short/near	1	1	p06		0.5-1	
Fused arithmetic and branch 1	Fused arithmetic and branch		1	1	p6		1-2	
and branch J(E/R)CXZ short Z Z D D D D D Short Z S D D D D D D D D D D D D	Fused arithmetic		1 1	1	90g		0.5-1	
LOOP LOOP(N)E short short 7 7 7 5 6 7 7 7 7 7 7 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 3 3 3 2 <td>and branch</td> <td></td> <td></td> <td></td> <td>'</td> <td></td> <td></td> <td></td>	and branch				'			
LOOP LOOP(N)E short short 7 7 7 5 6 7 7 7 7 7 7 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 3 3 3 2 <td>J(E/R)CXZ</td> <td>short</td> <td>2</td> <td>2</td> <td>p0156 p6</td> <td></td> <td>0.5-2</td> <td></td>	J(E/R)CXZ	short	2	2	p0156 p6		0.5-2	
LOOP(N)E short near 11 11 11 near 2 3 p237 p4 p6 2 1 3 3 2 2 2 1 3 3 2 2 1 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2	1 1		1					
CALL near 2 3 p237 p4 p6 2 CALL r 2 3 p237 p4 p6 2 CALL m 3 4 2p237 p4 p6 3 RET 1 2 p237 p6 1 RET i 3 4 p23 2p6 p015 2 BOUND r,m 15 15 8 not 64 bit INTO 4 4 4 5 not 64 bit String instructions 3 3 2p0156 p23 1 1 LODSB/W 3 3 2p0156 p23 1 2n LODSB/W 2 2 p0156 p23 1 2n STOS 3 3 p23 p0156 p23 1 2n REP STOS 3 3 p23 p0156 p23 1 2n REP STOS 2 2n 2n 2n 2n MOVS 5 5 2p23 p0156 4 1n								
CALL CALL r 2 3 p237 p4 p6 2 CALL RET m 3 4 2p237 p4 p6 3 RET RET i 3 4 p23 2p6 p015 2 RET BOUND INTO r,m 15 15 15 8 not 64 bit INTO 4 4 4 5 not 64 bit 5 not 64 bit 1 2 p23 2p6 p015 2 8 not 64 bit 1 not 64 bit 1 1 2 p23 p0156 p23 1 1 1 2 p20 p0156 p23 1 1 2 2 p20 p0156 p23 1 2	` '				n237 n4 n6			
CALL m 3 4 2p237 p4 p6 p237 p6 p237 p6 3 RET i 3 4 p23 2p6 p015 2 BOUND r,m 15 15 8 not 64 bit INTO 4 4 4 5 not 64 bit String instructions LODSB/W 3 3 2p0156 p23 1 LODSD/Q 2 2 p0156 p23 1 REP LODS 5n+12 3 3 p23 p0156 p4 1 REP STOS 3 3 p23 p0156 p4 1 ~2n REP STOS 2n 2n 1/32B worst case best case aligned by 32 MOVS 5 5 2p23 p4 2p0156 4 <1n			1		1			
RET 1			1					
RET i 3 4 p23 2p6 p015 2 not 64 bit BOUND r,m 15 15 15 15 8 not 64 bit String instructions LODSB/W 3 3 2p0156 p23 1 1 LODSB/W 1 2p0156 p23 2p0156 p23 1 2p0156 p23		m						
BOUND INTO r,m 15 4 4 4 8 5 not 64 bit not 64 bit not 64 bit not 64 bit String instructions LODSB/W LODSD/Q REP LODS STOS 3 3 2p0156 p23 1 2 2 p0156 p23 1 1 2 2 p0156 p23 1 1 2 p0156 p23 1 1 2 p0156 p23 1 1 2 p0156 p23 p0156 p4 1 2 p0156 p4 1 p0156 p23 p0156 p4 1 p0156 p4 p0156 p1 p015					1 1			
String instructions LODSB/W 3 3 2p0156 p23 1 LODSD/Q 2 2 p0156 p23 1 REP LODS 5n+12 ~2n ~2n STOS 3 3 p23 p0156 p4 1 REP STOS <2n	RET	i	3	4	p23 2p6 p015			
String instructions LODSB/W 3 3 2p0156 p23 1 LODSD/Q 2 2 p0156 p23 1 REP LODS 5n+12 2 p0156 p23 1 STOS 3 3 p23 p0156 p4 1 REP STOS <2n	BOUND	r,m	15	15			8	not 64 bit
tions 3 3 2p0156 p23 1 LODSD/Q 2 2 p0156 p23 1 REP LODS 5n+12 2 p0156 p23 1 STOS 3 3 p23 p0156 p4 1 REP STOS <2n	INTO		4	4			5	not 64 bit
LODSB/W 3 3 2p0156 p23 1 LODSD/Q 2 2 p0156 p23 1 REP LODS 5n+12 ~2n ~2n STOS 3 3 p23 p0156 p4 1 REP STOS <2n	String instruc-							
LODSD/Q 2 2 p0156 p23 1 REP LODS 5n+12 3 3 p23 p0156 p4 1 STOS 3 3 p23 p0156 p4 1 REP STOS <2n			2	3	2n0156 n23		1	
REP LODS 5n+12 3 3 p23 p0156 p4 1 ~2n 1 REP STOS <2n								
STOS 3 3 p23 p0156 p4 1 ~0.5n worst case best case aligned by 32 MOVS 5 5 2p23 p4 2p0156 4 <1n					pu 156 p25		•	
REP STOS <2n			1					
REP STOS 2.6/32B 1/32B best case aligned by 32 MOVS 5 5 2p23 p4 2p0156 4 REP MOVS -2n 4/32B < 1n				3	p23 p0156 p4			
MOVS 5 5 2p23 p4 2p0156 4 < 1n								
REP MOVS ~2n 4/32B < 1n	REP STOS		2.6/32B				1/32B	
REP MOVS ~2n 4/32B < 1n	MOVS		5	5	2p23 p4 2p0156		4	_
REP MOVS 4/32B 1/32B best case aligned by 32 SCAS 3 3 p23 2p0156 1 REP SCAS ≥6n ≥2n ≥2n CMPS 5 5 2p23 3p0156 4 REP CMPS ≥8n ≥2n	REP MOVS		~2n		' '		< 1n	worst case
SCAS 3 3 p23 2p0156 1 REP SCAS ≥6n 5 5 2p23 3p0156 4 CMPS ≥8n ≥2n ≥2n Synchronization instructions								
SCAS 3 3 p23 2p0156 1 REP SCAS ≥6n 5 2p23 3p0156 4 CMPS ≥8n ≥2n Synchronization instructions			., 525				.,025	
REP SCAS ≥6n 5 5 2p23 3p0156 ≥2n CMPS 5 ≥8n ≥2n 4 ≥2n Synchronization instructions	SCAS		3	3	p23 2p0156		1	
CMPS 5 5 2p23 3p0156 4 REP CMPS ≥8n ≥2n Synchronization instructions								
REP CMPS ≥8n ≥2n Synchronization instructions			1	5	2n23 3n0156			
Synchronization instructions				3	Zp23 3p0 130			
	NEF CIVIPS		∠ 0(1					
	Synchronization	instructions						
	XADD		4	5			6	

LOCK XADD LOCK ADD CMPXCHG LOCK CMPXCHG CMPXCHG8B LOCK CMPXCHG8B CMPXCHG16B	m,r m,r m,r m,r m,r	9 8 5 10 15 19 22	9 8 6 10 15 19 22			21 21 7 21 8 21 15	
LOCK CMPXCHG16B	m,r m,r	24	24			27	
	,.						
Other							
NOP (90)		1	0	none		0.25	
Long NOP (0F 1F)		1	0	none		0.25	
PAUSE		5	5	p05 3p6		9	
ENTER	a,0	12	12			8	
ENTER	a,b	~14+7b	~45+7b		~87+2b		
LEAVE		3	3	2p0156 p23		5	
XGETBV		8	8			5	XGETBV
RDTSC		15	15			24	
RDTSCP		21	21			30	RDTSCP
RDPMC		34	34			37	
RDRAND	r	16	16	p23 15p0156		~230	RDRAND
RDSEED	r	16	16	p23 15p0156		~230	RDSEED

Floating point x87 instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc-							
tions							
FLD	r	1	1	p01	1	0.5	
FLD	m32/64	1	1	p23	3	0.5	
FLD	m80	4	4	2p01 2p23	4	2	
FBLD	m80	43	43		47	22	
FST(P)	r	1	1	p01	1	0.5	
FST(P)	m32/m64	1	2	p4 p237	4	1	
FSTP	m80	7	7	3p0156 2p23 2p4	5	5	
FBSTP	m80	238	226		269	267	
FXCH	r	2	0	none	0	0.5	
FILD	m	1	2	p01 p23	6	1	
FIST(P)	m	3	3	p1 p23 p4	7	1	
FISTTP	m	3	3	p1 p23 p4	7	2	SSE3
FLDZ		1	1	p01		1	
FLD1		2	2	2p01		2	
FLDPI FLDL2E et	tc.	2	2	2p01		2	
FCMOVcc	r	3	3	2p0 p5	2	2	
FNSTSW	AX	2	2	p0 p0156	6	1	
FNSTSW	m16	2	3	p0 p4 p237	6	1	
FLDCW	m16	3	3	p01 p23 p6	7	2	
FNSTCW	m16	2	3	p237 p4 p6	6	1	
FINCSTP FDECS	TP	1	1	p01	0	0.5	
FFREE(P)	r	1	1	p01		0.5	

FNSAVE	m	152	152		173	173	
FRSTOR	m	95	95		175	175	
Arithmetic in-							
structions							
FADD(P)				_	_	_	
FSUB(R)(P)	r	1	1	p1	3	1	
FADD(P)	m	1	2	p1 p23		1	
FSUB(R)(P)	m	1	1	· ·	5	1	
FMUL(P) FMUL(P)	r	1	2	p0	5	1	
1 ' '	m	1	1	p0 p23	10-15	4-5	
FDIV(R)(P)	r	1	2	p0	10-15	4-5 4-5	
FDIV(R)(P) FABS	m		1	p0 p23	4		
FCHS		1 1	1	p0	1 1	1 1	
	_			p0	3		
FCOM(P) FUCOM	r	1	1 2	p1	3	1	
FCOMPRISION	m IDD	1 2	2	p1 p23		1 1	
FCOMPP FUCOM	IPP			2p01		I	
FCOMI(P) FUCOMI(P)	r	3	3	3p01	7	1.5	
FIADD FISUB(R)	m	2	3	2p1 p23	,	2	
FIMUL	m	2	3	p0 p1 p23		2	
FIDIV(R)	m	2	3	p0 p1 p23		2	
FICOM(P)	m	2	3	2p1 p23		2	
FTST	111	1	1	p1	3	1	
FXAM		2	2	2p1	6	2	
FPREM		28	28	Ζρι	20-24	13	
FPREM1		28	28		23-48	13	
FRNDINT		17	17		11	23	
ITANDINI		17	17		11	23	
Math							
FSCALE		27	27		125	130	
FXTRACT		17	17		123	11	
FSQRT		1 1	1	p0	10-23	4-9	
FSIN		75-100	'	Po	48-106	4-3	
FPATAN		27-71			97-147		
Other							
		1	1	n01		0.5	
						-	
				P0100			
FCOS FSINCOS F2XM1 FYL2X FYL2XP1 FPTAN FPATAN Other FNOP WAIT FNCLEX FNINIT		70-100 70-110 16-86 55-96 56 71-102 27-71 1 2 5 26	1 2 5 26	p01 p01 p0156	49-112 52-124 63-68 92 74 132 97-147	0.5 1 22 84	

Integer vector instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μορs each port	Latonov	Recipro- cal through put	Comments
Move instruc-	Operanus	uomam	uomam	pops each port	Latericy	put	Comments
tions							
MOVD	r32/64,mm/x	1	1	p0	1	1	
MOVD	m32/64,mm/x	1	2	p237 p4	3	1	
MOVD	mm/x,r32/64	1	1	p237 p4	1	1	
MOVD	1	1 1	1		3	0.5	
MOVQ	mm/x,m32/64	1	1	p23			
	r64,mm/x	1	1	p0	1	1	
MOVQ	mm/x,r64		1	p5	1	1	
MOVQ	mm/x,mm/x	1		p015	1	0.33	
MOVQ	mm/x,m64	1	1	p23	3	0.5	
MOVQ	m64, mm/x	1	2	p237 p4	3	1	
MOVDQA/U	X,X	1	1	p015	0-1	0.25	may be elim.
MOVDQA/U	x, m128	1	1	p23	3	0.5	
MOVDQA/U	m128, x	1	2	p237 p4	3	1	
							AVX
VMOVDQA/U	y,y	1	1	p015	0-1	0.25	may be elim.
VMOVDQA/U	y,m256	1	1	p23	3	0.5	AVX
VMOVDQA/U	m256,y	1	2	p237 p4	4	1	AVX
LDDQU	x, m128	1	1	p23	3	0.5	SSE3
MOVDQ2Q	mm, x	2	2	p01 p5	1	1	
MOVQ2DQ	x,mm	1	1	p015	1	0.33	
MOVNTQ	m64,mm	1	2	p237 p4	~400	1	
MOVNTDQ	m128,x	1	2	p237 p4	~400	1	
VMOVNTDQ	m256,y	1	2	p237 p4	~400	1	AVX2
MOVNTDQA	x, m128	1	1	p23	3	0.5	SSE4.1
VMOVNTDQA	y,m256	1	1	p23	3	0.5	AVX2
PACKSSWB/DW	y,200		•	P20		0.0	7,7,7,2
PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW	,				_		
PACKUSWB	mm,m64	3	3	p23 2p5		2	
PACKSSWB/DW	, -						
PACKUSWB	x,x / y,y,y	1	1	p5	1	1	
PACKSSWB/DW							
PACKUSWB	x,m / y,y,m	1	2	p23 p5		1	
PACKUSDW	x,x / y,y,y	1	1	p5	1	1	SSE4.1
PACKUSDW	x,m / y,y,m	1	2	p23 p5		1	SSE4.1
PUNPCKH/L	, ,,,,,			' '			
BW/WD/DQ	v,v / v,v,v	1	1	p5	1	1	
PUNPCKH/L							
BW/WD/DQ	v,m / v,v,m	1	2	p23 p5		1	
PUNPCKH/L							
QDQ	x,x / y,y,y	1	1	p5	1	1	
PUNPCKH/L							
QDQ	x,m / y,y,m	2	2	p23 p5		1	
PMOVSX/ZX BW				_			
BD BQ DW DQ	X,X	1	1	p5	1	1	SSE4.1
PMOVSX/ZX BW		4		"OO " T		4	00544
BD BQ DW DQ	x,m	1	2	p23 p5		1	SSE4.1
VPMOVSX/ZX BW BD BQ DW DQ	VV	1	1	p5	3	1	AVX2
VPMOVSX/ZX BW	y,x	'		μο		'	7772
BD BQ DW DQ	y,m	2	2	p5 p23		1	AVX2

1		ı		1		1	1
PSHUFB	v,v / v,v,v	1	1	p5	1	1	SSSE3
PSHUFB	v,m / v,v,m	2	2	p23 p5		1	SSSE3
PSHUFW	mm,mm,i	1	1	p5	1	1	
PSHUFW	mm,m64,i	2	2	p23 p5		1	
PSHUFD	v,v,i	1	1	p5	1	1	
PSHUFD	v,m,i	2	2	p23 p5		1	
PSHUFL/HW	v,v,i	1	1	p5	1	1	
PSHUFL/HW	v,m,i	2	2	p23 p5		1	
PALIGNR	v,v,i / v,v,v,i	1	1	p5	1	1	SSSE3
PALIGNR	v,m,i / v,v,m,i	2	2	p23 p5		1	SSSE3
PBLENDVB	x,x,xmm0	2	2	2p5	2	2	SSE4.1
PBLENDVB	x,m,xmm0	3	3	2p5 p23		2	SSE4.1
VPBLENDVB	V,V,V,V	2	2	2p5	2	2	AVX2
VPBLENDVB	v,v,m,v	3	3	2p5 p23		2	AVX2
PBLENDW	x,x,i / v,v,v,i	1	1	p5	1	1	SSE4.1
PBLENDW	x,m,i / v,v,m,i	2	2	p23 p5		1	SSE4.1
VPBLENDD	v,v,v,i	1	1	p015	1	0.33	AVX2
VPBLENDD	v,v,m,i	2	2	p015 p23		0.5	AVX2
VPERMD	y,y,y	_ 1	1	p5	3	1	AVX2
VPERMD	y,y,m	1	2	p5 p23		1	AVX2
VPERMQ	y,y,iii y,y,i	1	1	p5 p20	3	1	AVX2
VPERMQ	y,y,i y,m,i	2	2	p5 p23			AVX2
VPERM2I128	y,,,,,i y,y,y,i	1	1	p5 p25	3		AVX2
VPERM2I128		2	2	p5 p23	3		AVX2 AVX2
MASKMOVQ	y,y,m,i	4	4	p0 p4 2p23	18-500		AVAZ
MASKMOVDQU	mm,mm	10	10	4p04 2p56 4p23	18-500	6	
	X,X	3	3			2	AVX2
VPMASKMOVD/Q	v,v,m			p23 2p5	4	1	
VPMASKMOVD/Q	m,v,v	4	4	p0 p1 p4 p23	15		AVX2
PMOVMSKB	r,v	1	1	p0	3	1	00544
PEXTRB/W/D/Q	r32,x,i	2	2	p0 p5	2	1	SSE4.1
PEXTRB/W/D/Q	m8,x,i	2	3	p23 p4 p5		1	SSE4.1
VEXTRACTI128	x,y,i	1	1	p5	3	1	AVX2
VEXTRACTI128	m,y,i	2	2	p23 p4	4	1	AVX2
PINSRB	x,r32,i	2	2	p5	2	2	SSE4.1
PINSRB	x,m8,i	2	2	p23 p5		1	SSE4.1
PINSRW	mm/x,r32,i	2	2	p5	2	2	
PINSRW	mm/x,m16,i	2	2	p23 p5	_	1	
PINSRD/Q	x,r32,i	2	2	p5	2	2	SSE4.1
PINSRD/Q	x,m32,i	2	2	p23 p5		1	SSE4.1
VINSERTI128	y,y,x,i	1	1	p5	3	1	AVX2
VINSERTI128	y,y,m,i	2	2	p015 p23	4	0.5	AVX2
VPBROADCAST		4	_		_	_	A) ()/O
B/W/D/Q	x,x	1	1	p5	1	1	AVX2
VPBROADCAST B/W	x,m8/16	3	3	p01 p23 p5	5	1	AVX2
VPBROADCAST	λ,1110/10	3		po 1 p23 p3	J		AVAZ
D/Q	x,m32/64	1	1	p23	4	0.5	AVX2
VPBROADCAST	,						
B/W/D/Q	y,x	1	1	p5	3	1	AVX2
VPBROADCAST							
B/W	y,m8/16	3	3	p01 p23 p5	7	1	AVX2
VPBROADCAST	V == 20/04	4		m00	_	0.5	A) ()/O
D/Q	y,m32/64	1	1	p23	5	0.5	AVX2
VBROADCASTI128	y,m128	1	1	p23	3	0.5	AVX2
VPGATHERDD	x,[r+s*x],x	10	10			6	AVX2

VPGATHERDD	y,[r+s*y],y	14	14			7	AVX2
VPGATHERQD	x,[r+s*x],x	9	9			6	AVX2
VPGATHERQD	x,[r+s*y],x	10	10			6	AVX2
VPGATHERDQ	x,[r+s*x],x	7	7			5	AVX2
VPGATHERDQ	y,[r+s*x],y	9	9			6	AVX2
VPGATHERQQ	x,[r+s*x],x	7	7			5	AVX2 AVX2
VPGATHERQQ	y,[r+s*y],y	9	9			6	AVX2
Arithmetic in-							
structions							
PADD/SUB(S,US)		4	4	m 1 F	_	0.5	
B/W/D/Q	v,v / v,v,v	1	1	p15	1	0.5	
PADD/SUB(S,US) B/W/D/Q	v,m / v,v,m	1	2	p15 p23		0.5	
PHADD(S)W/D	V,111 / V,V,111			p 13 p23		0.5	
PHSUB(S)W/D	v,v / v,v,v	3	3	p1 2p5	3	2	SSSE3
` '	V,V / V,V,V	J	3	p i 2p3	3		333E3
PHADD(S)W/D PHSUB(S)W/D	y m / y y m	4	4	n1 2n5 n22		2	SSSE3
` '	v,m / v,v,m	4	4	p1 2p5 p23			333E3
PCMPEQB/W/D PCMPGTB/W/D	N. V. I. V. V. V.	1	1	p15	1	0.5	
	v,v / v,v,v	I	I	pis	'	0.5	
PCMPEQB/W/D PCMPGTB/W/D	\m_ /\\\m_	1	2	n15 n02		0.5	
	v,m / v,v,m			p15 p23	_		00544
PCMPEQQ	v,v / v,v,v	1	1	p15	1	0.5	SSE4.1
PCMPEQQ	v,m / v,v,m	1	2	p15 p23	_	0.5	SSE4.1
PCMPGTQ	v,v / v,v,v	1	1	p0	5	1	SSE4.2
PCMPGTQ	v,m / v,v,m	1	2	p0 p23		1	SSE4.2
PMULL/HW							
PMULHUW	v,v / v,v,v	1	1	p0	5	1	
PMULL/HW							
PMULHUW	v,m / v,v,m	1	2	p0 p23		1	
PMULHRSW	v,v / v,v,v	1	1	p0	5	1	SSSE3
PMULHRSW	v,m / v,v,m	1	2	p0 p23		1	SSSE3
PMULLD	x,x / y,y,y	2	2	2p0	10	2	SSE4.1
PMULLD	x,m / y,y,m	3	3	2p0 p23		2	SSE4.1
PMULDQ	x,x / y,y,y	1	1	p0	5	1 1	SSE4.1
PMULDQ	x,m / y,y,m	1	2	p0 p23		1	SSE4.1
PMULUDQ	v,v / v,v,v	1	1	p0	5	1	
PMULUDQ	v,m / v,v,m	1	2	p0 p23		1	
PMADDWD	v,v / v,v,v	1	1	p0 p20	5	1	
PMADDWD	v,w / v,v,w v,m / v,v,m	1	2	p0 p23		1	
PMADDUBSW		1	1		5		SSSE3
_	v,v / v,v,v			p0	5	1 1	
PMADDUBSW	v,m / v,v,m	1	2	p0 p23		1	SSSE3
PAVGB/W	v,v / v,v,v	1	1	p15	1	0.5	
PAVGB/W	v,m / v,v,m	1	2	p15 p23		0.5	
PMIN/PMAX							
SB/SW/SD	,	_	_				
UB/UW/UD	x,x / y,y,y	1	1	p15	1	0.5	SSE4.1
PMIN/PMAX							
SB/SW/SD				45.00			00=4.4
UB/UW/UD	x,m / y,y,m	1	2	p15 p23	_	0.5	SSE4.1
PHMINPOSUW	X,X	1	1	p0	5	1	SSE4.1
PHMINPOSUW	x,m128	1	2	p0 p23		1 1	SSE4.1
PABSB/W/D	V,V	1	1	p15	1	0.5	SSSE3
PABSB/W/D	v,m	1	2	p15 p23		0.5	SSSE3
PSIGNB/W/D	v,v / v,v,v	1	1	p15	1	0.5	SSSE3

1	1		1			ı	1
PSIGNB/W/D	v,m / v,v,m	1	2	p15 p23		0.5	SSSE3
PSADBW	v,v / v,v,v	1	1	p0	5	1	
PSADBW	v,m / v,v,m	1	2	p0 p23		1	
MPSADBW	x,x,i / v,v,v,i	3	3	p0 2p5	6	2	SSE4.1
MPSADBW	x,m,i / v,v,m,i	4	4	p0 2p5 p23		2	SSE4.1
Logic instruc-							
PAND PANDN POR PXOR	v,v / v,v,v	1	1	p015	1	0.33	
PAND PANDN	,			0.45			
POR PXOR	v,m / v,v,m	1	2	p015 p23	_	0.5	
PTEST	V,V	2	2	p0 p5	2	1	SSE4.1
PTEST	v,m	2	3	p0 p5 p23		1	SSE4.1
PSLLW/D/Q							
PSRLW/D/Q							
PSRAW/D/Q	mm,mm	1	1	p0	1	1	
PSLLW/D/Q							
PSRLW/D/Q							
PSRAW/D/Q	mm,m64	1	2	p0 p23		1	
PSLLW/D/Q							
PSRLW/D/Q							
PSRAW/D/Q	x,x / v,v,x	2	2	p0 p5	2	1	
PSLLW/D/Q							
PSRLW/D/Q							
PSRAW/D/Q	x,m / v,v,m	2	2	p0 p23		1	
PSLLW/D/Q							
PSRLW/D/Q							
PSRAW/D/Q	v,i / v,v,i	1	1	p0	1	1	
VPSLLVD/Q							
VPSRAVD							
VPSRLVD/Q	V,V,V	3	3	2p0 p5	2	2	AVX2
VPSLLVD/Q							
VPSRAVD							
VPSRLVD/Q	v,v,m	4	4	2p0 p5 p23		2	AVX2
PSLLDQ							
PSRLDQ	x,i / v,v,i	1	1	p5	1	1	
String instruc- tions							
PCMPESTRI	x,x,i	8	8	6p05 2p16	4	4	SSE4.2
PCMPESTRI	x,m128,i	8	8	3p0 2p16 2p5 p23	•	4	SSE4.2
PCMPESTRM	x,x,i	9	9	3p0 2p16 4p5	11	11	SSE4.2
PCMPESTRM	x,m128,i	9	9	6p05 2p16 p23	1 1	5	SSE4.2
PCMPISTRI		3	3	1	3	3	SSE4.2 SSE4.2
	X,X,İ			3p0	3		
PCMPISTRI	x,m128,i	4	4	3p0 p23	4.4	3	SSE4.2
PCMPISTRM	x,x,i	3	3	3p0	11	11	SSE4.2
PCMPISTRM	x,m128,i	4	4	3p0 p23		3	SSE4.2
Encryption instr	uctions						
PCLMULQDQ	x,x,i	1	1	p0	5	1	CLMUL
PCLMULQDQ	x,m,i	2	2	p0 p23		1	CLMUL
AESDEC,							
AESDECLAST,							
AESENC,							
AESENCLAST	x,x	1	1	p5	7	1	AES

AESDEC, AESDECLAST, AESENC, AESENCLAST AESIMC AESIMC AESKEYGENAS SIST AESKEYGENAS SIST	x,m x,x x,m x,x,i x,m,i	2 2 3 10	2 2 3 10	p5 p23 2p5 2p5 p23 2p0 8p5 2p0 p23 7p5	14	1.5 2 2 9	AES AES AES AES	
Other EMMS		31	31			12		

Floating point XMM and YMM instructions

rioating point		μορs fused	μορs unfused			Recipro- cal through	
Instruction	Operands	domain	domain	µops each port	Latency	put	Comments
Move instruc-							
tions	-				0.4		
MOVAPS/D	X,X	1	1	p5	0-1	1	may be elim.
VMOVAPS/D	у,у	1	1	p5	0-1	1	may be elim.
MOVAPS/D MOVUPS/D	x,m128	1	1	p23	3	0.5	
VMOVAPS/D	λ,20		•	β20		0.0	
VMOVUPS/D	y,m256	1	1	p23	3	0.5	AVX
MOVAPS/D	"						
MOVUPS/D	m128,x	1	2	p237 p4	3	1	
VMOVAPS/D							
VMOVUPS/D	m256,y	1	2	p237 p4	4	1	AVX
MOVSS/D	x,x	1	1	p5	1	1	
MOVSS/D	x,m32/64	1	1	p23	3	0.5	
MOVSS/D	m32/64,x	1	2	p237 p4	3	1	
MOVHPS/D	x,m64	1	2	p23 p5	4	1	
MOVHPS/D	m64,x	1	2	p4 p237	3	1	
MOVLPS/D	x,m64	1	2	p23 p5	4	1	
MOVLPS/D	m64,x	1	2	p4 p237	3	1	
MOVHLPS	X,X	1	1	p5	1	1	
MOVLHPS	X,X	1	1	p5	1	1	
MOVMSKPS/D	r32,x	1	1	p0	3	1	
VMOVMSKPS/D	r32,y	1	1	p0	3	1	
MOVNTPS/D	m128,x	1	2	p4 p237	~400	1	
VMOVNTPS/D	m256,y	1	2	p4 p237	~400	1	AVX
SHUFPS/D	x,x,i / v,v,v,i	1	1	p5	1	1	
SHUFPS/D	x,m,i / v,v,m,i	2	2	p5 p23		1	
VPERMILPS/PD	v,v,i	1	1	p5	1	1	AVX
VPERMILPS/PD	v,m,i	2	2	p5 p23		1	AVX
VPERMILPS/PD	V,V,V	1	1	p5	1	1	AVX
VPERMILPS/PD	v,v,m	2	2	p5 p23		1	AVX
VPERM2F128	y,y,y,i	1	1	p5	3	1	AVX
VPERM2F128	y,y,m,i	2	2	p5 p23		1	AVX
VPERMPS	y,y,y	1	1	p5	3	1	AVX2
VPERMPS	y,y,m	1	2	p5 p23		1	AVX2

VPERMPD	y,y,i	1	1	p5	3	1	AVX2
VPERMPD	y,m,i	2	2	p5 p23		1	AVX2
BLENDPS/PD	x,x,i / v,v,v,i	1	1	p015	1	0.33	SSE4.1
BLENDPS/PD	x,m,i / v,v,m,i	2	2	p015 p23		0.5	SSE4.1
BLENDVPS/PD	x,x,xmm0	2	2	2p5	2	2	SSE4.1
BLENDVPS/PD	x,m,xmm0	3	3	2p5 p23		2	SSE4.1
VBLENDVPS/PD	V,V,V,V	2	2	2p5	2	2	AVX
VBLENDVPS/PD	v,v,m,v	3	3	2p5 p23		2	AVX
MOVDDUP	V,V	1	1	p5	1	1	SSE3
MOVDDUP	v,m	1	1	p23	3	0.5	SSE3
VBROADCASTSS	x,m32	1	1	p23	4	0.5	AVX
VBROADCASTSS	y,m32	1	1	p23	5	0.5	AVX
VBROADCASTSS	x,x	1	1	p5	1	1	AVX2
VBROADCASTSS	y,x	1	1	p5	3	1	AVX2
VBROADCASTSD	y,m64	1	1	p23	5	0.5	AVX
VBROADCASTSD	y,x	1	1	p5	3	1	AVX2
VBROADCASTF128	y,m128	1	1	p23	4	0.5	AVX
MOVSH/LDUP	V,V	1	1	p5	1	1	SSE3
MOVSH/LDUP	v,m	1	1	p23	3	0.5	SSE3
UNPCKH/LPS/D	x,x / v,v,v	1	1	p5	1	1	SSE3
UNPCKH/LPS/D	x,m / v,v,m	1	2	p5 p23		1	SSE3
EXTRACTPS	r32,x,i	2	2	p0 p5		1	SSE4.1
EXTRACTPS	m32,x,i	2	3	p0 p5 p23	4	1	SSE4.1
VEXTRACTF128	x,y,i	1	1	p5	3	1	AVX
VEXTRACTF128	m128,y,i	2	2	p23 p4	4	1	AVX
INSERTPS	x,x,i	1	1	p5	1	1	SSE4.1
INSERTPS	x,m32,i	2	2	p23 p5	4	1	SSE4.1
VINSERTF128	y,y,x,i	1	1	p5	3	1	AVX
VINSERTF128	y,y,m128,i	2	2	p015 p23	4	2	AVX
VMASKMOVPS/D	v,v,m	3	3	2p5 p23	4	2	AVX
VMASKMOVPS/D	m128,x,x	4	4	p0 p1 p4 p23	15	1	AVX
VMASKMOVPS/D	m256,y,y	4	4	p0 p1 p4 p23	16	1	AVX
VGATHERDPS	x,[r+s*x],x	10	10	po p. p. p20		6	AVX2
VGATHERDPS	y,[r+s*y],y	14	14			7	AVX2
VGATHERQPS	x,[r+s*x],x	9	9			6	AVX2
VGATHERQPS	x,[r+s*y],x	10	10			6	AVX2
VGATHERDPD	x,[r+s*x],x	7	7			5	AVX2
VGATHERDPD	y,[r+s*x],y	9	9			6	AVX2
VGATHERQPD	x,[r+s*x],x	7	7			5	AVX2
VGATHERQPD	y,[r+s*y],y	9	9			6	AVX2
VOXTTILITOR D	y,[i · O y],y						7,17,12
Conversion							
CVTPD2PS	x,x	2	2	p1 p5	4	1	
CVTPD2PS	x,m128	2	3	p1 p5 p23		1	
VCVTPD2PS	x,y	2	2	p1 p5	5	1	AVX
VCVTPD2PS	x,m256	2	3	p1 p5 p23		1	AVX
CVTSD2SS	x,x	2	2	p1 p5	4	1	
CVTSD2SS	x,m64	2	3	p1 p5 p23		1	
CVTPS2PD	x,x	2	2	p0 p5	2	1	
CVTPS2PD	x,m64	2	2	p0 p23		1	
VCVTPS2PD	y,x	2	2	p0 p5	5	1	AVX
VCVTPS2PD	y,m128	2	2	p0 p23		1	AVX

			_				
CVTSS2SD	x,x	2	2	p0 p5	2	1	
CVTSS2SD	x,m32	2	2	p0 p23		1	
CVTDQ2PS	x,x	1	1	p1	3	1	
CVTDQ2PS	x,m128	1	2	p1 p23		1	
VCVTDQ2PS	y,y	1	1	p1	3	1	AVX
VCVTDQ2PS	y,m256	1	2	p1 p23		1	AVX
	-		1		3	1	
CVT(T) PS2DQ	X,X		2	p1	3	-	
CVT(T) PS2DQ	x,m128	1		p1 p23		1	A) /)/
VCVT(T) PS2DQ	у,у	1	1	p1	3	1	AVX
VCVT(T) PS2DQ	y,m256	1	2	p1 p23		1	AVX
CVTDQ2PD	X,X	2	2	p1 p5	4	1	
CVTDQ2PD	x,m64	2	2	p1 p23		1	
VCVTDQ2PD	y,x	2	2	p1 p5	6	1	AVX
VCVTDQ2PD	y,m128	2	2	p1 p23		1	AVX
CVT(T)PD2DQ	X,X	2	2	p1 p5	4	1	
CVT(T)PD2DQ	x,m128	2	3	p1 p5 p23		1	
VCVT(T)PD2DQ	x,y	2	2	p1 p5	6	1	AVX
VCVT(T)PD2DQ	x,m256	2	3	p1 p5 p23		1	AVX
CVTPI2PS	x,mm	1	1	p1	4	4	
CVTPI2PS	x,m64	1	2	p1 p23		3	
CVT(T)PS2PI	mm,x	2	2	p1 p5	4	1	
CVT(T)PS2PI	mm,m128	2	2	p1 p23	7	1	
CVTPI2PD		2	2		4	1	
	x,mm			p1 p5	4		
CVTPI2PD	x,m64	2	2	p1 p23		1	
CVT(T) PD2PI	mm,x	2	2	p1 p5	4	1	
CVT(T) PD2PI	mm,m128	2	3	p1 p5 p23		1	
CVTSI2SS	x,r32	2	2	p1 p5	4	3	
CVTSI2SS	x,r64	3	3	p1 2p5	5	4	
CVTSI2SS	x,m32	1	2	p1 p23		3	
CVT(T)SS2SI	r32,x	2	2	p0 p1	4	1	
CVT(T)SS2SI	r32,m32	2	3	p0 p1 p23		1	
CVTSI2SD	x,r32/64	2	2	p1 p5	4	3	
CVTSI2SD	x,m32	2	2	p1 p23		3	
CVT(T)SD2SI	r32/64,x	2	2	p0 p1	4	1	
CVT(T)SD2SI	r32,m64	2	3	p0 p1 p23		1	
VCVTPS2PH	x,v,i	2	2	p1 p5	4-6	1	F16C
VCVTPS2PH	m,v,i	3	3	p1 p4 p23		1	F16C
VCVTPH2PS	V,X	2	2	p1 p5	4-6	1	F16C
VCVTPH2PS	v,m	2	2	p1 p23	'	1	F16C
V 0 V 11 1121 0	V ,	_	_	p 1 p20		'	1100
Arithmetic							
ADDSS/D PS/D							
SUBSS/D PS/D	x,x / v,v,v	1	1	p1	3	1	
ADDSS/D PS/D	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	'		Pi			
SUBSS/D PS/D	x,m / v,v,m	1	2	p1 p23		1	
ADDSUBPS/D	x,x / v,v,v	1	1	p1	3	1	SSE3
ADDSUBPS/D	x,x / v,v,v x,m / v,v,m		2	p1 p23		1	SSE3
HADDPS/D	X,111 / V,V,111	'		p i p23		ı	SSES
HSUBPS/D	V V I V V V	3	3	n1 2n5	5	2	SSE3
HADDPS/D	x,x / v,v,v	٥	٥	p1 2p5	ا		SSES
HSUBPS/D	x,m / v,v,m	4	4	p1 2p5 p23		2	SSE3
MULSS/D PS/D			1		3	0.5	JULU
MULSS/D PS/D	x,x / v,v,v	1	2	p01			
1	x,m / v,v,m	1		p01 p23	4.4	0.5	
DIVSS	X,X	1	1	p0	11	2.5	

DIVPS	x,x	1	1	p0	11	5	
DIVSS DIVPS	x,m	1	2	p0 p23		3-5	
DIVSD	x,x	1	1	p0	10-14	4-5	
DIVPD	x,x	1	1	p0	10-14	8	
DIVSD DIVPD		1	2		10-14	4-5	
	x,m	=		p0 p23	47		A) /)/
VDIVPS	y,y,y	3	3	2p0 p15	17	10	AVX
VDIVPS	y,y,m256	4	4	2p0 p15 p23		10	AVX
VDIVPD	y,y,y	3	3	2p0 p15	19-23	16	AVX
VDIVPD	y,y,m256	4	4	2p0 p15 p23		16	AVX
RCPSS/PS	X,X	1	1	p0	5	1	
RCPSS/PS	x,m128	1	2	p0 p23		1	
VRCPPS	y,y	3	3	2p0 p15	7	2	AVX
VRCPPS	y,m256	4	4	2p0 p15 p23	·	2	AVX
CMPccSS/D	y,111200	7	_	2p0 p 10 p20		_	/ (/ /
CMPccPS/D	x,x / v,v,v	1	1	p1	3	1	
CMPccSS/D	A,A / V,V,V	'	ı	μι	3	ı	
CMPccPS/D	x,m / v,v,m	2	2	p1 p23		1	
1 -						-	
(U)COMISS/D	X,X	1	1	p1		1	
(U)COMISS/D	x,m32/64	2	2	p1 p23		1	
MAXSS/D PS/D	,			_			
MINSS/D PS/D	x,x / v,v,v	1	1	p1	3	1	
MAXSS/D PS/D							
MINSS/D PS/D	x,m / v,v,m	1	2	p1 p23		1	
		•		0.4	•		00544
ROUNDSS/D PS/D	v,v,i	2	2	2p1	6	2	SSE4.1
DOLINDSS/D DS/D	v m i	3	3	251 522		2	SSE4.1
ROUNDSS/D PS/D	v,m,i			2p1 p23	40		
DPPS	x,x,i / v,v,v,i	4	4	2p0 p1 p5	12	2	SSE4.1
DPPS	x,m,i / v,v,m,i	6	6	2p0 p1 p5 p23 p6		4	SSE4.1
DPPD	x,x,i	3	3	p0 p1 p5	7	1	SSE4.1
DPPD	x,m128,i	4	4	p0 p1 p5 p23		1	SSE4.1
VFMADD							
(all FMA instr.)	V,V,V	1	1	p01	5	0.5	FMA
VFMADD							
(all FMA instr.)	v,v,m	1	2	p01 p23		0.5	FMA
Math							
SQRTSS	x,x	1	1	p0	11	4	
SQRTPS	x,x	1	1	p0	11	7	
SQRTSS/PS	x,m128	1	2	p0 p23		4-7	
VSQRTPS	·	3	3	2p0 p15	19	14	AVX
	y,y v m256	3 4	4		19	14	
VSQRTPS	y,m256	· · · · · · · · ·		2p0 p15 p23	45.40		AVX
SQRTSD	X,X	1	1	p0	15-16	4-8	
SQRTPD	X,X	1	1	p0	15-16	8-14	
SQRTSD/PD	x,m128	1	2	p0 p23		4-14	
VSQRTPD	y,y	3	3	2p0 p15	27-29	16-28	AVX
VSQRTPD	y,m256	4	4	2p0 p15 p23		16-28	AVX
RSQRTSS/PS	x,x	1	1	p0	5	1	
RSQRTSS/PS	x,m128	1	2	p0 p23	-	1	
VRSQRTPS	у,у у,у	3	3	2p0 p15	7	2	AVX
VRSQRTPS	y,y y,m256	4	4	2p0 p15 2p0 p15 p23	'	2	AVX
VINOUNIFO	y,111230	4	4	2pu p 10 p20			~~^
Logic							
AND/ANDN/OR/ XORPS/PD	vy/vyv	1	1	p5	1	1	
NONFOIFU	x,x / v,v,v	I	ļ i	pο	I	ı	

AND/ANDN/OR/ XORPS/PD	x,m / v,v,m	1	2	p5 p23		1	
Other							
VZEROUPPER		4	4	none		1	AVX
							AVX,
VZEROALL		12	12	none		10	32 bit
							AVX,
VZEROALL		20	20	none		8	64 bit
LDMXCSR	m32	3	3	p0 p6 p23	6	3	
STMXCSR	m32	3	4	p0 p4 p6 p237	7	1	
FXSAVE	m4096	111			66	66	32 bit mode
FXSAVE	m4096	141			66	66	64 bit mode
FXRSTOR	m4096	107			80	80	32 bit mode
FXRSTOR	m4096	115			80	80	64 bit mode
XSAVE		174			70	70	32 bit mode
XSAVE		224			84	84	64 bit mode
XRSTOR		172			111	111	32 bit mode
XRSTOR		173			112	112	64 bit mode
XSAVEOPT	m	114			51	51	

Intel Skylake

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the same data.

Instructions with or without V name prefix behave the same unless otherwise noted.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm register, mm/

x = mmx or xmm register, y = 256 bit ymm register, v = any vector register (mmx, xmm,

ymm). m = memory operand, m32 = 32-bit memory operand, etc.

μορs fused domain:

The number of µops at the decode, rename and allocate stages in the pipeline. Fused

uops count as one.

μορs unfused domain:

The total number of µops for all execution port. Fused µops count as two. Fused macro-

ops count as one. The instruction has µop fusion if this number is higher than the number under fused domain. Some operations are not counted here if they do not go to any

execution port or if the counters are inaccurate.

μορs each port: The number of μops for each execution port. p0 means a μop to execution port 0.

p01means a μop that can go to either port 0 or port 1. p0 p1 means two μops going to

port 0 and 1, respectively.

Port 0: Integer, f.p. and vector ALU, mul, div, branch

Port 1: Integer, f.p. and vector ALU

Port 2: Load Port 3: Load Port 4: Store

Port 5: Integer and vector ALU Port 6: Integer ALU, branch Port 7: Store address

Latency:

This is the delay that the instruction generates in a dependency chain. The numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Where hyperthreading is enabled, the use of the same execution units in the other thread leads to inferior performance. Denormal numbers, NAN's and infinity do not increase the latency. The time unit used is core clock cycles, not the reference clock cycles given by the time stamp counter.

ence clock cycles given by the time stamp counter.

Reciprocal throughput:

The average number of core clock cycles per instruction for a series of independent in-

structions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc-							
tions							
MOV	r,i	1	1	p0156		0.25	
MOV	r8/16,r8/16	1	1	p0156	1	0.25	
MOV	r32/64,r32/64	1	1	p0156	0-1	0.25	may be elim.
MOV	r8l,m	1	2	p23 p0156		0.5	
MOV	r8h,m	1	1	p23		0.5	
MOV	r16,m	1	2	p23 p0156		0.5	
MOV	r32/64,m	1	1	p23	2	0.5	all addressing modes
MOV	m,r	1	2	p237 p4	2	1	
MOV	m,i	1	2	p237 p4		1	
MOVNTI	m,r	2	2	p23 p4	~400	1	

						1	
MOVSX MOVZX	r,r	1	1	p0156	1	0.25	
MOVSXD	100	4		000450		0.5	
MOVSX MOVZX MOVSX MOVZX	r16,m8	1	2	p23 p0156		0.5	all athan
MOVSX MOVZX	r,m	1	1	p23		0.5	all other combinations
CMOVcc	r,r	1	1	p06	1	0.5	Combinations
CMOVcc	r,m	2	2	p06 p23	•	0.5	
XCHG	r,r	3	3	3p0156	2	1	
XCHG	r,m	8	8	Оролоо	23	•	implicit lock
XLAT	.,	3	3	p23 2p0156	7	2	miphot look
PUSH	r	1	2	p237 p4	3	1	
PUSH	i	1	2	p237 p4	Ü	1	
PUSH	m	2	3	p4 2p237		1	
PUSH	stack pointer	2	3	p0156 p237 p4		1	
PUSHF(D/Q)	otaon pointo.	3	4	p1 p4 p237 p06		1	
PUSHA(D)		11	19	p : p : p20; p00		8	not 64 bit
POP	r	1	1	p23	2	0.5	
POP	stack pointer	3	3	p23 2p0156	_	3	
POP	m	2	3	2p237 p4		1	
POPF(D/Q)		9	9	_p_s. p.		20	
POPA(D)		18	18			8	not 64 bit
LAHF SAHF		1	1	p06	1	1	1101 0 1 211
SALC		3	3	3p0156	1	1	not 64 bit
LEA	r16,m	2	2	p1 p05	2-4	1	16 or 32 bit
	,	_	_	p . p . c		·	address size
LEA	r32/64,m	1	1	p15	1	0.5	1 or 2 compo-
							nents in
							address
LEA	r32/64,m	1	1	p1	3	1	3 components
	22/24					_	in address
LEA	r32/64,m	1	1	p1		1	rip relative address
BSWAP	r32	1	1	p15	1	0.5	address
BSWAP	r64	2	2	p06 p15	2	1	
MOVBE	r16,m16	3	3	2p0156 p23	2	0.5-1	MOVBE
MOVBE	r32,m32	2	2	p15 p23		0.5	MOVBE
MOVBE	r64,m64	3	3	2p0156 p23		0.75	MOVBE
MOVBE	m16,r16	2	3	p06 p237 p4		1	MOVBE
MOVBE	m32,r32	2	3	p15 p237 p4		1	MOVBE
MOVBE	m64,r64	3	4	p06 p15 p237 p4		1	MOVBE
WOVBE	11104,104	O	_	P00 P10 P201 P4		•	IVIOVBE
PREFETCHNTA/	m	1	1	p23		0.5	
0/1/2		•	•	P20		0.0	
PREFETCHW	m	1	1	p23		1	PREFETCHW
LFENCE		2		none counted		4	
MFENCE		4	4	p23 p4		33	
SFENCE		2	2	p23 p4		6	
Arithmetic in-							
structions							
ADD SUB	r,r/i	1	1	p0156	1	0.25	
ADD SUB	r,m	1	2	p0156 p23		0.5	
ADD SUB	m,r/i	2	4	2p0156 2p237 p4	5	1	
ADC SBB	r,r/i	1	1	p06	1	1	

ADC SBB	r,m	2	2	p06 p23		1	1
ADC SBB	m,r/i	4	6	3p0156 2p237 p4	5	2	
CMP	r,r/i	1	1	p0156	1	0.25	
CMP			2	·	1	0.23	
INC DEC NEG	m,r/i	1 1	1	p0156 p23	1		
NOT	r			p0156		0.25	
INC DEC NOT	m	3	4	p0156 2p237 p4	5-6	1	
NEG	m	2	4	p0156 2p237 p4	5-6	1	
AAA		2	2	p1 p56	4		not 64 bit
AAS		2	2	p1 p056	4		not 64 bit
DAA DAS		3	3	p1 2p056	4		not 64 bit
AAD		3	3	p1 2p056	4		not 64 bit
AAM		11	11	p0 p1 p5 p6	23	7	not 64 bit
MUL IMUL	r8	1	1	p1	3	1	
MUL IMUL	r16	4	4	p1 p0156	4	2	
MUL IMUL	r32	3	3	p1 p0156	4	1	
MUL IMUL	r64	2	2	p1 p6	3	1	
MUL IMUL	m8	1	2	p1 p23		1	
MUL IMUL	m16	4	5	p1 3p0156 p23		2	
MUL IMUL	m32	3	4	p1 2p0156 p23		2	
MUL IMUL	m64	2	3			1	
IMUL		1	1	p1 p6 p23	3		
	r,r	-		p1	<u> </u>	1	
IMUL	r,m	1	2	p1 p23	_	1	
IMUL	r16,r16,i	2	2	p1 p0156	4	1	
IMUL	r32,r32,i	1	1	p1	3	1	
IMUL	r64,r64,i	1	1	p1	3	1	
IMUL	r16,m16,i	2	3	p1 p0156 p23		1	
IMUL	r32,m32,i	1	2	p1 p23		1	
IMUL	r64,m64,i	1	2	p1 p23		1	
MULX	r32,r32,r32	3	3	p1 2p056	4	1	BMI2
MULX	r32,r32,m32	3	4	p1 2p056 p23		1	BMI2
MULX	r64,r64,r64	2	2	p1 p5	4	1	BMI2
MULX	r64,r64,m64	2	3	p1 p6 p23		1	BMI2
DIV	r8	10	10	p0 p1 p5 p6	23	6	
DIV	r16	10	10	p0 p1 p5 p6	23	6	
DIV	r32	10	10	p0 p1 p5 p6	26	6	
DIV	r64	36	36	p0 p1 p5 p6	35-88	21-83	
IDIV	r8	11	11	p0 p1 p5 p6	24	6	
IDIV	r16	10	10	p0 p1 p5 p6	23	6	
IDIV	r32	10	10	p0 p1 p5 p6	26	6	
IDIV	r64	57	57	p0 p1 p5 p6	42-95	24-90	
CBW		1	1	p0156	1	2.00	
CWDE		1	1	p0156	1		
CDQE			1	p0156	1		
CWD		2	2	p0156	1		
CDQ		1	1	· ·	1		
				p06			
CQO		1	1	p06	1		00540
POPCNT	r,r	1	1	p1	3	1	SSE4.2
POPCNT	r,m	1	2	p1 p23		1	SSE4.2
CRC32	r,r	1	1	p1	3	1	SSE4.2
CRC32	r,m	1	2	p1 p23		1	SSE4.2

Logic instruc-							
AND OR XOR	r,r/i	1	1	p0156	1	0.25	
AND OR XOR	r,m	1	2	p0156 p23		0.5	
AND OR XOR	m,r/i	2	4	2p0156 2p237 p4	5	1	
TEST	r,r/i	1	1	p0156	1	0.25	
TEST	m,r/i	1 1	2	p0156 p23	1	0.23	
SHR SHL SAR		1	1		1	0.5	
SHR SHL SAR	r,i	1	4	p06	l 		
	m,i	3		2p06 p237 p4	2	2 2	
SHR SHL SAR	r,cl		3	3p06			
SHR SHL SAR	m,cl	5	6	3p06 2p23 p4	_	4	- l
ROR ROL	r,1	2	2	2p06	1	1	short form
ROR ROL	r,i	1	1	p06	1	0.5	
ROR ROL	m,i	4	5	2p06 2p237 p4		2	
ROR ROL	r,cl	3	3	3p06	2	2	
ROR ROL	m,cl	5	6	3p06 p23 p4	_	4	
RCR RCL	r,1	3	3	2p06 p0156	2	2	
RCR RCL	m,1	4	6			3	
RCR RCL	r,i	8	8	p0156	6	6	
RCR RCL	m,i	11	11			6	
RCR RCL	r,cl	8	8	p0156	6	6	
RCR RCL	m,cl	11	11			6	
SHRD SHLD	r,r,i	1	1	p1	3	1	
SHRD SHLD	m,r,i	3	5			2	
SHLD	r,r,cl	4	4	p0156	3	2	
SHRD	r,r,cl	4	4	p0156	4	2	
SHRD SHLD	m,r,cl	5	7			4	
SHLX SHRX SARX	r,r,r	1	1	p06	1	0.5	BMI2
SHLX SHRX SARX	r,m,r	2	2	p06 p23		0.5	BMI2
RORX	r,r,i	1	1	p06	1	0.5	BMI2
RORX	r,m,i	2	2	p06 p23		0.5	BMI2
ВТ	r,r/i	1	1	p06	1	0.5	
BT	m,r	10	10			5	
BT	m,i	2	2	p06 p23		0.5	
BTR BTS BTC	r,r/i	1	1	p06	1	0.5	
BTR BTS BTC	m,r	10	11			5	
BTR BTS BTC	m,i	3	4	p06 p4 p23		1	
BSF BSR	r,r	1	1	p1	3	1	
BSF BSR	r,m	1	2	p1 p23		1	
SETcc	r	1	1	p06	1	0.5	
SETcc	m	2	3	p06 p237 p4		1	
CLC		1	0	none		0.25	
STC		1	1	p0156		0.25	
CMC		1	1	p0156	1	1	
CLD STD		3	3	p15 p6		4	
LZCNT	r,r	1	1	p1	3	1	LZCNT
LZCNT	r,m	1 1	2	p1 p23		1	LZCNT
TZCNT	r,r	1 1	1	p1	3	1	BMI1
TZCNT	r,m		2	p1 p23		1	BMI1
ANDN		1 1	1	p1 p23	1	0.5	BMI1
ANDN	r,r,r	1 1	2	p15 p23	1	0.5	BMI1
BLSI BLSMSK	r,r,m	1 1	1	p15 p23	1	0.5	BMI1
BLSR	r,r	'	"	μισ	l I	0.5	ו וואום

BLSI BLSMSK	r,m	1	2	p15 p23		0.5	BMI1
BLSR							
BEXTR	r,r,r	2	2	2p0156	2	0.5	BMI1
BEXTR	r,m,r	3	3	2p0156 p23		1	BMI1
BZHI	r,r,r	1	1	p15	1	0.5	BMI2
BZHI	r,m,r	1	2	p15 p23		0.5	BMI2
PDEP	r,r,r	1	1	p1	3	1	BMI2
PDEP	r,r,m	1 1	2	p1 p23		1	BMI2
PEXT	r,r,r	1	1	p1	3	1	BMI2
PEXT	r,r,m	1	2	p1 p23	Ü	1	BMI2
Control transfer i	instructions						
JMP	short/near	1 1	1	p6		1-2	
				-			
JMP	r		1	p6		2	
JMP	m	1	2	p23 p6		2	
Conditional jump	short/near	1	1	p6		1-2	predicted taken
Conditional jump	short/near	1	1	p06		0.5-1	predicted not taken
Fused arithmetic and branch		1	1	p6		1-2	predicted taken
Fused arithmetic		1	1	p06		0.5-1	predicted not
and branch		'		Poo		0.5-1	taken
J(E/R)CXZ	short	2	2	p0156 p6		0.5-2	taken
LOOP		7	7	po 130 po			
	short					5	
LOOP(N)E	short	11	11	007 4 0		6	
CALL	near	2	3	p237 p4 p6		3	
CALL	r	2	3	p237 p4 p6		2	
CALL	m	3	4	2p237 p4 p6		3	
RET		1	2	p237 p6		1	
RET	i		2			2	
BOUND	r,m	15	15			8	not 64 bit
INTO	.,	5	5			6	not 64 bit
String instruc-							
tions				0.0450.00			
LODSB/W		3	3	2p0156 p23		1	
LODSD/Q		2	2	p0156 p23		1	
REP LODS		5n+12				~2n	
STOS		3	3	p23 p0156 p4		1	
REP STOS		<2n				~0.5n	worst case
REP STOS		2.6/32B				1/32B	best case
							aligned by 32
MOVS		5	5	2p23 p4 2p0156		4	
REP MOVS		~2n				< 1n	worst case
REP MOVS		4/32B				1/32B	best case
SCAS			2	22 220456		4	aligned by 32
SCAS		3	3	p23 2p0156		1	
REP SCAS		≥6n				≥2n	
CMPS		5	5	2p23 3p0156		4	
REP CMPS		≥8n				≥2n	
Synchronization	instructions						
XADD	m,r	4	5			5	
·							·

	1						1
LOCK XADD	m,r	9	9			18	
LOCK ADD	m,r	8	8			18	
CMPXCHG	m,r	5	6			6	
LOCK CMPXCHG	m,r	10	10			18	
CMPXCHG8B	m,r	16	16			11	
LOCK CMPXCHG8B	m,r	20	20			19	
CMPXCHG16B	m,r	23	23			16	
LOCK CMPXCHG16B	m,r	25	25			26	
Other							
NOP (90)		1	0	none		0.25	
Long NOP (0F		1	0	none		0.25	
1F)							
PAUSE		4	4	p6			
ENTER	a,0	12	12			8	
ENTER	a,b	~14+7b	~45+7b		~87+2b		
LEAVE		3	3	2p0156 p23		5	
XGETBV		15	15			9	XGETBV
CPUID		27-118				100-250	
RDTSC		20	20			25	
RDTSCP		22	22			32	RDTSCP
RDPMC		35	35			40	
RDRAND	r	16	16	p23 15p0156		~460	RDRAND
RDSEED	r	16	16	p23 15p0156		~460	RDSEED

Floating point x87 instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc- tions							
FLD	r	1	1	p05	1	0.5	
FLD	m32/64	1	1	p23	3	0.5	
FLD	m80	4	4	2p01 2p23	4	2	
FBLD	m80	43	43		46	22	
FST(P)	r	1	1	p05	1	0.5	
FST(P)	m32/m64	1	2	p4 p237	3	1	
FSTP	m80	7	7	3p0156 2p23 2p4	4	5	
FBSTP	m80	244	226		264	266	
FXCH	r	2	0	none	0	0.5	
FILD	m	1	2	p05 p23	5	1	
FIST(P)	m	3	3	p5 p23 p4	7	1	
FISTTP	m	3	3	p1 p23 p4	7	2	SSE3
FLDZ		1	1	p05		1	
FLD1		2	2	2p05		2	
FLDPI FLDL2E et	c.	2	2	2p05		2	
FCMOVcc	r	4	4	p0 p1 p56	3	2	
FNSTSW	AX	2	2	p0 p0156	6	2	
FNSTSW	m16	2	3	p0 p4 p237	6	1	
FLDCW	m16	3	3	p01 p23 p6	7	2	
FNSTCW	m16	2	3	p237 p4 p6	6	1	
FINCSTP FDECS	TP	1	1	p05	0	0.5	

FFREE(P)	r	1	1	p05		0.5	
FNSAVE	m	133	133	F	176	176	
FRSTOR	m	89	89		175	175	
TROTOR	•••				170	170	
Arithmetic in-							
structions							
FADD(P)							
FSUB(R)(P)	r	1	1	p5	3	1	
FADD(P)	·		•			•	
FSUB(R)(P)	m	2	3	p5 p23		1	
FMUL(P)	r	1	1	p0	5	1	
FMUL(P)	m	2	3	p0 p23		1	
FDIV(R)(P)	r	1	1	p0	14-16	4-5	
1 11 1		1	2		14-10	4-5 4-5	
FDIV(R)(P)	m			p0 p23	4		
FABS		1	1	p0	1 1	1	
FCHS		1	1	p0	1	1	
FCOM(P) FUCOM	r	1	1	p5	3	1	
FCOM(P) FUCOM	m	1	2	p5 p23		1	
FCOMPP FUCOM	IPP	2	2	p0 p5		1	
FCOMI(P)				_			
FUCOMI(P)	r	3	3	p5		1	
FIADD FISUB(R)	m	3	4	2p5 p23		2	
FIMUL	m	2	3	p0 p5 p23		1	
FIDIV(R)	m	2	3	p0 p5 p23			
FICOM(P)	m	2	3	2p5 p23		2	
FTST		1	1	p5	3	1	
FXAM		2	2	2p5	6	2	
FPREM		31	31		26-30	17	
FPREM1		31	31		30-57	17	
FRNDINT		17	17		21	11	
		''					
Math							
FSCALE		27	27		130	130	
FXTRACT		17	17		11	11	
FSQRT		1	1	p0	14-21	4-7	
FSIN		53-105	'	Po	50-120	⊣ -1	
FCOS		53-105			50-120		
FSINCOS		55-105			55-150		
F2XM1		1					
		16-90			65-80		
FYL2X		40-100			103		
FYL2XP1		56			77		
FPTAN		40-112			140-160		
FPATAN		30-160			100-160		
Other							
FNOP		1	1	p05		0.5	
WAIT		2	2	p05		2	
FNCLEX		5	5	•		22	
				p156			
FNINIT		18	18			78	

Integer vector instructions

I	On a manual a	μορs fused	μορs unfused			Recipro- cal through	Q
Instruction	Operands	domain	domain	µops each port	Latency	put	Comments
Move instruc-							
tions				0			
MOVD	r32/64,mm/x	1	1	p0	2	1	
MOVD	m32/64,mm/x	1	2	p237 p4	3	1	
MOVD	mm/x,r32/64	1	1	p5	2	1	
MOVD	mm/x,m32/64	1	1	p23	2	0.5	
MOVQ	r64,mm/x	1	1	p0	2	1	
MOVQ	mm/x,r64	1	1	p5	2	1	
MOVQ	mm,mm	1		p05	1	0.5	
MOVQ	X,X	1		p015	1	0.33	
MOVQ	mm/x,m64	1	1	p23	2	0.5	
MOVQ	m64, mm/x	1	2	p237 p4	3	1	
MOVDQA/U	X,X	1	1	p015	0-1	0.25	may eliminate
MOVDQA/U	x, m128	1	1	p23	2	0.5	
MOVDQA/U	m128, x	1	2	p237 p4	3	1	
VMOVDQA/U	y,y	1	1	p015	0-1	0.25	may eliminate
VMOVDQA/U	y,m256	1	1	p23	3	0.5	AVX
VMOVDQA/U	m256,y	1	2	p237 p4	3	1	AVX
LDDQU	x, m128	1	1	p23	3	0.5	SSE3
MOVDQ2Q	mm, x	2	2	p0 p5	2	1	
MOVQ2DQ	x,mm	2	2	p0 p15	2	1	
MOVNTQ	m64,mm	1	2	p237 p4	~418	1	
MOVNTDQ	m128,x	1	2	p237 p4	~450	1	
VMOVNTDQ	m256,y	1	2	p237 p4	~400	1	AVX2
MOVNTDQA	x, m128	2	2	p23 p015	3	0.5	SSE4.1
VMOVNTDQA	y,m256	2	2	p23 p015	3	0.5	AVX2
PACKSSWB/DW	y,111230			p23 p013		0.5	AVAZ
PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW	''''''			ρο			
PACKUSWB	mm,m64	3	3	p23 2p5		2	
PACKSSWB/DW	111111,11101			p20 2p0		_	
PACKUSWB	x,x / y,y,y	1	1	p5	1	1	
PACKSSWB/DW	,,,,,,						
PACKUSWB	x,m / y,y,m	1	2	p23 p5		1	
PACKUSDW	x,x / y,y,y	1	1	p5	1	1	SSE4.1
PACKUSDW	x,m / y,y,m	1	2	p23 p5		1	SSE4.1
PUNPCKH/L	74,, 3,3,		_	μ=0 μ0			332
BW/WD/DQ	v,v / v,v,v	1	1	p5	1	1	
PUNPCKH/L				•			
BW/WD/DQ	v,m / v,v,m	1	2	p23 p5		1	
PUNPCKH/L							
QDQ	x,x / y,y,y	1	1	p5	1	1	
PUNPCKH/L							
QDQ	x,m / y,y,m	1	2	p23 p5		1	
PMOVSX/ZX BW							
BD BQ DW DQ	x,x	1	1	p5	1	1	SSE4.1
PMOVSX/ZX BW							00544
BD BQ DW DQ	x,m	1	2	p23 p5		1	SSE4.1
VPMOVSX/ZX BW		1	1	p5	3	1	AVX2
BD BQ DW DQ VPMOVSX/ZX BW	y,x	'	'	μο	ا	'	7772
BD BQ DW DQ	y,m	2	2	p5 p23		1	AVX2

PSHUFF V,m / v,v m 2	PSHUFB	www.	1	1	n5	1	1	SSSE3
PSHUFW		v,v / v,v,v			p5	ı	-	
PSHUFN						4		333E3
PSHUFD					· ·	I		
PSHUFL/HW PSHUFL/HW PSHUFL/HW PSHUFL/HW V,mi 2 2 2 p23 p5 1 1 PSLIGNR PSHUFL/HW V,mi 2 2 2 p23 p5 1 1 PSSE3 PALIGNR V,mi / v,v,mi 1 1 p5 1 1 SSE3 PALIGNR P,mi / v,v,mi 2 2 p23 p5 1 1 SSSE3 PBLENDVB X,x,xmm0 1 1 p015 1 1 SSSE3 PBLENDVB X,x,xmm0 2 2 p015 p23 2 SSE4.1 PSLENDVB V,v,m,v 3 3 2p015 p23 2 SSE4.1 PSLENDW V,v,m,v 3 3 2p015 p23 2 AVX2 PBLENDW V,v,m,v 1 1 p5 1 1 SSE4.1 PSLENDW V,v,m,v 1 1 p5 1 1 SSE4.1 PSLENDW V,v,m,v 1 1 p5 1 1 SSE4.1 PSLENDW V,v,m,v 1 1 p5 1 1 SSE4.1 PSLENDW V,v,m,v 1 1 p5 1 1 SSE4.1 PSLENDW V,v,m,v 1 1 p5 1 1 SSE4.1 PSLENDW V,v,m,v 1 1 p5 1 1 SSE4.1 PSLENDW V,v,m,v 1 1 p5 1 1 SSE4.1 PSLENDW V,v,m,v 1 1 p5 3 1 AVX2 VPERMD V,v,mi 2 2 p015 p23 D,5 AVX2 VPERMD V,v,mi 1 2 p5 p23 1 AVX2 VPERMD V,v,mi 1 2 p5 p23 1 AVX2 VPERMD V,v,mi 1 2 p5 p23 1 AVX2 VPERMD V,v,mi 1 2 p5 p23 1 AVX2 VPERMQ V,v,mi 1 2 p5 p23 1 AVX2 VPERMQ V,v,mi 1 2 p5 p23 1 AVX2 VPERMQ V,v,mi 1 2 p5 p23 1 AVX2 VPERMQ V,v,mi 1 2 p5 p23 1 AVX2 VPERMQ V,v,mi 1 p5 3 1 AVX2 VPERMQ V,v,mi 1 p5 3 1 AVX2 VPERMQ V,v,mi 2 PSPARSKMOVO mm,mm 4 4 p0 p4 2p36 4p23 14 AVX2 VPERMOVDIQ V,v,m 2 PSPARSKMOVO M,v,m 2 PSPARSKMOVO M,v,m 2 PSPARSKMOVO M,v,m 2 PSPARSKMOVO M,v,m 2 PSPARSKMOVO M,v,m 2 PSPARSKMOVO M,v,m 2 PSPARSKMOVO M,v,m 3 AVX2 VPERMB V,v,mi 1 p6 3 AVX2 VPERMS AVX2 VPERMB V,v,mi 2 PSPARSW/MO M,v,v 3 APP PSPARSW M,vi 1 AVX2 VPEXTRENWINQ M,v,v 3 AP PSPARSW M,vi 1 AVX2 VPEXTRENWINQ M,v,v 3 AP PSPARSW M,vi 1 AVX2						4	1	
DSHUFL/HW			· ·		· ·	1	1	
PSHUFL/HW PALIGNR V,M,i V,M,i 2 2 2 p23 p5						,	1	
PALIGNR						1	1	
PALIGNR PBLENDVB PBLENDVB X,x,xmm0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1							· ·	
PBLENDVB						1	1	
PBLENDVB		v,m,i / v,v,m,i	2				1	
VPBLENDVB V,V,V,V 2 2 2p015 p23 2 1 AVX2 VPBLENDW x,xi,V,v,w,i 1 1 p5 1 1 SSE4.1 PBLENDW x,m,i/v,v,m,i 2 2 p23 p5 1 1 SSE4.1 VPBLENDD v,v,v,i 1 1 p015 p23 0.5 AVX2 VPBERMD y,y,y 1 1 p5 3 1 AVX2 VPERMD y,y,y 1 1 p5 3 1 AVX2 VPERMQ y,y,i 1 1 p5 3 1 AVX2 VPERMQ128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM2128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM2128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM2128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM21	PBLENDVB	x,x,xmm0	· ·		p015	1	· ·	
VPBLENDVB V,V,m,V 3 3 2p015 p23 2 AVX2 PBLENDW x,xi,I/v,v,vi,I 1 1 p5 1 1 SSE4.1 VPBLENDD v,v,v,i 1 1 p015 1 0.33 AVX2 VPBLENDD v,v,v,i 1 1 p015 p23 0.5 AVX2 VPERMD y,v,y,i 1 1 p5 p23 0.5 AVX2 VPERMD y,y,i 1 1 p5 p23 1 AVX2 VPERMQ y,y,i 1 1 p5 p23 1 AVX2 VPERMQ y,y,i 1 1 p5 p3 1 AVX2 VPERM2128 y,y,y,i 1 1 p5 p3 1 AVX2 VPERM2128 y,y,mi 2 2 p5 p23 1 AVX2 VPERM2128 y,y,mi 2 2 p5 p23 1 AVX2 VPERM2128 y,y,mi 2 2	PBLENDVB	x,m,xmm0		1	p015 p23		2	SSE4.1
PBLENDW	VPBLENDVB	V,V,V,V		2	2p015	2	1	AVX2
PBLENDW	VPBLENDVB	v,v,m,v	3	3	2p015 p23		2	AVX2
VPBLENDD V,V,V,i 1 1 1 p015 1 0.33 AVX2 VPBLENDD V,V,m,i 2 2 2 p015 p23 0.5 AVX2 VPERMD Y,Y,y 1 1 p5 3 1 AVX2 VPERMD Y,Y,M 1 2 p5 p23 1 AVX2 VPERMQ Y,Y,I 1 1 p5 3 1 AVX2 VPERMQ Y,Y,II 1 1 p5 3 1 AVX2 VPERM2I128 Y,Y,JI,I 1 1 p5 3 1 AVX2 VPERM2I128 Y,Y,JI,I 1 1 p5 3 1 AVX2 VPERM2I128 Y,Y,JI,JI 1 1 p5 3 1 AVX2 VPERM2I128 Y,Y,JI,JI 1 1 p5 3 1 AVX2 VPERM2INGO T,X,X 10 10 4p04 2p56 4p23 14 </td <td>PBLENDW</td> <td>x,x,i / v,v,v,i</td> <td>1</td> <td>1</td> <td>p5</td> <td>1</td> <td>1</td> <td>SSE4.1</td>	PBLENDW	x,x,i / v,v,v,i	1	1	p5	1	1	SSE4.1
VPBLENDD V,V,v,i 1 1 p015 1 0.33 AVX2 VPBLENDD V,v,m,i 2 2 p015 p23 0.5 AVX2 VPERMD Y,y,y 1 1 p5 3 1 AVX2 VPERMD Y,y,y 1 1 p5 3 1 AVX2 VPERMQ Y,y,i 1 1 p5 3 1 AVX2 VPERMQ1128 Y,y,m,i 2 2 p5 p23 1 AVX2 VPERM21128 Y,y,m,i 2 2 p5 p23 1 AVX2 VPERM21128 Y,y,m,i 2 2 p5 p23 1 AVX2 VPERM21128 Y,y,m,i 4 4 p0 p4 2p23 -450 2 MASKMOVDQU m,x 10 4p04 2p56 4p23 18-500 6 VPMASKMOVD/Q m,v,m 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVD/Q m,v,i<	PBLENDW	x,m,i / v,v,m,i	2	2	p23 p5		1	SSE4.1
VPBLENDD v,v,m,i 2 2 p015 p23 0.5 AVX2 VPERMD y,y,y 1 1 p5 3 1 AVX2 VPERMD y,y,m 1 2 p5 p23 1 AVX2 VPERMQ y,y,i 1 1 p5 3 1 AVX2 VPERMQ y,m,i 2 2 p5 p23 1 AVX2 VPERMU128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERMU128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERMU128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERMI2128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERMSKMOVDQ mm,mm 4 4 p042p23 ~450 2 MASKMOVDQ mx,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVD/Q mx,v 3 3<	VPBLENDD		1	1		1	0.33	AVX2
VPERMD y,y,y 1 1 p5 3 1 AVX2 VPERMD y,y,m 1 2 p5 p23 1 AVX2 VPERMQ y,y,i 1 1 p5 3 1 AVX2 VPERMQ y,m,i 2 2 p5 p23 1 AVX2 VPERM2I128 y,y,y,i 1 1 p5 3 1 AVX2 VPERM2I128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM2I128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM2I128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM3KMOVDQ mmmm 4 4 p0 p4 p223 ~450 2 MASKMOVDQ v,v,m 2 2 p23 p15 4 0.5 AVX2 VPMASKMOVDQ m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVDQ r,x,	VPBLENDD		2	2	· ·			AVX2
VPERMD y,y,m 1 2 p5 p23 1 AVX2 VPERMQ y,y,i 1 1 p5 p23 1 AVX2 VPERMQ y,m,i 2 2 p5 p23 1 AVX2 VPERM2I128 y,y,y,i 1 1 p5 p23 1 AVX2 VPERM2I128 y,y,m,i 2 2 p0 p4 p23 1 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p5 3 1 SE4.1 PEXTRB/W/D/Q r,x,i 2 <td< td=""><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td><td></td></td<>						3		
VPERMQ y,y,i 1 1 p5 3 1 AVX2 VPERMQ y,m,i 2 2 p5 p23 1 AVX2 VPERM2I128 y,y,y,i 1 1 p5 3 1 AVX2 VPERM2I128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM2I128 y,y,m,i 2 2 p5 p23 1 AVX2 MASKMOVDQ mm,mm 4 4 p0 p4 2p23 ~450 2 MASKMOVDDU x,x 10 10 4p04 2p56 4p23 18-500 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVD/Q m,v,i 2 2 p0 p5 3 1 SSE4.1						Ū	-	
VPERMQ y,m,i 2 2 p5 p23 1 AVX2 VPERM2I128 y,y,y,i 1 1 p5 3 1 AVX2 VPERM2I128 y,y,m,i 2 2 p5 p23 1 AVX2 VPERM2I128 y,y,m,i 2 2 p5 p23 1 AVX2 MASKMOVDQ mm,mm 4 4 404 2p56 4p23 18-500 6 VPMASKMOVDQ x,y,m 2 2 p0 p4 2p3 14 1 AVX2 VPMASKMOVDQ m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVDQ m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVDQ m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMOVMSKB r,v 1 1 p0 2-3 1 T PSE4.1 PSE4.1 PSE4.1 PSE4.1 PSE4.1 PSE4.1 PSE4.1 PSE4.1						3	-	
VPERM2I128 y,y,y,i 1 1 p5 p23 1 AVX2 VPERM2I128 y,y,m,i 2 2 p5 p23 1 AVX2 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~450 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 18-500 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 PMOVMSKB r,v 1 1 p0 2-3 1 SE4.1 PEXTRBM/D/Q r,x,i 2 2 p0 p5 3 1 SSE4.1 VEXTRACTI128 m,y,i 2 2 p5 3	· ·					0		
VPERM2I128 y,y,m,i 2 2 p5 p23 7450 2 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~450 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 18-500 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 PMOVMSKB r,v 1 1 p0 2-3 1 PEXTRBW/ID/Q r,x,i 2 2 p0 p5 3 1 SSE4.1 PEXTRB/WID/Q m,x,i 2 2 p0 p5 3 1 SSE4.1 PEXTRB/WID/Q m,x,i 2 2 p23 p4 p5 1 SSE4.1 PEXTRB/WID/Q m,x,i 2 2 p23 p4 4 1 AVX2 VEXTRACTI128 m,y,i 2 2 p25 p5 3 2 SSE4.1		I -				2	· ·	
MASKMOVQ MASKMOVDQU mm,mm x,x 4 4 4 p0 p4 2p23 4p04 2p56 4p23 ~450 2 VPMASKMOVDQU VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPX,i 1 1 0 0 23 p015 4 4 0.5 AVX2 AVX2 PDF VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPMASKMOVD/Q VPX,i 1 1 p0 p2 p3 p4 2-3 1 1 AVX2 PDF 1 AVX2 PDF 3 1 SSE4.1 AVX2 PDF 1 AVX2 PDF 3 1 SSE4.1 AVX2 PDF 2 3 2 2 2						3	· ·	
MASKMOVDQU x,x 10 10 4p04 2p56 4p23 18-500 6 VPMASKMOVD/Q V,v,m 2 2 p23 p015 4 0.5 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 PMOVMSKB r,v 1 1 p0 p5 3 1 PPXTRB/W/DQ PEXTRB/W/D/Q m,x,i 2 2 p0 p5 3 1 SSE4.1 PEXTRB/W/D/Q m,x,i 2 3 p23 p4 p5 1 SSE4.1 VEXTRACTI128 x,y,i 1 1 p5 3 1 AVX2 VEXTRACTI128 m,y,i 2 2 p23 p4 4 1 AVX2 VEXTRACTI128 m,y,i 2 2 p53 p5 3 2 SSE4.1 PINSRB m,r32,i 2 2 p23 p5 1 SSE4.1 PINSRW mm/x,m16,i 2 2 p23 p5						450	· ·	AVAZ
VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 AVX2 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 14 1 AVX2 PMOVMSKB r,v 1 1 p0 p5 3 1 SSE4.1 PEXTRB/W/D/Q m,x,i 2 2 p0 p5 3 1 SSE4.1 PEXTRB/W/D/Q m,x,i 2 3 p23 p4 p5 1 SSE4.1 PEXTRB/W/D/Q m,x,i 2 3 p23 p4 p5 1 SSE4.1 VEXTRACTI128 m,y,i 2 2 p23 p4 4 1 AVX2 VEXTRACTI128 m,y,i 2 2 p23 p5 3 2 SSE4.1 PINSRB m,r32,i 2 2 p25 3 2 SSE4.1 PINSRW mm/x,r32,i 2 2 p23 p5 1 SSE4.1 PINSRD/Q x,r32,i 2 2 p23 p5 1								
VPMASKMOVD/Q PMOVMSKB m,v,v 3 3 p0 p4 p23 14 1 AVX2 PMOVMSKB r,v 1 1 p0 2-3 1 PEXTRB/W/D/Q r,x,i 2 2 p0 p5 3 1 SSE4.1 PEXTRB/W/D/Q m,x,i 2 3 p23 p4 p5 1 SSE4.1 VEXTRACTI128 x,y,i 1 1 p5 3 1 AVX2 VEXTRACTI128 m,y,i 2 2 p23 p4 4 1 AVX2 VEXTRACTI128 x,y32,i 2 2 2p5 3 2 SSE4.1 VEXTRACTI128 x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRB x,r32,i 2 2 p5 3 2 SSE4.1 PINSRW mm/x,m16,i 2 2 p23 p5 1 SSE4.1 PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1<								A) ()/O
PMOVMSKB								
PEXTRB/W/D/Q								AVX2
PEXTRB/W/D/Q							· ·	
VEXTRACTI128 x,y,i 1 1 p5 3 1 AVX2 VEXTRACTI128 m,y,i 2 2 p23 p4 4 1 AVX2 PINSRB x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRB x,m8,i 2 2 p23 p5 1 SSE4.1 PINSRW mm/x,r32,i 2 2 p5 3 2 PINSRD/Q x,r32,i 2 2 p23 p5 1 SSE4.1 PINSRD/Q x,r32,i 2 2 p23 p5 1 SSE4.1 PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1 PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1 VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VPBROADCAST B/W x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST B/W						3	-	
VEXTRACTI128 m,y,i 2 2 p23 p4 4 1 AVX2 PINSRB x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRB x,m8,i 2 2 p23 p5 1 SSE4.1 PINSRW mm/x,r32,i 2 2 p5 3 2 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1 VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VPBROADCAST B/W/D/Q x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST B/W/D/Q y,m8/16 2 2 p23 p5 7 1 <								
PINSRB x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRB x,m8,i 2 2 p23 p5 1 SSE4.1 PINSRW mm/x,r32,i 2 2 p5 3 2 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1 VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VPBROADCAST B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W x,m32/64 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5		x,y,i		· ·			· ·	
PINSRB x,m8,i 2 2 p23 p5 1 SSE4.1 PINSRW mm/x,r32,i 2 2 p5 3 2 PINSRW mm/x,m16,i 2 2 p23 p5 1 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,m32,i 2 2 2p23 p5 1 SSE4.1 PINSRD/Q x,m32,i 2 2 2p23 p5 1 SSE4.1 VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VPBROADCAST y,y,m,i 2 2 p015 p23 3 0.5 AVX2 VPBROADCAST x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST y,m32/64 1 1 p5 3 1 AVX2 VPBROADCAST y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST							· ·	
PINSRW mm/x,r32,i 2 2 p5 3 2 PINSRW mm/x,m16,i 2 2 p23 p5 1 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1 VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VINSERTI128 y,y,m,i 2 2 p015 p23 3 0.5 AVX2 VPBROADCAST x,x 1 1 p5 1 1 AVX2 VPBROADCAST x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST y,x 1 1 p5 3 1 AVX2 VPBROADCAST y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST y,m8/16 2 2 p23 p5 7 1 AVX2	PINSRB	x,r32,i		1	2p5	3	2	
PINSRW mm/x,m16,i 2 2 p23 p5 1 PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1 VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VINSERTI128 y,y,m,i 2 2 p015 p23 3 0.5 AVX2 VPBROADCAST B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W/D/Q x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST B/W/D/Q y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 <td>PINSRB</td> <td>x,m8,i</td> <td>2</td> <td>1</td> <td>p23 p5</td> <td></td> <td>1</td> <td>SSE4.1</td>	PINSRB	x,m8,i	2	1	p23 p5		1	SSE4.1
PINSRD/Q x,r32,i 2 2 2p5 3 2 SSE4.1 PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1 VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VINSERTI128 y,y,m,i 2 2 p015 p23 3 0.5 AVX2 VPBROADCAST B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST y,m128 <td>PINSRW</td> <td>mm/x,r32,i</td> <td>2</td> <td>2</td> <td>p5</td> <td>3</td> <td>2</td> <td></td>	PINSRW	mm/x,r32,i	2	2	p5	3	2	
PINSRD/Q x,m32,i 2 2 p23 p5 1 SSE4.1 VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VINSERTI128 y,y,m,i 2 2 p015 p23 3 0.5 AVX2 VPBROADCAST B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W/D/Q x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2	PINSRW	mm/x,m16,i	2	2	p23 p5		1	
VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VINSERTI128 y,y,m,i 2 2 p015 p23 3 0.5 AVX2 VPBROADCAST B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST B/W/D/Q y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2	PINSRD/Q	x,r32,i	2	2	2p5	3	2	SSE4.1
VINSERTI128 y,y,x,i 1 1 p5 3 1 AVX2 VINSERTI128 y,y,m,i 2 2 p015 p23 3 0.5 AVX2 VPBROADCAST B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST B/W/D/Q x,m32/64 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2	PINSRD/Q	x,m32,i	2	2	p23 p5		1	SSE4.1
VINSERTI128 y,y,m,i 2 2 p015 p23 3 0.5 AVX2 VPBROADCAST B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST B/W/D/Q y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2	VINSERTI128		1	1	p5	3	1	AVX2
VPBROADCAST B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST B/W/D/Q y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2	VINSERTI128	1	2	2	· ·		0.5	AVX2
B/W/D/Q x,x 1 1 p5 1 1 AVX2 VPBROADCAST B/W x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST B/W y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		, ,,,,			' '			
B/W x,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST B/W/D/Q y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		X,X	1	1	p5	1	1	AVX2
VPBROADCAST x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST y,x 1 1 p5 3 1 AVX2 VPBROADCAST y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2								
D/Q x,m32/64 1 1 p23 4 0.5 AVX2 VPBROADCAST B/W/D/Q y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		x,m8/16	2	2	p23 p5	7	1	AVX2
VPBROADCAST B/W/D/Q y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		00/04						1100
B/W/D/Q y,x 1 1 p5 3 1 AVX2 VPBROADCAST B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		x,m32/64	1	1	p23	4	0.5	AVX2
VPBROADCAST y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		V V	4	4	25	2	1	A1/V2
B/W y,m8/16 2 2 p23 p5 7 1 AVX2 VPBROADCAST D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		у,х	'	'	μυ	J	l	AVA2
VPBROADCAST y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		v m8/16	2	2	n23 n5	7	1	AVX2
D/Q y,m32/64 1 1 p23 3 0.5 AVX2 VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		j,,,,,,	_	_	P20 P0	'	'	, , , , , _
VBROADCASTI128 y,m128 1 1 p23 3 0.5 AVX2		y,m32/64	1	1	p23	3	0.5	AVX2
		1 -	1	1				
-	VPGATHERDD	x,[r+s*x],x	4	4	p0 p1 p23 p5		4	AVX2

VDCATUEDDD			1 4	04005			A) ()(0
VPGATHERDD	y,[r+s*y],y	4	4	p0 p1 p23 p5		5	AVX2
VPGATHERQD	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
VPGATHERQD	x,[r+s*y],x	4	4	p0 p1 p23 p5		4	AVX2
VPGATHERDQ	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
VPGATHERDQ	y,[r+s*x],y	4	4	p0 p1 p23 p5		4	AVX2
VPGATHERQQ	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
VPGATHERQQ	y,[r+s*y],y	4	4	p0 p1 p23 p5		4	AVX2
Arithmetic in-							
structions							
PADD/SUB(S,US)		_		045		0.00	
B/W/D/Q	v,v / v,v,v	1	1	p015	1	0.33	
PADD/SUB(S,US) B/W/D/Q	v,m / v,v,m	1	2	p015 p23		0.5	
PHADD(S)W/D	V,111 / V,V,111	!	_	p013 p23		0.5	
PHSUB(S)W/D	v,v / v,v,v	3	3	p01 2p5	3	2	SSSE3
PHADD(S)W/D	V, V / V, V, V			p012p0	0	_	OOOLO
PHSUB(S)W/D	v,m / v,v,m	4	4	p01 2p5 p23		2	SSSE3
PCMPEQB/W/D	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	•	•	po 1 2po p2o		_	00020
PCMPGTB/W/D	mm,mm	1	1	р0	1	1	
PCMPEQB/W/D	,						
PCMPGTB/W/D	x,x / y,y,y	1	1	p01	1	0.5	
PCMPEQB/W/D				•			
PCMPGTB/W/D	x,m / y,y,m	1	2	p01 p23		0.5	
PCMPEQQ	v,v / v,v,v	1	1	p01	1	0.5	SSE4.1
PCMPEQQ	v,m / v,v,m	1	2	p01 p23		0.5	SSE4.1
PCMPGTQ	v,v / v,v,v	1	1	p5	3	1	SSE4.2
PCMPGTQ	v,m / v,v,m	1	2	p5 p23		1	SSE4.2
PMULL/HW							
PMULHUW	mm,mm	1	1	p0	5	1	
PMULL/HW							
PMULHUW	x,x / y,y,y	1	1	p01	5	0.5	
PMULL/HW			_				
PMULHUW	x,m / y,y,m	1	2	p01 p23		0.5	
PMULHRSW	mm,mm	1	1	p0	5	1	SSSE3
PMULHRSW	x,x / y,y,y	1	1	p01	5	0.5	SSSE3
PMULHRSW	x,m / y,y,m	1	2	p01 p23		0.5	SSSE3
PMULLD	x,x / y,y,y	2	2	2p01	10	1	SSE4.1
PMULLD	x,m / y,y,m	3	3	2p01 p23		1	SSE4.1
PMULDQ	x,x / y,y,y	1	1	p01	5	0.5	SSE4.1
PMULDQ	x,m / y,y,m	1	2	p01 p23		0.5	SSE4.1
PMULUDQ	mm,mm	1	1	p0	5	1	
PMULUDQ	x,x / y,y,y	1	1	p01	5	0.5	
PMULUDQ	x,m / y,y,m	1	2	p01 p23		0.5	
PMADDWD	mm,mm	1	1	p0	5	1	
PMADDWD	x,x / y,y,y	1	1	p01	5	0.5	
PMADDWD	x,m / y,y,m	1	2	p01 p23		0.5	
PMADDUBSW	mm,mm	1	1	p0	5	1	SSSE3
PMADDUBSW	x,x / y,y,y	1	1	p01	5	0.5	SSSE3
PMADDUBSW	x,m / y,y,m	1	2	p01 p23		0.5	SSSE3
PAVGB/W	mm,mm	1	1	p0	1	1	
PAVGB/W	x,x / y,y,y	1	1	p01	1	0.5	
PAVGB/W	x,m / y,y,m	1	2	p01 p23		0.5	
1		1	1		'	'	'

			OI.	yiako			
PMIN/PMAX							
SB/SW/SD							
UB/UW/UD	mm,mm	1	1	p0	1	1	SSE4.1
PMIN/PMAX							
SB/SW/SD	,	_		0.4	_	0.5	00544
UB/UW/UD	x,x / y,y,y	1	1	p01	1	0.5	SSE4.1
PMIN/PMAX							
SB/SW/SD UB/UW/UD	y m / y y m	1	2	n01 n22		0.5	SSE4.1
PHMINPOSUW	x,m / y,y,m	1	1	p01 p23	4		SSE4.1 SSE4.1
	X,X		1	p0	4	1	SSE4.1 SSE4.1
PHMINPOSUW	x,m128	1	2	p0 p23	_	1	
PABSB/W/D	mm,mm	1	1	p0	1	1	SSSE3
PABSB/W/D	x,x / y,y	1	1	p01	1	0.5	SSSE3
PABSB/W/D	x,m / y,m	1	2	p01 p23		0.5	SSSE3
PSIGNB/W/D	mm,mm	1	1	p0	1	1	SSSE3
PSIGNB/W/D	x,x / y,y,y	1	1	p01	1	0.5	SSSE3
PSIGNB/W/D	x,m / y,y,m	1	2	p01 p23		0.5	SSSE3
PSADBW	v,v / v,v,v	1	1	p5	3	1	
PSADBW	v,m / v,v,m	1	2	p5 p23		1	
MPSADBW	x,x,i / v,v,v,i	2	2	2p5	4	2	SSE4.1
MPSADBW	x,m,i / v,v,m,i	3	3	2p5 p23		2	SSE4.1
Logic instruc-							
tions							
PAND PANDN							
POR PXOR	mm,mm	1	1	p05	1	0.5	
PAND PANDN	,	_		0.45	_	0.00	
POR PXOR	x,x / y,y,y	1	1	p015	1	0.33	
PAND PANDN		4		m015 m02		0.5	
POR PXOR	v,m / v,v,m	1	2	p015 p23	_	0.5	00544
PTEST	V,V	2 2	2	p0 p5	3	1	SSE4.1
PTEST	v,m	2	3	p0 p5 p23		1	SSE4.1
PSLLW/D/Q							
PSRLW/D/Q PSRAW/D/Q	mm,mm	1	1	p0	1	1	
PSLLW/D/Q	111111,111111			ρū	!	, I	
PSRLW/D/Q							
PSRAW/D/Q	mm,m64	2	2	p0 p23		1	
PSLLW/D/Q			_	F			
PSRLW/D/Q							
PSRAW/D/Q	x,x / v,v,x	2	2	p01 p5	1	1	
PSLLW/D/Q							
PSRLW/D/Q							
PSRAW/D/Q	x,m / v,v,m	2	2	p01 p23		0.5	
PSLLW/D/Q							
PSRLW/D/Q		_			_	_	
PSRAW/D/Q	mm,i	1	1	p0	1	1	
PSLLW/D/Q							
PSRLW/D/Q	vi/vvi	1	1	n01	4	0.5	
PSRAW/D/Q VPSLLVD/Q	x,i / y,y,i	1	1	p01	1	0.5	
VPSRAVD							
VPSRLVD/Q	V,V,V	1	1	p01	1	0.5	AVX2
VPSLLVD/Q	, , , , ,	•	'		'	0.0	,,,,,,
VPSRAVD							
VPSRLVD/Q	v,v,m	1	2	p01 p23		0.5	AVX2
1			1		1		1

PSLLDQ							
PSRLDQ	x,i / v,v,i	1	1	p5	1	1	
String instruc- tions							
PCMPESTRI	x,x,i	8	8	6p05 2p16	12	4	SSE4.2
PCMPESTRI	x,m128,i	8	8	3p0 2p16 2p5 p23		4	SSE4.2
PCMPESTRM	x,x,i	9	9	3p0 2p16 4p5	9	5	SSE4.2
PCMPESTRM	x,m128,i	9	9	6p05 2p16 p23		5	SSE4.2
PCMPISTRI	x,x,i	3	3	3p0	10	3	SSE4.2
PCMPISTRI	x,m128,i	4	4	3p0 p23		3	SSE4.2
PCMPISTRM	x,x,i	3	3	3p0	9	3	SSE4.2
PCMPISTRM	x,m128,i	4	4	3p0 p23		3	SSE4.2
Encryption instru	uctions						
PCLMULQDQ	x,x,i	1	1	p5	7	1	CLMUL
PCLMULQDQ	x,m,i	2	2	p5 p23		1	CLMUL
AESDEC,							
AESDECLAST,							
AESENC,			_		_	_	450
AESENCLAST	X,X	1	1	р0	4	1	AES
AESDEC,							
AESDECLAST, AESENC,							
AESENCLAST	x,m	2	2	p0 p23		1.5	AES
AESIMC	x,x	2	2	2p0	8	2	AES
AESIMC	x,m	3	3	2p0 p23		2	AES
AESKEYGENAS	,						
SIST	x,x,i	13	13	p0 p5	12	12	AES
AESKEYGENAS							
SIST	x,m,i	13	13			12	AES
Other							
EMMS		10	10	p05		6	

Floating point XMM and YMM instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instruc- tions							
MOVAPS/D	X,X	1	1	p015	0-1	0.25	may eliminate
VMOVAPS/D	y,y	1	1	p015	0-1	0.25	may eliminate
MOVAPS/D MOVUPS/D	x,m128	1	1	p23	2	0.5	
VMOVAPS/D VMOVUPS/D MOVAPS/D	y,m256	1	1	p23	3	0.5	AVX
MOVUPS/D VMOVAPS/D	m128,x	1	2	p237 p4	3	1	
VMOVUPS/D	m256,y	1	2	p237 p4	3	1	AVX
MOVSS/D	X,X	1	1	p5	1	1	
MOVSS/D	x,m32/64	1	1	p23	3	0.5	
MOVSS/D	m32/64,x	1	2	p237 p4	3	1	

1	1				ı	ı	
MOVHPS/D	x,m64	1	2	p23 p5	4	1	
MOVHPS/D	m64,x	1	2	p4 p237	3	1	
MOVLPS/D	x,m64	1	2	p23 p5	4	1	
MOVLPS/D	m64,x	1	2	p4 p237	3	1	
MOVHLPS	x,x	1	1	p5	1	1	
MOVLHPS	x,x	1	1	p5	1	1	
MOVMSKPS/D	r32,x	1	1	p0	2	1	
VMOVMSKPS/D	r32,y	1	1	p0	3	1	
MOVNTPS/D	m128,x	1	2	p4 p237	~400	1	
VMOVNTPS/D	m256,y	1	2	p4 p237	~400	1	AVX
SHUFPS/D	x,x,i / v,v,v,i	1	1	p5	1	1	
SHUFPS/D	x,m,i / v,v,m,i	2	2	p5 p23		1	
VPERMILPS/PD	v,v,i	1	1	p5	1	1	AVX
VPERMILPS/PD	v,m,i	2	2	p5 p23		1	AVX
VPERMILPS/PD	V,V,V	1	1	p5	1	1	AVX
VPERMILPS/PD	v,v,m	2	2	p5 p23	•	1	AVX
VPERM2F128	y,y,y,i	1	1	p5	3	1	AVX
VPERM2F128	y,y,y,i y,y,m,i	2	2	p5 p23		1	AVX
VPERMPS		1	1	p5 p25 p5	3	1	AVX AVX2
VPERMPS	у,у,у	1	2	•	J	1	AVX2 AVX2
VPERMPD	y,y,m	1	1	p5 p23	3	1	AVX2 AVX2
	y,y,i	_		p5	<u>ى</u>	1	
VPERMPD	y,m,i	2	2	p5 p23	_	0.00	AVX2
BLENDPS/PD	x,x,i / v,v,v,i	1	1	p015	1	0.33	SSE4.1
BLENDPS/PD	x,m,i / v,v,m,i	2	2	p015 p23		0.5	SSE4.1
BLENDVPS/PD	x,x,xmm0	1	1	p015	1	1	SSE4.1
BLENDVPS/PD	x,m,xmm0	2	2	p015 p23	_	1	SSE4.1
VBLENDVPS/PD	V,V,V,V	2	2	2p015	2	1	AVX
VBLENDVPS/PD	v,v,m,v	3	3	2p015 p23		1	AVX
MOVDDUP	V,V	1	1	p5	1	1	SSE3
MOVDDUP	v,m	1	1	p23	3	0.5	SSE3
VBROADCASTSS	x,m32	1	1	p23	2	0.5	AVX
VBROADCASTSS	y,m32	1	1	p23	3	0.5	AVX
VBROADCASTSS	x,x	1	1	p5	1	1	AVX2
VBROADCASTSS	y,x	1	1	p5	3	1	AVX2
VBROADCASTSD	y,m64	1	1	p23	3	0.5	AVX
VBROADCASTSD	y,x	1	1	p5	3	1	AVX2
VBROADCASTF128	y,m128	1	1	p23	3	0.5	AVX
MOVSH/LDUP	V,V	1	1	p5	1	1	SSE3
MOVSH/LDUP	v,m	1	1	p23	3	0.5	SSE3
UNPCKH/LPS/D	x,x / v,v,v	1	1	p5	1	1	SSE3
UNPCKH/LPS/D	x,m / v,v,m	1	2	p5 p23		1	SSE3
EXTRACTPS	r32,x,i	2	2	p0 p5		1	SSE4.1
EXTRACTPS	m32,x,i	2	3	p4 p5 p23	5	1	SSE4.1
VEXTRACTF128	x,y,i	1	1	p5	3	1	AVX
VEXTRACTF128	m128,y,i	2	2	p23 p4	6	1	AVX
INSERTPS	x,x,i	1	1	p5	1	1	SSE4.1
INSERTPS	x,m32,i	2	2	p23 p5	4	1	SSE4.1
VINSERTF128	y,y,x,i	1	1	p5	3	1	AVX
VINSERTF128	y,y,x,i y,y,m128,i	2	2	p015 p23	5	0.5	AVX
VMASKMOVPS/D	v,v,m	2	2	p015 p23	3	0.5	AVX
VMASKMOVPS/D	m128,x,x	4	4	p0 r3 p23 p0 p4 p23	13	1	AVX
VMASKMOVPS/D	m256,y,y	4	4	p0 p4 p23 p0 p4 p23	13	1	AVX
VGATHERDPS	x,[r+s*x],x	4	4	p0 p4 p23 p0 p1 p23 p5	12	4	AVX AVX2
VOATHERDES	^,[ı · ɔ ^],^	-	-	PO P 1 P20 P0	14		~v^2

VGATHERDPS	v [r+c*v] v		4	n0 n1 n23 n5	13	5	AVX2
VGATHERDPS	y,[r+s*y],y	5	5	p0 p1 p23 p5	13	2	AVX2 AVX2
· ·	x,[r+s*x],x	4	4	p0 p1 p23 p5		4	AVX2 AVX2
VGATHERQPS	x,[r+s*y],x		5	p0 p1 p23 p5			
VGATHERDPD	x,[r+s*x],x	5		p0 p1 p23 p5		2 4	AVX2
VGATHERDPD	y,[r+s*x],y	4	4	p0 p1 p23 p5			AVX2
VGATHERQPD	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
VGATHERQPD	y,[r+s*y],y	4	4	p0 p1 p23 p5		4	AVX2
Conversion							
CVTPD2PS	x,x	2	2	p01 p5	5	1	
CVTPD2PS	x,m128	2	3	p01 p5 p23		1	
VCVTPD2PS	x,111120 x,y	2	2	p01 p5 p25	7	1	AVX
VCVTPD2PS	x,m256	2	3	p01 p5 p23	'	1	AVX
CVTSD2SS	X,111250 X,X	2	2	p01 p3 p23	5	1	AVA
CVTSD2SS	x,m64	2	3	p01 p5 p23	J	1	
CVT9D233 CVTPS2PD	•	2	2	p01 p5 p25	5	1	
CVTPS2PD	X,X	1	2	p01 p5 p23	3	0.5	
VCVTPS2PD	x,m64	2	2		7	1	AVX
	y,x			p01 p5	/	-	
VCVTPS2PD	y,m128	1	2	p01 p5 p23	_	0.5	AVX
CVTSS2SD	X,X	2	2	p01 p5	5	2	
CVTSS2SD	x,m32	1	2	p01 p5 p23		2	
CVTDQ2PS	X,X	1	1	p01	4	0.5	
CVTDQ2PS	x,m128	1	2	p01 p23	_	0.5	
VCVTDQ2PS	y,y	1	1	p01	4	0.5	AVX
VCVTDQ2PS	y,m256	1	2	p01 p23		0.5	AVX
CVT(T) PS2DQ	X,X	1	1	p01	4	0.5	
CVT(T) PS2DQ	x,m128	1	2	p01 p23		0.5	
VCVT(T) PS2DQ	y,y	1	1	p01	4	0.5	AVX
VCVT(T) PS2DQ	y,m256	1	2	p01 p23		0.5	AVX
CVTDQ2PD	X,X	2	2	p01 p5	5	1	
CVTDQ2PD	x,m64	2	2	p01 p23		0.5	
VCVTDQ2PD	y,x	2	2	p01 p5	7	1	AVX
VCVTDQ2PD	y,m128	1	2	p01 p23		0.5	AVX
CVT(T)PD2DQ	x,x	2	2	p01 p5	5	1	
CVT(T)PD2DQ	x,m128	3	3	p01 p23 p5		1	
VCVT(T)PD2DQ	x,y	2	2	p01 p5	7	1	AVX
VCVT(T)PD2DQ	x,m256	2	3	p01 p23 p5		1	AVX
CVTPI2PS	x,mm	2	2	p0 p1	6	2	
CVTPI2PS	x,m64	1	2	p01 p23		3	
CVT(T)PS2PI	mm,x	2	2	p0 p5	7	1	
CVT(T)PS2PI	mm,m128	2	2	p0 p23		1	
CVTPI2PD	x,mm	2	2	p01 p5	5	1	
CVTPI2PD	x,m64	1	2	p01 p23		0.5	
CVT(T) PD2PI	mm,x	2	2	p01 p5	5	1	
CVT(T) PD2PI	mm,m128	2	3	p01 p23 p5		1	
CVTSI2SS	x,r32	2	2	p01 p5	6	2	
CVTSI2SS	x,r64	3	3	p01 2p5	7	2	
CVTSI2SS	x,m32	1	2	p1 p23		3	
CVT(T)SS2SI	r32,x	2	2	2p01	6	1	
CVT(T)SS2SI	r64,x	3	3	2p01 p5	7	1	
CVT(T)SS2SI	r32,m32	3	3	2p01 p3	'	1	
CVT(1)55251	x,r32/64	2	2	p01 p5	6	2	
CVTSI2SD	x,m32	1	2	p01 p23		2	
0 1 1 1 1 2 2 2 2	7,11102	1	_	ρυιρέυ		_ _	

			OK	ylako			
CVT(T)SD2SI	r32/64,x	2	2	p0 p1	6	1	
CVT(T)SD2SI	r32,m64	3	3	2p01 p23		1	
VCVTPS2PH	x,v,i	2	2	p01 p5	5-7	1	F16C
VCVTPS2PH	m,v,i	3	3	p01 p4 p23		1	F16C
VCVTPH2PS	V,X	2	2	p01 p5	5-7	1	F16C
VCVTPH2PS	v,m	1	2	p01 p23		1	F16C
	•,	•	_	po. p20		•	
Arithmetic							
ADDSS/D PS/D							
SUBSS/D PS/D	x,x / v,v,v	1	1	p01	4	0.5	
ADDSS/D PS/D	, , ,						
SUBSS/D PS/D	x,m / v,v,m	1	2	p01 p23		0.5	
ADDSUBPS/D	x,x / v,v,v	1	1	p01	4	0.5	SSE3
ADDSUBPS/D	x,m / v,v,m	1	2	p01 p23		0.5	SSE3
HADDPS/D							
HSUBPS/D	x,x / v,v,v	3	3	p01 2p5	6	2	SSE3
HADDPS/D							
HSUBPS/D	x,m / v,v,m	4	4	p1 2p5 p23		2	SSE3
MULSS/D PS/D	x,x / v,v,v	1	1	p01	4	0.5	
MULSS/D PS/D	x,m / v,v,m	1	2	p01 p23		0.5	
DIVSS	x,x	1	1	p0	11	3	
DIVPS	x,x	1	1	p0	11	3	
DIVSS DIVPS	x,m	1	2	p0 p23		3-5	
DIVSD	x,x	1	1	p0	13-14	4	
DIVPD	x,x	1	1	p0	13-14	4	
DIVSD DIVPD	x,m	1	2	p0 p23		4	
VDIVPS	y,y,y	1	1	p0	11	5	AVX
VDIVPS	y,y,m256	1	2	p0 p23		5	AVX
VDIVPD	y,y,y	1	1	p0	13-14	8	AVX
VDIVPD	y,y,m256	4	4	p0 p23		8	AVX
RCPSS/PS	V,V	1	1	p0	4	1	
RCPSS/PS	v,m	1	2	p0 p23	-	1	
CMPccSS/D	.,	·	_	P P P = 0		·	
CMPccPS/D	x,x / v,v,v	1	1	p01	4	0.5	
CMPccSS/D				,			
CMPccPS/D	x,m / v,v,m	2	2	p01 p23		0.5	
(U)COMISS/D	x,x	1	1	p0		1	
(U)COMISS/D	x,m32/64	2	2	p0 p23		1	
MAXSS/D PS/D							
MINSS/D PS/D	x,x / v,v,v	1	1	p01	4	0.5	
MAXSS/D PS/D							
MINSS/D PS/D	x,m / v,v,m	1	2	p01 p23		0.5	
				0.04			00544
ROUNDSS/D PS/D	v,v,i	2	2	2p01	8	1	SSE4.1
ROUNDSS/D PS/D	v,m,i	3	3	2p01 p23		1	SSE4.1
DPPS	x,x,i / v,v,v,i	4	4	3p01 p5	13	1.5	SSE4.1
DPPS	x,x,i / v,v,v,i x,m,i / v,v,m,i	6	6	3p01 p3	13	1.5	SSE4.1
DPPD	x,111,1 / v,v,111,1 X,X,İ	3	3	2p01 p5	9	1.5	SSE4.1
DPPD		4	4		9	1	SSE4.1
VFMADD	x,m128,i	4	4	2p01 p23 p5		'	33E4.1
(all FMA instr.)	V,V,V	1	1	p01	4	0.5	FMA
VFMADD	v, v, v	'	'	ρο ι	–	0.5	1141/7
(all FMA instr.)	v,v,m	1	2	p01 p23		0.5	FMA
	, ,		_				
1		I	I	I	I	I	ı

Math							
SQRTSS/PS	x,x	1	1	p0	12	3	
SQRTSS/PS	x,m128	1	2	p0 p23		3	
VSQRTPS	y,y	1	1	p0	12	6	AVX
VSQRTPS	y,m256	4	4	p0 p23		6	AVX
SQRTSD	X,X	1	1	p0	15-16	4-6	
SQRTPD	x,x	1	1	p0	15-16	4-6	
SQRTSD/PD	x,m128	1	2	p0 p23		4-6	
VSQRTPD	y,y	1	1	p0	15-16	9-12	AVX
VSQRTPD	y,m256	4	4	p0 p23		9-12	AVX
RSQRTSS/PS	V,V	1	1	p0	4	1	
RSQRTSS/PS	v,m	1	2	p0 p23		1	
Logic							
AND/ANDN/OR/ XORPS/PD	x,x / v,v,v	1	1	p015	1	0.33	
AND/ANDN/OR/ XORPS/PD	x,m / v,v,m	1	2	p015 p23		0.5	
Other							
VZEROUPPER		4	4	none		1	AVX AVX,
VZEROALL		25	25	p0 p1 p5 p6		12	32 bit AVX,
VZEROALL		34	34	p0 p1 p5 p6		12	64 bit
LDMXCSR	m32	4	4	p0 p5 p6 p23	5	3	0.2.
STMXCSR	m32	3	4	p0 p4 p6 p237	5	2	
FXSAVE	m4096	106			78	78	32 bit mode
FXSAVE	m4096	136			64	64	64 bit mode
FXRSTOR	m4096	105			76	76	32 bit mode
FXRSTOR	m4096	121			77	77	64 bit mode
XSAVE		247			107	107	32 bit mode
XSAVE		304			107	107	64 bit mode
XRSTOR		257			122	122	32 bit mode
XRSTOR		257			122	122	64 bit mode
XSAVEOPT	m	168			74	74	

Intel Skylake-X

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the same

data. Instructions with or without V name prefix behave the same unless otherwise

noted.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm register,

mm/x = mmx or xmm register, y = 256 bit ymm register, z = 512 bit zmm register, xy = xmm or ymm register, y = xmm or ymm, ym, y

operand, m32 = 32-bit memory operand, etc.

μορs fused domain:

The number of µops at the decode, rename, allocate and retire stages in the pipeline.

Fused uops count as one.

μops unfused do-

one unfused do The total number of uses

main: macro-op

The total number of μ ops for all execution ports. Fused μ ops count as two. Fused macro-ops count as one. The instruction has μ op fusion if this number is higher than the number under fused domain. Some operations are not counted here if they do not

go to any execution port or if the counters are inaccurate.

μops each port: The number of μops for each execution port. p0 means a μop to execution port 0.

p01means a μ op that can go to either port 0 or port 1. p0 p1 means two μ ops going

to port 0 and 1, respectively.

Port 0: Integer, f.p. and 256 bit vector ALU, mul, div, branch

Port 1: Integer, f.p. and 256 bit vector ALU (re-routed to port 0 for 512-bit vectors)

Port 2: Load Port 3: Load Port 4: Store

Port 5: Integer and 512 bit vector ALU

Port 6: Integer ALU, branch Port 7: Store address

Latency: This is the delay that the instruction generates in a dependency chain. The numbers

are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Where hyperthreading is enabled, the use of the same execution units in the other thread leads to inferior performance. Denormal numbers, NAN's and infinity do not increase the latency. The time unit used is core clock cy-

cles, not the reference clock cycles given by the time stamp counter.

Reciprocal throughput:

The average number of core clock cycles per instruction for a series of independent

instructions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instructions							
MOV	r,i	1	1	p0156		0.25	
MOV	r8/16,r8/16	1	1	p0156	1	0.25	
MOV	r32/64,r32/64	1	1	p0156	0-1	0.25	may elimi- nate
MOV	r8I,m	1	2	p23 p0156		0.5	
MOV	r8h,m	1	1	p23		0.5	
MOV	r16,m	1	2	p23 p0156		0.5	
MOV	r32/64,m	1	1	p23	2	0.5	all address- ing modes
MOV	m,r	1	2	p237 p4	2	1	
MOV	m,i	1	2	p237 p4		1	
MOVNTI	m,r	2	2	p23 p4	~400	1	

MOVSX MOVZX				Citylai				
MOVSX MOVZX r16,m8		r,r	1	1	p0156	1	0.25	
MOVSX MOVZX r,m		r16 m0	4	2	n22 n0156	2	0.5	
MOVSXD								all other
CMOVcc		1,111	'	!	μ23	2	0.5	combina-
CMOVcc	CMOVcc	r,r	1	1	p06	1	0.5	uons
XCHG								
XCHG						2		
PUSH								implicit lock
PUSH	YLAT		3	1	n23 2n0156	7	1	
PUSH		r						
PUSH		;				3		
PUSHF(D/Q) PUSHF(D/Q) PUSHF(D/Q) PUSHA(D) PUSHA(D) PUSHA(D) PUSHA(D) PUSHA(D) PUSHA(D) POP Stack pointer 3 3 p23 2p0156 3 p23 2p0156 3 p23 2p0156 3 p23 2p0156 3 p23 2p0156 3 p23 2p0156 3 p23 2p0156 p3 p23 2p0156 p3 p23 2p0156 p3 p23 2p0156 p3 p23 2p0156 p3 p23 2p0156 p3 p23 2p0156 p3 p23 2p0156 p3 p23 2p0156 p3 p23 2p0156 p3 p3 p3 p23 2p0156 p3 p3 p3 p3 p3 p3 p3 p		n n		1	1 1			
PUSHF(D/Q) PUSHA(D) PUSHA(D) PUSHA(D) PUSHA(D) PUSHA(D) PUSHA(D) PUSHA(D) POP POP POP Stack pointer 3					1			
PUSHA(D)		Stack pointer			1			
POP POP POP POP POPF(D/Q) r stack pointer m 1 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3	` ′				p1 p4 p237 p00			not 64 hit
POP	, ,	r			n22	2		110t 04 bit
POP POPF(D/Q) m 2 3 2p237 p4 1 20 POPA(D) 18 18 18 8 not 64 bit LAHF SAHF 1 1 p06 1 1 not 64 bit SALC 3 3 3p0156 1 1 not 64 bit LEA r16,m 2 2 p1 p05 2-4 1 LEA r32/64,m 1 1 p15 1 0.5 1 or 2 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 1 1 or 2 components in address LEA r32/64,m 1 1 p1 1 1 rip relative address BSWAP r32 1 1 p1 1 0.5 1 1 1		· .				2		
POPF(D/Q)		·		1	1			
POPA(D)		111		1	2p237 p4		1 -	
LAHF SAHF SALC							1	not 64 hit
SALC r16,m 2 2 p1 p05 2-4 1 not 64 bit LEA r32/64,m 1 1 p15 1 0.5 1 or 2 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 1 1 rip rip relative address LEA r32/64,m 1 1 p15 1 0.5 1 7 1 1 p15 1 0.5 1 1 1 1 p15 1 0.5 1 1 0.5 1 1 0.5 1 0.5 1 0.5 </td <td></td> <td></td> <td></td> <td></td> <td>506</td> <td>1</td> <td></td> <td>110t 04 bit</td>					506	1		110t 04 bit
LEA r16,m 2 2 p1 p05 2-4 1 1 or 2 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 1 rost in address BSWAP r32 1 1 p15 1 0.5 1 rip relative address BSWAP r64 2 2 p06 p15 2 1 1 0.5 <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>not 64 hit</td>								not 64 hit
LEA r32/64,m 1 1 p15 1 0.5 1 or 2 components in address LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 1 rip relative address BSWAP r32 1 1 p15 1 0.5 1 rip relative address BSWAP r64 2 2 p06 p15 2 1 rip relative address BSWAP r64 2 2 p06 p15 2 1 rip relative address BSWAP r64 2 2 p06 p15 2 1 0.5 1 rip relative address BSWAP r64 2 2 p06 p15 2 1 0.5 1 rip relative address BSWAP r64 3 3 2p0156 p23 0.5-1 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5		r16 m				· ·		110t 04 bit
LEA r32/64,m 1 1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 p1 3 1 3 components in address LEA r32/64,m 1 1 p1 p1 1 1 p1 p1 p1 p1 p1 p1 p1 p1 p					1			1 or 2 com
LEA r32/64,m 1 1 p1 p1 1 1 o.5 BSWAP r32 1 1 p15 1 0.5 BSWAP r64 2 2 p06 p15 2 1 MOVBE r16,m16 3 3 2p0156 p23 0.5-1 MOVBE r32,m32 2 2 p15 p23 0.5 MOVBE r64,m64 3 3 2p0156 p23 0.75 MOVBE m16,r16 2 3 p06 p237 p4 1 MOVBE m32,r32 2 3 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 MFENCE 2 p23 p4 6 Arithmetic instructions ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 p23 p4 5 1	LEA	132/04,111	ı	I	ртэ	ı	0.5	ponents in
LEA r32/64,m 1 1 p1 1 rip relative address BSWAP r32 1 1 p15 1 0.5 BSWAP r64 2 2 p06 p15 2 1 MOVBE r16,m16 3 3 2p0156 p23 0.5-1 MOVBE r32,m32 2 2 p15 p23 0.5 MOVBE m16,r16 2 3 p06 p237 p4 1 MOVBE m32,r32 2 3 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 MFENCE 4 4 p23 p4 6 ADD SUB r,m 1	LEA	r32/64,m	1	1	p1	3	1	nents in
BSWAP r64 2 2 p06 p15 2 1 MOVBE r16,m16 3 3 2p0156 p23 0.5-1 MOVBE r32,m32 2 2 p15 p23 0.5 MOVBE r64,m64 3 3 2p0156 p23 0.75 MOVBE m16,r16 2 3 p06 p237 p4 1 MOVBE m32,r32 2 3 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 4 MFENCE 4 4 p23 p4 3 SFENCE 2 2 p23 p4 6 ADD SUB r,r/i 1 1 p0156 1 0.25 ADD SUB m,r/i 2 4 2p015	LEA	r32/64,m	1	1	p1		1	rip relative
BSWAP r64 2 2 p06 p15 2 1 MOVBE r16,m16 3 3 2p0156 p23 0.5-1 MOVBE r32,m32 2 2 p15 p23 0.5 MOVBE r64,m64 3 3 2p0156 p23 0.75 MOVBE m16,r16 2 3 p06 p237 p4 1 MOVBE m32,r32 2 3 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 4 MFENCE 4 4 p23 p4 3 SFENCE 2 2 p23 p4 6 Arithmetic instructions 7,r/i 1 1 p0156 1 0.25 ADD SUB m,r/i 2 4	BSWAP	r32	1	1	p15	1	0.5	
MOVBE r16,m16 3 3 2p0156 p23 0.5-1 MOVBE r32,m32 2 2 p15 p23 0.5 MOVBE r64,m64 3 3 2p0156 p23 0.75 MOVBE m16,r16 2 3 p06 p237 p4 1 MOVBE m32,r32 2 3 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 4 MFENCE 4 4 p23 p4 3 SFENCE 2 2 p23 p4 6 Arithmetic instructions 7,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p		r64	2	2	1	2		
MOVBE r64,m64 3 3 2p0156 p23 0.75 MOVBE m16,r16 2 3 p06 p237 p4 1 MOVBE m32,r32 2 3 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 4 MFENCE 4 4 p23 p4 33 SFENCE 2 2 p23 p4 6 Arithmetic instructions 2 2 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	MOVBE	r16,m16	3	3	2p0156 p23		0.5-1	
MOVBE m16,r16 2 3 p06 p237 p4 1 MOVBE m32,r32 2 3 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 4 MFENCE 4 4 p23 p4 33 SFENCE 2 2 p23 p4 6 Arithmetic instructions 2 2 p0156 1 0.25 ADD SUB r,r/i 1 1 p0156 1 0.25 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	MOVBE	r32,m32	2		p15 p23		0.5	
MOVBE m32,r32 2 3 p15 p237 p4 1 MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 MFENCE 4 4 p23 p4 33 SFENCE 2 2 p23 p4 6 Arithmetic instructions 2 2 p23 p4 6 ADD SUB r,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	MOVBE	r64,m64	3	3	2p0156 p23		0.75	
MOVBE m64,r64 3 4 p06 p15 p237 p4 1 PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 MFENCE 4 4 p23 p4 33 SFENCE 2 2 p23 p4 6 Arithmetic instructions 7,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	MOVBE	m16,r16	2		p06 p237 p4		1	
PREFETCHNTA/0/1/2 m 1 1 p23 0.5 PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 4 MFENCE 4 4 p23 p4 33 SFENCE 2 2 p23 p4 6 Arithmetic instructions 7,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	MOVBE	m32,r32	2	3			1	
PREFETCHW m 1 1 p23 1 LFENCE 2 none counted 4 MFENCE 4 4 p23 p4 33 SFENCE 2 2 p23 p4 6 Arithmetic instructions 7,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	MOVBE	m64,r64	3	4	p06 p15 p237 p4		1	
LFENCE 2 none counted 4 MFENCE 4 4 p23 p4 33 SFENCE 2 2 p23 p4 6 Arithmetic instructions ADD SUB r,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	PREFETCHNTA/0/1/2	m	1	1	p23		0.5	
MFENCE SFENCE 4 4 p23 p4 p23 p4 33 6 Arithmetic instructions 7,r/i 1 1 p0156 1 0.25 p0156 p23 p0156 p23 p0156 2p237 p4 1 0.25 p0156 2p237 p4 0.5 p0156 2p237 p4	PREFETCHW	m	1	1	p23		1	
SFENCE 2 2 p23 p4 6 Arithmetic instructions r,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	LFENCE				none counted		4	
Arithmetic instructions ADD SUB r,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	MFENCE			1	p23 p4		1	
tions r,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1	SFENCE		2	2	p23 p4		6	
ADD SUB r,r/i 1 1 p0156 1 0.25 ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1								
ADD SUB r,m 1 2 p0156 p23 0.5 ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1		r r/i	1	1	p0156	1	0.25	
ADD SUB m,r/i 2 4 2p0156 2p237 p4 5 1								
						5		

			,				
ADC SBB	r,m	1	2	p06 p23		1	
ADC SBB	m,r/i	4	6	3p0156 2p237 p4	5	2	
ADCX ADOX	r,r	1	1	p06	1	1	
CMP	r,r/i	1	1	p0156	1	0.25	
CMP	m,r/i	1	2	p0156 p23	1	0.5	
INC DEC NEG NOT	r	1	1	p0156	1	0.25	
INC DEC NOT	m	3	4	p0156 2p237 p4	5	1	
NEG	m	2	4	p0156 2p237 p4	5	1	
AAA		2	2	p1 p56	4	-	not 64 bit
AAS		2	2	p1 p056	4		not 64 bit
DAA DAS		3	3	p1 2p056	4		not 64 bit
AAD		3	3	p1 2p056	4		not 64 bit
AAM		11	11	p0 p1 p5 p6	23	7	not 64 bit
MUL IMUL	r8	1	1	p1	3	1	
MUL IMUL	r16	4	4	p1 p0156	4	2	
MUL IMUL	r32	3	3	p1 p0156	4	1	
MUL IMUL	r64	2	2	p1 p6	3	1	
MUL IMUL	m8	1	2	p1 p23	0	1	
MUL IMUL	m16	4	5	p1 3p0156 p23		2	
MUL IMUL	m32	3	4	p1 2p0156 p23		2	
MUL IMUL	m64	2	3			1	
IMUL		1	1	p1 p6 p23 p1	3	1	
IMUL	r,r	1	2	1	J	1	
IMUL	r,m		2	p1 p23	4	1	
	r16,r16,i	2	1	p1 p0156	4 3	1 1	
IMUL	r32,r32,i	1		p1	3	1 1	
IMUL	r64,r64,i			p1	3	1	
IMUL	r16,m16,i	2	3	p1 p0156 p23		1	
IMUL	r32,m32,i	1	2	p1 p23			
IMUL	r64,m64,i	1	2	p1 p23	4	1	
MULX	r32,r32,r32	3	3	p1 2p056	4	1	
MULX	r32,r32,m32	3	4	p1 2p056 p23		1	
MULX	r64,r64,r64	2	2	p1 p5	4	1	
MULX	r64,r64,m64	2	3	p1 p6 p23	00	1	
DIV	r8	10	10	p0 p1 p5 p6	23	6	
DIV	r16	10	10	p0 p1 p5 p6	23	6	
DIV	r32	10	10	p0 p1 p5 p6	26	6	
DIV	r64	36	36	p0 p1 p5 p6	35-88	21-83	
IDIV	r8	11	11	p0 p1 p5 p6	24	6	
IDIV	r16	10	10	p0 p1 p5 p6	23	6	
IDIV	r32	10	10	p0 p1 p5 p6	26	6	
IDIV	r64	57	57	p0 p1 p5 p6	42-95	24-90	
CBW		1	1	p0156	1	0.5	
CWDE		1	1	p0156	1	1	
CDQE		1	1	p0156	1	0.5	
CWD		2	2	p0156	1	1	
CDQ		1	1	p06	1	1	
CQO		1	1	p06	1	0.5	
POPCNT	r,r	1	1	p1	3	1	
POPCNT	r,m	1	2	p1 p23		1	
CRC32	r,r	1	1	p1	3	1	
CRC32	r,m	1	2	p1 p23		1	
Logic instructions							

AND OR XOR	1	1					1	
AND OR XOR TEST (r,t/i 1 1 1 p0156 p23 FEST (r,t/i 1 1 1 p0156 p23 FEST (r,t/i 1 1 1 p0156 p23 FEST (r,t/i 1 1 1 p0156 p23 FEST SHR SHL SAR (r,t) SHR SHR SHR SHR (r,t) SHR SHR SHR SHR (r,t) SHR SHR SHR SHR (r,t) SHR SHR SHR SHR (r,t) SHR SHR SHR SHR (r,t) SHR SH SHR SHR (r,t) SHR SH SHR SHR (r,t) SHR SH SHR SHR (r,t) SHR SH SHR SHR (r,t) SHR SH SHR (r,	AND OR XOR	r,r/i	1			1		
TEST							0.5	
TEST SHR SHL SAR SHL SAR SHR SHL SAR SHR SHL SAR SHR SHL SAR SHR SHL SAR SHR SHL SHR SAR SHL SHR SAR SHR SHR SAR SHL SAR SHR SHR SHR SAR SHL SAR SHR SHR SHR SHR SHR SHR SHR SHR SHR SHR			2	4		5	-	
SHR SHL SAR SHR SHR SHR SHR SHR SHR SHR SHR SHR SHR		r,r/i	1			1		
SHR SHL SAR		m,r/i	1	2		1		
SHR SHL SAR SHR SHR SHR SHR SHR SHL SHR SHR SHR SHR SHR SHR SHL SHR SHR SHR SHR SHR SHR SHR SHR SHR SHR	SHR SHL SAR	r,i	1	1	p06	1	0.5	
SHR SHL SAR ROR ROL F.1 2 2 2 2 2 2 2 2 2	SHR SHL SAR	m,i	3	4	2p06 p237 p4			
ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR RO	SHR SHL SAR	r,cl	3	3	3p06	2	2	
ROR ROL ROR ROR ROR ROL ROR ROL ROR ROL ROR ROL ROR ROL ROR ROL ROR ROL ROR ROR ROR ROL ROR ROL ROR ROR ROR RO	SHR SHL SAR	m,cl	5	6	3p06 2p23 p4		4	
ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR RO	ROR ROL	r,1	2	2	2p06	1	1	short form
ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROL ROR ROR ROR ROR ROR ROL ROR ROR ROR ROR ROR ROR ROR ROR ROR ROR ROR	ROR ROL	r,i	1	1	p06	1	0.5	
ROR ROL RCR RCL RCR RCL RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL RCR RCL R,1 RCR RCL R,2 RCR RCL R,3 RCR RCL R,4 RCR RCL R,6 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL R,1 RCR RCL RCR RCL R,1 R,1 R,1 R,1 R,1 R,1 R,1 R,1 R,1 R,1	ROR ROL	m,i	4	5	2p06 2p237 p4		2	
ROR ROL RCR RCR RCR RCL RCR RCR RCR RCL RCR RCL RCR RCR RCR RCL RCR RCR RCR RCL RCR RCR RCR RCL RCR RC	ROR ROL	r,cl	3	3	3p06	2	2	
RCR RCL RCR RCR RCR RCL RCR RCR RCR RCL RCR RC	ROR ROL		5	6				
RCR RCL RCR RCR RCR RCR RCL RCR RCL RCR RCL RCR RCR RCR RCR RCL RCR RCR RCR RCR RCL RCR RCR RCR RCR RC	RCR RCL		3	3		2	2	
RCR RCL r,i 8 8 p0156 6 6 RCR RCL m,i 11 11 6 6 RCR RCL r,cl 8 8 p0156 6 6 RCR RCL m,cl 11 11 11 1 6 6 SHRD SHLD r,r,i 1 1 p1 3 1 3 1 SHRD SHLD m,r,cl 4 4 p0156 3 2 2 SHRD SHLD m,r,cl 5 7 4 4 4 p0156 4 2 2 SHRD SHLD 4,r,r,cl 4 4 p0156 4 2 2 SHRD SHR SHRX SARX r,r,r,r,r 1 1 p06 1 0.5 5 N 4 SHLX SHRX SARX r,r,r,r 1 1 p06 1 0.5 N N N 0.5 N N N N N N N	RCR RCL		4	6				
RCR RCL m,i 11 11 11 11 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 7 6 8 8 8 p0156 6 6 6 6 6 7 7 7 7 7 7 7 7 7 7 7 7 7 8 7 7 9	1		8		p0156	6		
RCR RCL r,cl 8 8 p0156 6 6 RCR RCL m,cl 11 11 11 5 6 6 SHRD SHLD r,r,cl 1 1 1 p1 3 1 SHRD SHLD m,r,cl 4 4 p0156 3 2 SHRD T,r,cl 4 4 p0156 4 2 SHRD SHLD m,r,cl 5 7 5 7 SHRD SHLD m,r,cl 5 7 7 4 4 SHRD SHRD SHRX SARX r,r,r,r 1 1 p06 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5 1 0.5								
RCR RCL					p0156	6		
SHRD SHLD r,r,i 1 1 p1 3 1 SHRD SHLD m,r,i 3 5 2 SHLD r,r,cl 4 4 p0156 3 2 SHRD r,r,cl 4 4 p0156 4 2 SHRD SHLD m,r,cl 5 7 4 4 p0156 4 2 SHRD SHLD m,r,cl 5 7 4 4 p0156 4 2 SHRD SHLD m,r,cl 5 7 4 4 p0156 4 2 SHRD SHLD m,r,cl 4 4 p0156 4 2 2 SHRD SHLD m,r,cl 5 7 4 4 p0156 4 2 2 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5	1							
SHRD SHLD m,r,i 3 5 p0156 3 2 SHRD r,r,cl 4 4 4 p0156 4 2 SHRD SHLD m,r,cl 5 7 4 4 7 4 4 8 1 2 2 906 p23 0.5 8 1 0.5 8					n1	3		
SHLD r,r,cl 4 4 4 p0156 3 2 SHRD r,r,cl 4 4 p0156 4 2 SHRD SHLD m,r,cl 5 7 4 SHLX SHRX SARX r,r,r 1 1 p06 1 0.5 SHLX SHRX SARX r,m,r 2 2 p06 p23 0.5 0.5 RORX r,m,i 1 1 p06 1 0.5 0.5 RORX r,m,i 1 1 p06 1 0.5 0.5 BT r,r/i 1 1 p06 1 0.5 0.5 BT m,i 10 10 5 0.5 0.5 0.5 BTR BTS BTC m,r 10 11 p06 1 0.5 0.5 BTR BTS BTC m,r 10 11 p1 3 1 0.5 0.5 0.5 0.5 0.5 0.5					μ.	Ū		
SHRD r,r,cl 4 4 4 p0156 4 2 SHRD SHLD m,r,cl 5 7 p06 1 0.5 SHLX SHRX SARX r,r,r 1 1 p06 p23 0.5 SHLX SHRX SARX r,r,ri 1 1 p06 p23 0.5 RORX r,m,i 2 2 p06 p23 0.5 BT m,r 10 10 5 BT m,r 10 10 5 BT m,i 2 2 p06 p23 0.5 BT BTS BTC m,ri 1 1 p06 1 0.5 BTR BTS BTC m,ri 10 11 p06 1 0.5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 1 BF BSR r,r 1 1 p1 p23 1 1 SETcc r 1 1 p06 p237 p4 1 1					p0156	3		
SHRD SHLD m,r,cl 5 7 no6 1 4 SHLX SHRX SARX r,r,r 1 1 p06 1 0.5 SHLX SHRX SARX r,m,r 2 2 p06 p23 0.5 RORX r,r,ri 1 1 p06 1 0.5 RORX r,r,mi 2 2 p06 p23 0.5 0.5 BT m,r 10 10 5 0.5 0.5 0.5 BTR BTS BTC m,r 10 10 5 0.5 <								
SHLX SHRX SARX r,r,r 1 1 p06 1 0.5 SHLX SHRX SARX r,m,r 2 2 p06 p23 0.5 RORX r,r,i 1 1 p06 p23 0.5 RORX r,mi 2 2 p06 p23 0.5 BT r,r/i 1 1 p06 1 0.5 BT m,r 10 10 5 10.5					po.00	•		
SHLX SHRX SARX r,m,r 2 2 p06 p23 0.5 RORX r,r,i 1 1 p06 1 0.5 RORX r,m,i 2 2 p06 p23 0.5 BT r,r/i 1 1 p06 1 0.5 BT m,r 10 10 5 5 BTR BTS BTC m,i 2 2 p06 p23 0.5 BTR BTS BTC m,r 10 11 p06 1 0.5 BTR BTS BTC m,r 10 11 p06 1 0.5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 1 BF BSR r,r 1 1 p1 3 1 1 BF BSR r,r 1 1 p1 p23 1 1 0.5 1 SETcc r 1 1 p06 p23 p24 1 1 0.25 0.25 0.25					p06	1		
RORX r,r,i 1 1 p06 1 0.5 RORX r,m,i 2 2 p06 p23 0.5 BT r,r/i 1 1 p06 1 0.5 BT m,r 10 10 5 5 BT m,i 2 2 p06 p23 0.5 BTR BTS BTC m,i 1 1 p06 1 0.5 BTR BTS BTC m,r 10 11 5 5 5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 1 0.5 5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 1 1 8 1 1 1 0.5 1 1 1 1 1 0.5 1 1 1 1 1 1 1 0.5 1 1 0.5 1 1 0.5 1 1 0.5 1 <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>								
RORX r,m,i 2 2 p06 p23 0.5 BT r,r/i 1 1 p06 1 0.5 BT m,r 10 10 5 BT m,r 10 11 p06 1 0.5 BTR BTS BTC m,r/i 1 1 p06 1 0.5 BTR BTS BTC m,r 10 11 5 5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 1 BSF BSR r,r 1 1 p1 3 1 1 8 1 <td>1</td> <td></td> <td></td> <td></td> <td></td> <td>1</td> <td></td> <td></td>	1					1		
BT r,r/i 1 1 p06 1 0.5 BT m,r 10 10 10 5 BT m,r 10 10 p06 p23 0.5 BTR BTS BTC m,r/i 1 1 p06 1 0.5 BTR BTS BTC m,r 10 11 5 5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 1 BSF BSR r,r 1 1 p1 3 1 1 BSF BSR r,rm 1 2 p1 p23 1 1 SETcc r 1 1 p06 1 0.5 SETcc r 1 1 p06 1 0.5 SETcc m 2 3 p06 p237 p4 1 1 CLC 1 0 none 0.25 0.25 O.25 O.25<					· ·	•		
BT m,r 10 10 p06 p23 5 BTR BTS BTC r,r/i 1 1 p06 p4 p23 0.5 BTR BTS BTC m,r 10 11 5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 BSF BSR r,r 1 1 p1 3 1 BSF BSR r,m 1 2 p1 p23 1 1 SETcc r 1 1 p06 1 0.5 1 0.5 SETcc m 2 3 p06 p237 p4 1 1 0.5 SETcc m 2 3 p06 p237 p4 1 1 0.5 SETcc 1 1 0 none 0.25						1		
BT m,i 2 2 p06 p23 0.5 BTR BTS BTC r,r/i 1 1 p06 1 0.5 BTR BTS BTC m,r 10 11 5 5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 BSF BSR r,r 1 1 p1 3 1 BSF BSR r,m 1 2 p1 p23 1 1 SETcc r 1 1 p06 1 0.5 1 SETcc r 1 1 p06 1 0.5 1 SETcc m 2 3 p06 p237 p4 1 1 0.5 1 SETcc m 2 3 p06 p237 p4 1 1 0.25 STC 1 1 p0156 0.25 0.25 0.25 0.25 CMC 1 1 p0156 1 1 1 0.					Poo	•		
BTR BTS BTC r,r/i 1 1 p06 1 0.5 BTR BTS BTC m,r 10 11 5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 BSF BSR r,r 1 1 p1 3 1 BSF BSR r,m 1 2 p1 p23 1 1 SETcc r 1 1 p06 1 0.5 1 SETcc m 2 3 p06 p237 p4 1 1 0.5 1 0.5 0.25					n06 n23			
BTR BTS BTC m,r 10 11 p06 p4 p23 5 BTR BTS BTC m,i 3 4 p06 p4 p23 1 BSF BSR r,r 1 1 p1 3 1 BSF BSR r,m 1 2 p1 p23 1 1 SETcc r 1 1 p06 1 0.5 1 SETcc m 2 3 p06 p237 p4 1 1 0.25 1 1 0.25						1		
BTR BTS BTC m,i 3 4 p06 p4 p23 1 BSF BSR r,r 1 1 p1 3 1 BSF BSR r,m 1 2 p1 p23 1 1 SETcc r 1 1 p06 1 0.5 SETcc m 2 3 p06 p237 p4 1 1 CLC 1 0 none 0.25 0.25 STC 1 1 p0156 0.25 CMC 1 1 p0156 1 CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 p23 1 LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 p23 1 TZCNT r,r 1 1 p1 p23 1 ANDN r,r,r,r 1 1 p15 1 0.5<					Poo	•		
BSF BSR r,r 1 1 p1 3 1 BSF BSR r,m 1 2 p1 p23 1 SETcc r 1 1 p06 1 0.5 SETcc m 2 3 p06 p237 p4 1 1 CLC 1 0 none 0.25 0.25 STC 1 1 p0156 0.25 CMC 1 1 p0156 1 CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 p23 1 LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 p23 1 ANDN r,r,r,r 1 1 p15 1 0.5 ANDN r,r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 1 0.5 </td <td></td> <td></td> <td></td> <td></td> <td>n06 n4 n23</td> <td></td> <td></td> <td></td>					n06 n4 n23			
BSF BSR r,m 1 2 p1 p23 1 SETcc r 1 1 p06 1 0.5 SETcc m 2 3 p06 p237 p4 1 1 CLC 1 0 none 0.25 0.25 STC 1 1 p0156 0.25 CMC 1 1 p0156 1 CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 3 1 LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 3 1 TZCNT r,m 1 2 p1 p23 1 1 ANDN r,r,r 1 1 p15 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSI BLSMSK r r r <	1					3		
SETcc r 1 1 p06 1 0.5 SETcc m 2 3 p06 p237 p4 1 CLC 1 0 none 0.25 STC 1 1 p0156 0.25 CMC 1 1 p0156 1 CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 3 1 LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 p23 1 TZCNT r,m 1 2 p1 p23 1 ANDN r,r,r 1 1 p15 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5						J		
SETcc m 2 3 p06 p237 p4 1 CLC 1 0 none 0.25 STC 1 1 p0156 0.25 CMC 1 1 p0156 1 CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 3 1 LZCNT r,m 1 2 p1 p23 1 1 TZCNT r,r 1 1 p1 p23 1 1 ANDN r,r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5						1		
CLC 1 0 none 0.25 STC 1 1 p0156 0.25 CMC 1 1 p0156 1 CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 3 1 LZCNT r,m 1 2 p1 p23 1 1 TZCNT r,m 1 2 p1 p23 1 1 ANDN r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR 0.5 0.5 0.5 0.5 0.5 0.5					1	ı		
STC 1 1 p0156 0.25 CMC 1 1 p0156 1 CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 3 1 LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 3 1 TZCNT r,m 1 2 p1 p23 1 ANDN r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR 0.5 0.5 0.5 0.5 0.5 0.5		111					-	
CMC 1 1 p0156 1 CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 3 1 LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 3 1 TZCNT r,m 1 2 p1 p23 1 ANDN r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR 0.5 0.5 0.5 0.5 0.5 0.5			1					
CLD STD 3 3 p15 p6 4 LZCNT r,r 1 1 p1 3 1 LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 3 1 TZCNT r,m 1 2 p1 p23 1 ANDN r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR	1		1			1	0.23	
LZCNT r,r 1 1 p1 3 1 LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 3 1 TZCNT r,m 1 2 p1 p23 1 ANDN r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR 0.5 0.5 0.5 0.5 0.5 0.5	1				1	ı	1	
LZCNT r,m 1 2 p1 p23 1 TZCNT r,r 1 1 p1 p23 1 TZCNT r,m 1 2 p1 p23 1 ANDN r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR -	1	rr				2		
TZCNT r,r 1 1 p1 3 1 TZCNT r,m 1 2 p1 p23 1 ANDN r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR - </td <td></td> <td></td> <td></td> <td></td> <td></td> <td>3</td> <td></td> <td></td>						3		
TZCNT r,m 1 2 p1 p23 1 ANDN r,r,r 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR -	1					2		
ANDN r,r,r 1 1 1 p15 1 0.5 ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR					-	3		
ANDN r,r,m 1 2 p15 p23 1 0.5 BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR			1			1		
BLSI BLSMSK r,r 1 1 p15 1 0.5 BLSR 1 0.5					·			
BLSR								
		1,1	!	1	pio	I	0.5	
IDEOLDEONIOIS I IIII I I L C I DIODEO I UIU		r m	1	2	n15 n23		0.5	
BLSR		1,111	'	_	ρ10 μ20		0.0	

			,				
BEXTR	r,r,r	2	2	2p0156	2	0.5	
BEXTR	r,m,r	3	3	2p0156 p23	_	1	
BZHI		1	1	p15	1	0.5	
	r,r,r				ļ		
BZHI	r,m,r	1	2	p15 p23	_	0.5	
PDEP	r,r,r	1	1	p1	3	1	
PDEP	r,r,m	1	2	p1 p23		1	
PEXT	r,r,r	1 1	1	p1	3	1	
PEXT	r,r,m	1	2	p1 p23		1	
Control transfer inst	 tructions						
JMP	short/near	1 1	1	p6		1-2	
JMP	r	1 1	1	p6		2	
JMP			2			2	
	m	1 1		p23 p6		1	
Conditional jump	short/near	1	1	p6		1-2	predicted taken
Conditional jump	short/near	1	1	p06		0.5-1	predicted not taken
Fused arithmetic and branch		1	1	p6		1-2	predicted taken
Fused arithmetic and		1	1	p06		0.5-1	predicted
branch						3.0 .	not taken
J(E/R)CXZ	short	2	2	p0156 p6		0.5-2	
LOOP	short	7	7	ρο του ρο		4	
		10	, 10			1	
LOOP(N)E	short					6	
CALL	near	2	3	p237 p4 p6		3	
CALL	r	2	3	p237 p4 p6		3	
CALL	m	3	4	2p237 p4 p6		4	
RET		1 1	2	p237 p6		1	
RET	i		2			1	
BOUND	r,m	15	15			8	not 64 bit
INTO	,	5	5			6	not 64 bit
String instructions							
LODSB/W		3	3	2p0156 p23		1	
LODSD/Q		2	2	p0156 p23		1	
REP LODS		5n+12	_	p0100 p20		~2n	
STOS			3	p23 p0156 p4		1	
		3	3	p23 p0136 p4			
REP STOS		<2n				~0.5n	worst case
REP STOS		2.6/32B				1/32B	best case aligned by 32
MOVS		5	5	2p23 p4 2p0156		4	ungricu by 62
REP MOVS		~2n		2p20 p4 2p0 100		< 1n	worst soss
							worst case best case
REP MOVS		4/32B				1/32B	aligned by 32
SCAS		3	3	p23 2p0156		1	
REP SCAS		≥8n				≥n	
CMPS		5	5	2p23 3p0156		4	
REP CMPS		≥8n				≥2n	
Synchronization ins	tructions						
XADD	m,r	4	5			5	
LOCK XADD	m,r	9	9			18	
LOCK ADD	,	8	8			18	
	m,r					1	
CMPXCHG	m,r	5	6			5	
LOCK CMPXCHG	m,r	10	10			18	1

CMPXCHG8B	m,r	16	16			11	
LOCK CMPXCHG8B	m,r	20	20			19	
CMPXCHG16B	m,r	23	23			16	
LOCK CMPXCHG16B	m,r	25	25			26	
Other							
NOP		1	0	none		0.25	
Long NOP		1	0	none		0.25	
PAUSE		4	4	p6		141	
ENTER	a,0	12	12			8	
ENTER	a,b	~14+7b	~45+7b		~87+2b		
LEAVE		3	3	2p0156 p23		4	
XGETBV		15	15			7	
CPUID		27-118				100-250	
RDTSC		20	20			25	
RDTSCP		22	22			32	
RDPMC		33	33			16	
RDRAND	r	16	16	p23 15p0156		~200	
RDSEED	r	16	16	p23 15p0156		~200	

Floating point x87 instructions

l loating point xo						Recipro-	
		µops	µops			cal	
Instruction	Operands	fused domain	unfused domain	μops each port	Latonov	through put	Comments
Move instructions	Operanus	uomam	uomam	pops each port	Latericy	put	Comments
FLD	r	1	1	p05	1	0.5	
FLD	m32/64	1	1	p23	3	0.5	
FLD	m80	4	4	2p01 2p23	4	2	
FBLD	m80	43	43	2001 2020	46	22	
FST(P)	r	1	1	p05	1	0.5	
FST(P)	m32/m64	1	2	p4 p237	3	1	
FSTP	m80	7	7	3p0156 2p23 2p4	4	5	
FBSTP	m80	244	226	оролоо 2р2о 2р г	264	266	
FXCH	r	2	0	none	0	0.5	
FILD	m	1	2	p05 p23	5	1	
FIST(P)	m	3	3	p5 p23 p4	7	1	
FISTTP	m	3	3	p1 p23 p4	7	2	
FLDZ		1	1	p05		1	
FLD1		2	2	2p05		2	
FLDPI FLDL2E etc.		2	2	2p05		2	
FCMOVcc	r	4	4	p0 p1 p56	3	2	
FNSTSW	AX	2	2	p0 p0156	6	1	
FNSTSW	m16	2	3	p0 p4 p237	6	1	
FLDCW	m16	3	3	p01 p23 p6	7	2	
FNSTCW	m16	2	3	p237 p4 p6	6	1	
FINCSTP FDECSTP		1	1	p05	0	0.5	
FFREE(P)	r	1	1	p05		0.5	
FNSAVE	m	133	133		176	176	
FRSTOR	m	89	89		175	175	
Arithmetic instructions							

			•				
FADD(P) FSUB(R)(P)	r	1	1	p5	3	1	
FADD(P)		•	•	μο		•	
FSUB(R)(P)	m	2	3	p5 p23		1	
FMUL(P)	r	1	1	p0	5	1	
FMUL(P)	m	2	3	p0 p23		1	
FDIV(R)(P)	r	1	1	p0	14-16	4-5	
FDIV(R)(P)	m	1	2	p0 p23		4-5	
FABS		1	1	p0	1	1	
FCHS		1	1	р0	1	1	
FCOM(P) FUCOM	r	1	1	p5	3	1	
FCOM(P) FUCOM	m	1	2	p5 p23		1	
FCOMPP FUCOMPP		2	2	p0 p5		1	
FCOMI(P)				' '			
FUCOMI(P)	r	3	3	p5	4	1	
FIADD FISUB(R)	m	3	4	2p5 p23		2	
FIMUL	m	3	4	p0 p5 p23		1	
FIDIV(R)	m	2	3	p0 p5 p23			
FICOM(P)	m	2	3	2p5 p23		2	
FTST		1	1	p5	3	1	
FXAM		2	2	2p5	6	2	
FPREM		31	31		26-30	17	
FPREM1		31	31		30-57	17	
FRNDINT		17	17		21	11	
Math							
FSCALE		27	27		11	11	
FXTRACT		17	17		11	11	
FSQRT		1	1	p0	14-21	4-7	
FSIN		53-105			50-120		
FCOS		53-105			50-130		
FSINCOS		55-120			55-150		
F2XM1		16-90			65-80		
FYL2X		40-100			103		
FYL2XP1		56			77		
FPTAN		40-112			140-160		
FPATAN		30-160			100-160		
Other							
FNOP		1	1	p05		0.5	
WAIT		2	2	p05		1	
FNCLEX		5	5	p156		22	
FNINIT		18	18			78	

Integer vector instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instructions							
MOVD	r32/64,mm/x	1	1	p0	2	1	
MOVD	m32/64,mm/x	1	2	p237 p4	3	1	
MOVD	mm/x,r32/64	1	1	p5	2	1	

			,				
MOVD	mm/x,m32/64	1	1	p23	2	0.5	
MOVQ	r64,mm/x	1	1	p0	2	1	
MOVQ	mm/x,r64	1	1	p5	2	1	
MOVQ	mm,mm	1		p05	1	0.5	
MOVQ	X,X	1		p015	1	0.33	
MOVQ	mm/x,m64	1	1	p23	2	0.5	
MOVQ	m64, mm/x	1	2	p237 p4	3	1	
WOVQ	11104, 11111/7	'	_	p201 p4		'	may
MOVDQA/U	x,x	1	1	p015	0-1	0.25	eliminate
MOVDQA/U	x, m128	1	1	p23	2	0.5	
MOVDQA/U	m128, x	1	2	p237 p4	3	1	
ovbarro	20, X	•	_	p201 p1			may
VMOVDQA/U	y,y	1	1	p015	0-1	0.25	eliminate
	, , ,			•			may
VMOVDQA/U	Z,Z	1	1	p05	0-1	0.33	eliminate
VMOVDQA32/64	xy{k},xy	1	1	p015	1	0.33	
VMOVDQA32/64	z{k},z	1	1	p05	1	0.5	
VMOVDQA/U/32/64	yz,m	1	1	p23	3	0.5	
VMOVDQA/U	,			'			
/8/16/32/64	v{k},m	1	2	p05 p23	4	0.5	
VMOVDQA/U/32/64	m,v	1	2	p237 p4	3	1	
VMOVDQA/U							
/8/16/32/64	m{k},v	1	2		14	1	
LDDQU	x, m128	1	1	p23	3	0.5	
MOVDQ2Q	mm, x	2	2	p0 p5	2	1	
MOVQ2DQ	x,mm	2	2	p0 p15	2	1	
MOVNTQ	m64,mm	1	2	p237 p4	~600	1	
MOVNTDQ	m128,x	1	2	p237 p4	~700	1	
VMOVNTDQ	m256,y	1	2	p237 p4	~700	1	
MOVNTDQA	v,m	2	2	p23 p015	3	0.5	
PACKSSWB/DW	,	_	_	p20 p0 10		0.0	
PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW	,			'			
PACKUSWB	mm,m64	3	3	p23 2p5		2	
PACKSSWB/DW							
PACKUSWB	x,x / v,v,v	1	1	p5	1	1	
PACKSSWB/DW							
PACKUSWB	x,m / y,y,m	1	2	p23 p5		1	
PACKUSDW	x,x / v,v,v	1	1	p5	1	1	
PACKUSDW	x,m / v,v,m	1	2	p23 p5		1	
PUNPCKH/L							
BW/WD/DQ	v,v / v,v,v	1	1	p5	1	1	
PUNPCKH/L			_				
BW/WD/DQ	v,m / v,v,m	1	2	p23 p5		1	
PUNPCKH/L	,	_		_			
QDQ	x,x / y,y,y	1	1	p5	1	1	
PUNPCKH/L		4	_	-00 -F		_	
QDQ	x,m / y,y,m	1	2	p23 p5		1	
PMOVSX/ZX BW BD BQ DW DQ	x,x	1	1	p5	1	1	
PMOVSX/ZX BW BD	λ,λ	•	'	Po	'	'	
BQ DW DQ	x,m	1	2	p23 p5		1	
VPMOVSX/ZX BW BD	,						
BQ DW DQ	y,x	1	1	p5	3	1	
VPMOVSX/ZX BW BD				_			
BQ DW DQ	z,xy	1	1	p5	3	1	

Image: Continue Image: Con				,			
DWX/DW/DW/DW/DW/DW/DW/DW/DW/DW/DW/DW/DW/DW/		y,m	2	2	p5 p23		1
WPMOVSWB/DB/QB DW/OW/OW/OD PMOVUSWB/DB/QB DW/OW/OW/OD PSHUFB mm,mm 3 3 p0 p5 3 1 PSHUFB mm,mm 3 3 p0 p5 3 1 PSHUFB mm,mm 1 1 p5 1 1 PSHUFB py,m / v,v,m 2 2 p23 p5 1 1 PSHUFW mm,mm4,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 1 1 PSHUFD v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,i 1 1 p5 3 1 PSHUFB/HW v,v,m,i 2 2 p23 p5 1 PSHUFB/HW v,v,m,i 2 2 p23 p5 1 PSHUFB/HW v,v,m,i 2 2 p219 p5 3 1 PSHUFB/HW v,v,v,v 2 2 p219 p5 3 1 PSHUFB/HW v,v,v,v 2 2 p2015 p23 2 PSHUFB/HW v,v,v,v 2 2 p015 p23 2 PSHUFB/HW v,v,v,v 1 1 p5 1 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p015 1 0.33 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW v,v,v,v 1 1 p5 3 1 PSHUFB/HW							
DWICWYOLD DWICWYOLD DWICWYOLD DWICWYOLD DWICWYOLD DWICWYOLD DWICWYOLD DWICWYOLD DWICKYOLD DWIC							
DWM/WW/QD	DW/QW/QD						
PSHUFB			2	2	2n5	4	2
PSHUFB				1			
PSHUFB PSHUFW PSHUFW PSHUFW PSHUFW PSHUFD PSHUFD PSHUFD PSHUFD PSHUFD PSHUFD PSHUFL/HW							
PSHUFW			· ·		· ·	!	
PSHUFW PSHUFD PSHUFD PSHUFD PSHUFD PSHUFD PSHUFD PSHUFD PSHUFL/HW PSHUFL/HW PSHUFL/HW PSHUFI/HW						1	
PSHUFD			l -		· ·		
PSHUFD v,m,i 1-2 2 p3 p5 1 1 PSHUFL/HW v,v,i 1 1 p5 1 1 PSHUFL/HW v,m,i 2 2 p23 p5 1 1 VSHUFI32X4 v,v,v,i 1 1 p5 3 1 VSHUFI64X2 v,v,v,i 1 1 p5 3 1 PALIGNR v,v,i/v,v,v,i 1 1 p5 3 1 VALIGND/Q v,v,v,i 1 1 p5 3 1 VALIGND/Q v,v,v,i 2 2 p23 p5 1 1 VALIGND/Q v,v,m,i 2 2 p23 p5 1 1 VALIGND/Q v,v,m,i 2 2 p015 p23 1 1 VPBLENDVB x,x,xmm0 1 1 p015 p23 1 1 VPBLENDWB x,v,v,v 2 2 2p015 p23 2 2 <t< td=""><td>1</td><td></td><td></td><td>1</td><td></td><td>1</td><td></td></t<>	1			1		1	
PSHUFL/HW							
PSHUFL/HW V,m,i V,m,i V,v,i 1 1 p5 3 1 VSHUFI32X4 V,v,v,i 1 1 p5 3 1 VSHUFI64X2 V,v,v,i 1 1 p5 3 1 V,m,i V,v,v,i 1 1 p5 3 1 V,m,i V,v,v,i 1 1 p5 1 1 V,m,i V,v,v,i 1 1 p5 3 1 VALIGNDR V,m,i V,v,v,i 1 1 p5 3 1 VALIGND/Q V,v,m,i 2 2 p23 p5 1 VALIGND/Q V,v,m,i 2 2 p23 p5 1 VALIGND/Q V,v,m,i 2 2 p23 p5 1 V,v,v,v,v,v,v,v,v,v,v,v,v,v,v,v,v,v,v,v						1	
VSHUFI32X4 V,V,V,I 1 1 p5 3 1 VSHUFI64X2 v,v,i,v,i 1 1 p5 3 1 PALIGNR v,v,i/v,v,mi 1 1 p5 3 1 PALIGNR v,mi/v,v,mi 2 2 p23 p5 1 1 VALIGND/Q v,v,v,i 1 1 p5 3 1 VALIGND/Q v,v,mi 2 2 p23 p5 1 1 VALIGND/Q v,v,mi 2 2 p015 p23 1 1 PBLENDVB x,x,xmm0 1 1 p015 1 0.5 PBLENDVB v,v,v,v 2 2 p015 p23 1 1 VPBLENDVB v,v,v,w 3 3 2p015 p23 2 1 VPBLENDWB x,x,i/v,v,vi 1 1 p5 1 1 VPBLENDW x,v,v,vi 1 1 p015 1 0.33						•	
VSHUFI64X2						3	
PALIGNR					· ·		-
PALIGNR v,m,i / v,v,m,i 2 2 p23 p5 1 VALIGND/Q v,v,v,i 1 1 p5 3 1 VALIGND/Q v,v,m,i 2 2 p23 p5 1 PBLENDVB x,x,xmm0 1 1 p015 1 0.5 PBLENDVB x,x,xmm0 2 2 p015 p23 1 1 VPBLENDVB v,v,v,v 2 2 2p015 p23 2 1 VPBLENDVB v,v,m,v 3 3 2p015 p23 2 2 PBLENDW x,x,i / v,v,v,i 1 1 p5 1 1 VPBLENDW x,x,i / v,v,m,i 2 2 p23 p5 1 1 VPBLENDW x,v,v,i,v,m,i 2 2 p23 p5 1 1 VPBLENDM y,v,v,m,i 2 2 p015 p23 0.5 0.5 VPBLENDMB/W v(k},v,v 1 1 p05 3 0.5 <td></td> <td></td> <td>-</td> <td>· ·</td> <td></td> <td></td> <td></td>			-	· ·			
VALIGND/Q v,v,v,i 1 1 p5 3 1 VALIGND/Q v,v,m,i 2 2 p23 p5 1 PBLENDVB x,x,xmm0 1 1 p015 p23 1 VBLENDVB x,m,xmm0 2 2 p015 p23 1 VPBLENDVB v,v,v,v 2 2 2p015 p23 2 VPBLENDW x,x,i/v,v,vi 1 1 p5 1 1 VPBLENDW x,x,i/v,v,vi 1 1 p5 1 1 VPBLENDDW x,x,i/v,v,m,i 2 2 p23 p5 1 1 VPBLENDDD v,v,v,i 1 1 p015 p23 0.5 0.5 VPBLENDDMB/W v{k},v,v 1 1 p015 p23 0.5 0.5 VPBLENDMB/W v{k},v,v 1 1 p05 3 0.5 VPBLENDMB/W v{k},v,v 1 1 p05 3 0.5 VPBLENDMB/W				2			
VALIGND/Q PBLENDVB						3	
PBLENDVB x,x,xmm0 1 1 p015 p23 1 0.5 PBLENDVB x,m,xmm0 2 2 p015 p23 1 1 VPBLENDVB v,v,w,v 2 2 2p015 p23 2 2 PBLENDW x,x,i / v,v,v,i 1 1 p5 1 1 PBLENDW x,x,i / v,v,w,i 1 1 p5 1 1 PBLENDDW x,x,i / v,v,w,i 1 1 p015 1 1 VPBLENDDD/Q v,v,m,i 2 2 p015 p23 0.5 0.5 VPBLENDMB/W v,kl,v,v 1 1 p015 1 0.33 VPBLENDMB/W v,kl,v,v 1 1 p05 3 0.5 VPBLENDMD/Q v,kl,v,v 1 1 p05 3 0.5 VPERMW v,v,v 2 2 p5 4 2 VPERMW v,v,w 1 1 p5 3	VALIGND/Q		2	2			1 1
PBLENDVB	PBLENDVB		1	1		1	0.5
VPBLENDVB v,v,m,v 3 3 2p015 p23 2 PBLENDW x,x,i / v,v,v,i 1 1 p5 1 1 PBLENDW x,m,i / v,v,m,i 2 2 p23 p5 1 1 VPBLENDDD v,v,v,i 1 1 p015 p23 0.5 0.5 VPBLENDMD/Q v,v,m,i 2 2 p015 p23 0.5 0.5 VPBLENDMB/W v,v,v,m,i 2 2 p015 p23 0.5 0.5 VPBLENDMB/W v,vk,v,v 1 1 p05 3 0.5 VPBLENDMB/W v(k},v,v 1 1 p05 3 0.5 VPBLENDMB/W v,v,v 2 2 p5 4 2 VPERMW v,v,w 1 1 p05 3 0.5 VPERMW v,v,m 3 3 1 2 2 p5 p23 1 1 VPERMD y,v,w 1 1	PBLENDVB		2	2			1
PBLENDW x,x,i/v,v,v,i 1 1 p5 1 1 PBLENDW x,m,i/v,v,m,i 2 2 p23 p5 1 VPBLENDDD/Q v,v,v,i 1 1 p015 1 0.33 VPBLENDMD/Q v,v,m,i 2 2 p015 p23 0.5 0.5 VPBLENDMD/Q v{k},v,v 1 1 p05 3 0.5 VPBLENDMD/Q v{k},v,v 1 1 p05 3 0.5 VPBLENDMD/Q v{k},v,v 1 1 p05 1-2 0.5 VPBLENDMD/Q v{k},v,v 1 1 p05 3 0.5 VPBLENDMD/Q v{k},v,v 1 1 p05 3 0.5 VPERMW v,v,v 2 2 p5 4 2 VPERMW v,v,m 3 3 1 vPERMS 1 1 p5 3 1 VPERMQ v,v,m 1 1 <td>VPBLENDVB</td> <td></td> <td>2</td> <td>2</td> <td></td> <td>2</td> <td>1</td>	VPBLENDVB		2	2		2	1
PBLENDW x,m,i / v,v,m,i 2 2 p23 p5 1 VPBLENDD v,v,v,i 1 1 p015 1 0.33 VPBLENDD/Q v,v,m,i 2 2 p015 p23 0.5 VPBLENDMB/W v{k},v,v 1 1 p05 3 0.5 VPBLENDMD/Q v{k},v,v 1 1 p05 1-2 0.5 VPERMW v,v,v 2 2 p5 4 2 VPERMW v,v,m 3 3 2 2 VPERMD y,y,y 1 1 p5 3 1 VPERMD y,y,m 1 2 p5 p23 1 1 VPERMQ v,v,i 1 1 p5 3 1 1 VPERMQ v,v,w 1 1 p5 3 1 1 p5 3 1 1 p5 3 1 1 p5 3 1 1<	VPBLENDVB	v,v,m,v	3	3	2p015 p23		2
VPBLENDD v,v,v,i 1 1 p015 1 0.33 VPBLENDD/Q v,v,m,i 2 2 p015 p23 0.5 VPBLENDMB/W v{k},v,v 1 1 p05 3 0.5 VPBLENDMD/Q v{k},v,v 1 1 p05 1-2 0.5 VPERMW v,v,v 2 2 p5 4 2 VPERMW v,v,m 3 3 2 2 VPERMW v,v,m 3 3 1 v VPERMD y,y,y 1 1 p5 3 1 VPERMD y,y,m 1 2 p5 p23 1 1 VPERMD y,y,mi 2 2 p5 p23 1 1 VPERMQ v,v,i 1 1 p5 3 1 1 1 p5 p23 1 1 1 p5 p23 1 1 1 1 p5 p23 1 1	PBLENDW	x,x,i / v,v,v,i	1	1	p5	1	1
VPBLENDD/Q v,v,m,i 2 2 p015 p23 0.5 VPBLENDMB/W v{k},v,v 1 1 p05 3 0.5 VPBLENDMD/Q v{k},v,v 1 1 p05 1-2 0.5 VPERMW v,v,v 2 2 p5 4 2 VPERMW v,v,m 3 3 2 2 VPERMW v,v,m 3 3 1 2 VPERMD y,y,y 1 1 p5 3 1 VPERMD y,y,m 1 2 p5 p23 1 1 VPERMQ v,v,i 1 1 p5 3 1 VPERMQ v,v,m 1 2 p5 p23 1 1 VPERMQ v,v,m 1 2 p5 p23 1 1 VPERM2128 y,y,y,i 1 1 p5 3 1 VPERM228 y,y,m,i 2 2	PBLENDW	x,m,i / v,v,m,i	2	2	p23 p5		1
VPBLENDMB/W V{k},v,v 1 1 p05 3 0.5 VPBLENDMD/Q V{k},v,v 1 1 p05 1-2 0.5 VPERMW V,v,v 2 2 p5 4 2 VPERMW V,v,m 3 3 2 2 VPERMD Y,v,m 1 1 p5 3 1 VPERMD Y,y,y,m 1 2 p5 p23 1 1 VPERMQ V,v,i 1 1 p5 3 1 1 VPERMQ V,v,i 1 1 p5 3 1 1 1 p5 3 1 1 1 1 1 p5 3 1	VPBLENDD	v,v,v,i	1	1	p015	1	0.33
VPBLENDMD/Q v{k},v,v 1 1 p05 1-2 0.5 VPERMW v,v,v 2 2 p5 4 2 VPERMW v,v,m 3 3 2 2 VPERMD y,y,y 1 1 p5 3 1 VPERMD y,y,y,m 1 2 p5 p23 1 1 VPERMQ v,v,i 1 1 p5 3 1	VPBLENDD/Q	v,v,m,i	2	2	p015 p23		0.5
VPERMW v,v,v 2 2 p5 4 2 VPERMW v,v,m 3 3 3 2 VPERMD y,y,y 1 1 p5 3 1 VPERMD y,y,m 1 2 p5 p23 1 1 VPERMQ v,v,i 1 1 p5 3 1 VPERMQ v,w,i 2 2 p5 p23 1 1 VPERMQ v,v,w 1 1 p5 3 1 1 vPERD 3 1 1 1 p5 p23 1 1 1 1 p5 p23 1 1	VPBLENDMB/W	v{k},v,v	1	1	p05	3	0.5
VPERMW v,v,m 3 3 p5 3 1 VPERMD y,y,y 1 1 p5 3 1 VPERMD y,y,m 1 2 p5 p23 1 VPERMQ v,v,i 1 1 p5 3 1 VPERMQ v,m,i 2 2 p5 p23 1 1 VPERMQ v,v,m 1 2 p5 p23 1 1 VPERMQ v,v,m 1 2 p5 p23 1 1 VPERM21128 y,y,m,i 2 2 p5 p23 1 1 VPERM2128 1 1 p5 3 1 1 VPERM2128 y,y,m,i 2 2 p5 p23 1 1 VPERM2128 y,y,m,i 2 2 p5 p23 1 1 NPERM2128 y,y,v,v 1 1 p5 3 1 1 NPERM2129 y,v,v 1 1 p5 3 <td< td=""><td></td><td>v{k},v,v</td><td>l -</td><td></td><td>p05</td><td>1-2</td><td></td></td<>		v{k},v,v	l -		p05	1-2	
VPERMD y,y,y 1 1 p5 3 1 VPERMD y,y,m 1 2 p5 p23 1 VPERMQ v,v,i 1 1 p5 p23 1 VPERMQ v,m,i 2 2 p5 p23 1 VPERMQ v,v,w 1 1 p5 p23 1 VPERMQ v,v,m 1 2 p5 p23 1 VPERM2I128 y,y,y,i 1 1 p5 p23 1 VPERM128 y,y,m,i 2 2 p5 p23 1 VPERM128 y,y,m,i 2 2 p5 p23 1 VPERM128 y,v,v 3 3 p01 2p5 7 2 VPERM129 v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 VPERMT2Q </td <td></td> <td>V,V,V</td> <td></td> <td>1</td> <td>p5</td> <td>4</td> <td></td>		V,V,V		1	p5	4	
VPERMD y,y,m 1 2 p5 p23 1 VPERMQ v,v,i 1 1 p5 p23 1 VPERMQ v,m,i 2 2 p5 p23 1 VPERMQ v,v,v 1 1 p5 p23 1 VPERMQ v,v,m 1 2 p5 p23 1 VPERMQ v,v,m 1 2 p5 p23 1 VPERM2I128 y,y,y,i 1 1 p5 3 1 VPERM12B y,y,m,i 2 2 p5 p23 1 1 VPERM12W v,v,v 3 3 p01 2p5 7 2 VPERM12D v,v,v 1 1 p5 3 1 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1		v,v,m					
VPERMQ v,v,i 1 1 p5 3 1 VPERMQ v,m,i 2 2 p5 p23 1 VPERMQ v,v,v 1 1 p5 3 1 VPERMQ v,v,m 1 2 p5 p23 1 1 VPERM2I128 y,y,y,i 1 1 p5 3 1 VPERM2I128 y,y,m,i 2 2 p5 p23 1 VPERM2I128 y,y,m,i 2 2 p5 p23 1 VPERMI2W v,v,v 3 3 p01 2p5 7 2 VPERMI2D v,v,v 1 1 p5 3 1 VPERMT2W v,v,v 3 3 p01 2p5 6 2 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVDQ mm,mm 4 4		y,y,y	1			3	
VPERMQ v,m,i 2 2 p5 p23 1 VPERMQ v,v,v 1 1 p5 p23 1 VPERMQ v,v,m 1 2 p5 p23 1 VPERM2I128 y,y,y,i 1 1 p5 p23 1 VPERM2I128 y,y,m,i 2 2 p5 p23 1 VPERM12W v,v,v 3 3 p01 2p5 7 2 VPERM12D v,v,v 1 1 p5 3 1 VPERM12Q v,v,v 1 1 p5 3 1 VPERM12W v,v,v 3 3 p01 2p5 6 2 VPERM12Q v,v,v 1 1 p5 3 1 VPERM12D v,v,v 1 1 p5 3 1 VPERM12Q v,v,v 1 1 p5 3 1 VPERM12Q v,v,v 1 1 p5 3							
VPERMQ v,v,v 1 1 p5 3 1 VPERMQ v,v,m 1 2 p5 p23 1 VPERM2I128 y,y,y,i 1 1 p5 p23 1 VPERM2I128 y,y,m,i 2 2 p5 p23 1 VPERMI2W v,v,v 3 3 p01 2p5 7 2 VPERMI2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mmm 4 4 p0 p4 2p23 ~300 2 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3						3	
VPERMQ v,v,m 1 2 p5 p23 1 VPERM2I128 y,y,y,i 1 1 p5 p23 1 VPERM2I128 y,y,m,i 2 2 p5 p23 1 VPERMI2W v,v,v 3 3 p01 2p5 7 2 VPERMI2D v,v,v 1 1 p5 3 1 VPERMI2Q v,v,v 1 1 p5 3 1 VPERMT2W v,v,v 3 3 p01 2p5 6 2 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p0 p4 2p23 ~300 2 MASKMOVDQQ x,x 10 10 </td <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>							
VPERM2I128 y,y,y,i 1 1 p5 3 1 VPERM2I128 y,y,m,i 2 2 p5 p23 1 VPERMI2W v,v,v 3 3 p01 2p5 7 2 VPERMI2D v,v,v 1 1 p5 3 1 VPERMI2Q v,v,v 1 1 p5 3 1 VPERMT2W v,v,v 3 3 p01 2p5 6 2 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1						3	
VPERM2I128 y,y,m,i 2 2 p5 p23 1 VPERMI2W v,v,v 3 3 p01 2p5 7 2 VPERMI2D v,v,v 1 1 p5 3 1 VPERMI2Q v,v,v 1 1 p5 3 1 VPERMT2W v,v,v 3 3 p01 2p5 6 2 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1							
VPERMI2W v,v,v 3 3 p01 2p5 7 2 VPERMI2D v,v,v 1 1 p5 3 1 VPERMI2Q v,v,v 1 1 p5 3 1 VPERMT2W v,v,v 3 3 p01 2p5 6 2 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 300-800 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1			l -		· ·	3	
VPERMI2D v,v,v 1 1 p5 3 1 VPERMI2Q v,v,v 1 1 p5 3 1 VPERMT2W v,v,v 3 3 p01 2p5 6 2 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 300-800 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1				1		7	
VPERMI2Q v,v,v 1 1 p5 3 1 VPERMT2W v,v,v 3 3 p01 2p5 6 2 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 300-800 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1							
VPERMT2W v,v,v 3 3 p01 2p5 6 2 VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 300-800 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1							
VPERMT2D v,v,v 1 1 p5 3 1 VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 300-800 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1						1	
VPERMT2Q v,v,v 1 1 p5 3 1 MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 300-800 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1	1						
MASKMOVQ mm,mm 4 4 p0 p4 2p23 ~300 2 MASKMOVDQU x,x 10 10 4p04 2p56 4p23 300-800 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1			-				
MASKMOVDQU x,x 10 10 4p04 2p56 4p23 300-800 6 VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1			· ·				
VPMASKMOVD/Q v,v,m 2 2 p23 p015 4 0.5 VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1							
VPMASKMOVD/Q m,v,v 3 3 p0 p4 p23 4 1							
				1			

			,				
PEXTRB/W/D/Q	r,x,i	2	2	p0 p5	3	1	
PEXTRB/W/D/Q	m,x,i	2	3	p23 p4 p5		1	
VEXTRACTI128	x,y,i	1	1	p5	3	1	
VEXTRACTI128	m,y,i	2	2	p23 p4	4	1	
PINSRB	x,r32,i	2	2	2p5	3	2	
PINSRB	x,m8,i	2	2	p23 p5		1	
PINSRW	mm/x,r32,i	2	2	p5	3	2	
PINSRW	mm/x,m16,i	2	2	p23 p5		1	
PINSRD/Q	x,r,i	2	2	2p5	3	2	
PINSRD/Q	x,m,i	2	2	p23 p5		1	
VINSERTI128	y,y,x,i	1	1	p5	3	1	
VINSERTI128	y,y,m,i	2	2	p015 p23	4-5	0.5	
VINSERTI32x4	z,z,x,i	1	1	p5	3	1	
VINSERTI32x4	z,z,m,i	1	2	p05 p23	4-5	0.5	
VINSERTI64x4	z,z,y,i	1	1	p5	3	1	
VINSERTI64x4	z,z,m,i	1	2	p05 p23	4-5	0.5	
VPBROADCAST	۵,۷,۰۰۰,۰		_	P00 P20		0.0	
B/W/D/Q	x,x	1	1	p5	1	1	
VPBROADCAST							
B/W	v,m	2	2	p23 p5		1	
VPBROADCAST		_		00		0.5	
D/Q	v,m	1	1	p23		0.5	
VPBROADCAST B/W/D/Q	VV	1	1	n5	3	1	
·	v,x y,m128	1	1	p5 p23	3	0.5	
VBROADCASTI128	-	1	1		3	1	
VBROADCASTI32X2	Z,X,İ	1	1	p5	4	0.5	
VBROADCASTI32X4/I64x2		1	1	p23			
VBROADCASTI32X8/i64x4		2	2	p23	4	0.5 2	
VPCOMPRESSD/Q	v{k},v	2	2	p5	3 3	2	
VPEXPANDD/Q	v{k},v			p5	3		
VPGATHERDD	x,[r+s*x],x	4	4	p0 p1 p23 p5		4	
VPGATHERDD	y,[r+s*y],y	4	4	p0 p1 p23 p5		5	
VPGATHERDQ	x,[r+s*x],x	4	5	p0 p1 p23 p5		2	
VPGATHERDQ	y,[r+s*x],y	4	4	p0 p1 p23 p5		4	
VPGATHERQD	x,[r+s*x],x	4	5	p0 p1 p23 p5		2	
VPGATHERQD	x,[r+s*y],x	4	4	p0 p1 p23 p5		4	
VPGATHERQQ	x,[r+s*x],x	4	5	p0 p1 p23 p5		2	
VPGATHERQQ	y,[r+s*y],y	4	4	p0 p1 p23 p5		4	
VPGATHERDD	x{k},[r+s*x]	4	4	p0 p1 p23 p5		4	
VPGATHERDD	y{k},[r+s*y]	4	4	p0 p1 p23 p5		5	
VPGATHERDD	z{k},[r+s*z]	4	4	p0 p1 p23 p5		9	
VPGATHERDQ	x{k},[r+s*x]	4	5	p0 p1 p23 p5		3	
VPGATHERDQ	y{k},[r+s*x]	4	4	p0 p1 p23 p5		4	
VPGATHERDQ	z{k},[r+s*y]	4	4	p0 p1 p23 p5		5	
VPGATHERQD	x{k},[r+s*x]	4	5	p0 p1 p23 p5		3	
VPGATHERQD	x{k},[r+s*y]	4	4	p0 p1 p23 p5		4	
VPGATHERQD	y{k},[r+s*z]	4	4	p0 p1 p23 p5		5	
VPGATHERQQ	x{k},[r+s*x]	4	5	p0 p1 p23 p5		3	
VPGATHERQQ	y{k},[r+s*y]	4	4	p0 p1 p23 p5		4	
VPGATHERQQ	$z\{k\},[r+s*z]$	4	4	p0 p1 p23 p5		5	
VPSCATTERDD	$[r+s*x]\{k\},x$	19	19			10	
VPSCATTERDD	[r+s*y]{k},y	27	27			12	
VPSCATTERDD	$[r+s*z]\{k\},z$	43	43		17-20	17	
VPSCATTERDQ	[r+s*x]{k},x	15	15			9	

			,				
VPSCATTERDQ	[r+s*x]{k},y	19	19			10	
VPSCATTERDQ	[r+s*y]{k},z	27	27			12	
VPSCATTERQD	[r+s*x]{k},x	15	15			9	
VPSCATTERQD	[r+s*y]{k},y	19	19			10	
VPSCATTERQD	[r+s*z]{k},z	27	27			12	
VPSCATTERQQ	[r+s*x]{k},x	15	15			9	
VPSCATTERQQ	[r+s*y]{k},y	19	9			10	
VPSCATTERQQ	[r+s*z]{k},z	27	27			12	
VIOCATILINGQ	[113 2][K],2		21			12	
Arithmetic instruc-							
tions							
PADD/SUB(S,US) B/							
W/D/Q	mm,mm	1	1	p05	1	0.5	
PADD/SUB(S,US) B/							
W/D/Q	x,x / xy,xy,xy	1	1	p015	1	0.33	
PADD/SUB(S,US) B/							
W/D/Q	Z,Z,Z	1	1	p05	1	0.5	
PADD/SUB(S,US) B/		_		04500		0.5	
W/D/Q	v,m / v,v,m	1	2	p015 p23		0.5	
PHADD(S)W/D				04.05			
PHSUB(S)W/D	v,v / v,v,v	3	3	p01 2p5	3	2	
PHADD(S)W/D				04.0			
PHSUB(S)W/D	v,m / v,v,m	4	4	p01 2p5 p23		2	
PCMPEQB/W/D		4	_	0	4	4	
PCMPGTB/W/D	mm,mm	1	1	p0	1	1	
PCMPEQB/W/D	**/***	4	4	501	1	0.5	
PCMPGTB/W/D	x,x / y,y,y	1	1	p01	Į.	0.5	
PCMPEQB/W/D PCMPGTB/W/D	x,m / y,y,m	1	2	p01 p23		0.5	
PCMPEQQ	v,v / v,v,v		1	p01 p23	1	0.5	
PCMPEQQ	v,w / v,v,w v,m / v,v,m		2	p01 p23	ı	0.5	
PCMPGTQ		1	1		3	1	
	v,v / v,v,v	1	2	p5	3	1	
PCMPGTQ	v,m / v,v,m		1	p5 p23	_	1 1	
VPCMPB/W/D/Q	k{k},v,v	1	1	p5	3	1	
PMULL/HW PMULHUW		4	_	0	5	4	
	mm,mm	1	1	p0	5	1	
PMULL/HW PMULHUW	v v / v v v	1	1	p01	5	0.5	
VPMULL/HW	x,x / y,y,y	'	1	ροι	5	0.5	
VPMULHUW	Z,Z,Z	1	1	p0	5	1	
PMULL/HW	2,2,2	'	1	Po]	!	
PMULHUW	x,m / xy,xy,m	1	2	p01 p23		0.5	
VPMULLQ	V,V,V	3	3	p01 p20	15	1.5-3	
PMULHRSW	mm,mm	1	1	p0	5	1.0 0	
PMULHRSW	x,x / y,y,y	1	1	p01	5	0.5	
PMULHRSW	x,m / y,y,m		2	p01 p23]	0.5	
PMULLD			2		10	1	
PMULLD	x,x / y,y,y	2 2	2	2p01	10	2	
	Z,Z,Z		3	2p0	10	1	
PMULLD	x,m / y,y,m	3		2p01 p23		1	
PMULDQ	x,x / y,y,y	1	1	p01	5	0.5	
PMULDQ	Z,Z,Z	2	2	2p0	10	2	
PMULDQ	x,m / y,y,m	1	2	p01 p23	_	0.5	
PMULUDQ	mm,mm	1	1	p0	5	1	
PMULUDQ	x,x / y,y,y	1	1	p01	5	0.5	
PMULUDQ	Z,Z,Z	1	1	p0	5	1	

			Ortylard	υ λ			
PMULUDQ	x,m / y,y,m	1	2	p01 p23		0.5	
PMADDWD	mm,mm	1	1	p0	5	1	
PMADDWD	x,x / y,y,y	1	1	p01	5	0.5	
PMADDWD	Z,Z,Z	1	1	p0	5	1	
PMADDWD	x,m / y,y,m	1	2	p01 p23		0.5	
PMADDUBSW	mm,mm	1	1	p01 p20	5	1	
PMADDUBSW		1	1			0.5	
	x,x / y,y,y			p01	5		
PMADDUBSW	Z,Z,Z	1	1	p0	5	1	
PMADDUBSW	x,m / y,y,m	1	2	p01 p23		0.5	
PAVGB/W	mm,mm	1	1	p0	1	1	
PAVGB/W	x,x / y,y,y	1	1	p01	1	0.5	
PAVGB/W	Z,Z,Z	1	1	p0	1	1	
PAVGB/W	x,m / y,y,m	1	2	p01 p23		0.5	
PMIN/PMAX							
SB/SW/SD				_			
UB/UW/UD	mm,mm	1	1	p0	1	1	
PMIN/PMAX							
SB/SW/SD		4		04		0.5	
UB/UW/UD	x,x / y,y,y	1	1	p01	1	0.5	
PMIN/PMAX SB/SW/SD							
UB/UW/UD	777	1	1	p0	1	1	
PMIN/PMAX	Z,Z,Z	ı	1	ρυ	'	1	
SB/SW/SD							
UB/UW/UD	x,m / y,y,m	1	2	p01 p23		0.5	
PHMINPOSUW	X,111 / y,y,111 X,X	1	1	p01 p20	4	1	
PHMINPOSUW	x,m128	1	2	p0 p23	4	1	
PABSB/W/D		1	1		1		
PABSB/W/D	mm,mm	1	1	p0		0.5	
	x,x / y,y			p01	1		
PABSB/W/D	Z,Z,Z	1	1	p0	1	1	
PABSB/W/D	x,m/y,m	1	2	p01 p23		0.5	
PSIGNB/W/D	mm,mm	1	1	p0	1	1	
PSIGNB/W/D	x,x / y,y,y	1	1	p01	1	0.5	
PSIGNB/W/D	x,m / y,y,m	1	2	p01 p23		0.5	
PSADBW	v,v / v,v,v	1	1	p5	3	1	
PSADBW	v,m / v,v,m	1	2	p5 p23		1	
MPSADBW	x,x,i / v,v,v,i	2	2	2p5	4	2	
MPSADBW	x,m,i / v,v,m,i	3	3	2p5 p23		2	
Logic instructions							
PAND PANDN							
POR PXOR	mm,mm	1	1	p05	1	0.5	
PAND PANDN							
POR PXOR	x,x / y,y,y	1	1	p015	1	0.33	
PAND PANDN							
POR PXOR	Z,Z,Z	1	1	p05	1	0.5	
PAND PANDN							
POR PXOR	v,m / v,v,m	1	2	p015 p23		0.5	
PTEST	V,V	2	2	p0 p5	3	1	
PTEST	v,m	2	3	p0 p5 p23		1	
VPTESTMB/W/D/Q		,					
VPTESTNMB/W/D/Q	k,v,v	1	1		3	1	
PSLLW/D/Q							
PSRLW/D/Q	mana mara	4	4	50	4	4	
PSRAW/D/Q	mm,mm	1	1	p0	1	1	

1
1
1
0.5
0.5
1
'
0.5
0.0
1
0.5
1
1
0.5
1
0.5
1
0.5
1
0.5
7
10
20
2
7
12
4
4
5
5
3
3
3
3
1
1

AESDEC, AESDECLAST, AESENC, AESENCLAST AESDEC, AESDECLAST, AESENC,	x,x	1	1	p0	4	1	
AESENCLAST	x,m	2	2	p0 p23		1	
AESIMC	x,x	2	2	2p0	8	2	
AESIMC	x,m	3	3	2p0 p23		2	
AESKEYGENASSIST	x,x,i	13	13	p0 p5	12	12	
AESKEYGENASSIST	x,m,i	13	13			12	
Other							
EMMS		10	10	p05		6	

Floating point vector instructions

Floating point ve		μοps fused	µops unfused			Recipro- cal through	
Instruction	Operands	domain	domain	µops each port	Latency	put	Comments
Move instructions							
MOVAPS/D	x,x	1	1	p015	0-1	0.25	may eliminate may
VMOVAPS/D	у,у	1	1	p015	0-1	0.25	eliminate may
VMOVAPS/D	z,z	1	1	p05	0-1	0.33	eliminate
MOVAPS/D				·			
MOVUPS/D	x,m128	1	1	p23	2	0.5	
VMOVAPS/D							
VMOVUPS/D	v,m	1	1	p23	3	0.5	
MOVAPS/D	400			007.4			
MOVUPS/D	m128,x	1	2	p237 p4	3	1	
VMOVAPS/D VMOVUPS/D	m256,y	1	2	p237 p4	3	1	
MOVSS/D	X,X	1	1	p237 p4	1	1	
MOVSS/D	x,m32/64	1	1	p23	2	0.5	
MOVSS/D	m32/64,x	1	2	p237 p4	3	1	
MOVHPS/D	x,m64	1	2	p237 p4	4	1 1	
MOVHPS/D	m64,x	1	2	p4 p237	3	1 1	
MOVLPS/D	x,m64	1 1	2	p23 p5	4	1	
MOVLPS/D	m64,x	1	2	p4 p237	3	1	
MOVHLPS	X,X	1	1	p4 p207	1	1	
MOVLHPS	X,X X,X	1	1	p5	1	1	
MOVMSKPS/D	r32,x	1	1	p0	2	1	
VMOVMSKPS/D	r32,y	1	1	p0	3	1	
MOVNTPS/D	m128,x	1	2	p4 p237	~400	1	
VMOVNTPS/D	m256,y	1	2	p4 p237	~400	1	
SHUFPS/D	x,x,i / v,v,v,i	1	1	p5	1	1	
SHUFPS/D	x,m,i / v,v,m,i	2	2	p5 p23		1	
VSHUFF32X4	v,v,v,i	1	1	p5	3	1	
VSHUFF64X2	v,v,v,i	1	1	p5	3	1	
VPERMILPS/PD	v,v,i	1	1	p5	1	1	
VPERMILPS/PD	v,m,i	2	2	p5 p23		1	

			Skylaki			
VPERMILPS/PD	v,v,v	1	1	p5	1	1 1
VPERMILPS/PD	v,v,m	2	2	p5 p23		1
VPERM2F128	y,y,y,i	1	1	p5	3	1 1
VPERM2F128	y,y,m,i	2	2	p5 p23	_	1 1
VPERMPS	V,V,V	1	1	p5	3	1
VPERMPS	y,y,m	1	2	p5 p23		1
VPERMPD	y,y,iii	1	1	p5	3	1
VPERMPD	y,y,i y,m,i	2	2	p5 p23		1
VPERMI2PS/PD	V,V,V	1	1	p5	3	1 1
VPERMT2PS/PD	V,V,V	1	1	p5	3	1 1
BLENDPS/PD	x,x,i / v,v,v,i	1	1	p015	1	0.33-0.5
BLENDPS/PD	x,m,i / v,v,m,i	2	2	p015 p23	'	0.55
BLENDVPS/PD	x,x,xmm0	1	1	p015 p25	1	1
BLENDVPS/PD	x,m,xmm0	2	2	p015 p23	!	1 1
VBLENDVPS/PD	V,V,V,V	2	2	2p015	2	1
VBLENDVPS/PD	v,v,v,v v,v,m,v	3	3	2p015 2p015 p23		
VBLENDWPS/PD		1	1	p05	1	0.5
	v{k},v,v		-	•		
MOVDDUP	V,V	1	1	p5	1	1 1
MOVDDUP	v,m	1	1	p23	3	0.5
VBROADCASTSS	x,m32	1	1	p23	2	0.5
VBROADCASTSS	y,m32	1	1	p23	3	0.5
VBROADCASTSS	z,m32	1	1	p23	4	0.5
VBROADCASTSS	X,X	1	1	p5	1	1
VBROADCASTSS	V,X	1	1	p5	3	1
VBROADCASTSD	y,m64	1	1	p23	3	0.5
VBROADCASTSD	z,m64	1	1	p23	4	0.5
VBROADCASTSD	V,X	1	1	p5	3	1
VBROADCASTF128	y,m128	1	1	p23	3	0.5
MOVSH/LDUP	V,V	1	1	p5	1	1
MOVSH/LDUP	v,m	1	1	p23	3-4	0.5
UNPCKH/LPS/D	x,x / v,v,v	1	1	p5	1	1
UNPCKH/LPS/D	x,m / v,v,m	1	2	p5 p23		1
EXTRACTPS	r32,x,i	2	2	p0 p5	3	1
EXTRACTPS	m32,x,i	2	3	p4 p5 p23	5	1
VEXTRACTF128	x,y,i	1	1	p5	3	1
VEXTRACTF128	m128,y,i	2	2	p23 p4	5	1
VEXTRACTF32X4						
VEXTRACTF64X2	x,z,i	1	1	p5	3	1
VEXTRACTF32X4	400 .	•				
VEXTRACTF64X2	m128,z,i	2	2	p23 p4	3	1
INSERTPS	x,x,i	1	1	p5	1	1
INSERTPS	x,m32,i	2	2	p23 p5	4	1
VINSERTF128	y,y,x,i	1	1	p5	3	1
VINSERTF128	y,y,m128,i	2	2	p015 p23	5	0.5
VEXTRACTF32X4				_	_	
VEXTRACTF64X2	z,z,x,i	1	1	p5	3	1
VEXTRACTF32X4	400 :	•		05.00	_	0.5
VEXTRACTF64X2	z,z,m128,i	2	2	p05 p23	5	0.5
VMASKMOVPS/D	v,v,m	2	2	p015 p23	3	0.5
VMASKMOVPS/D	m,xy,xy	3	3	p0 p4 p23	13	1
VCOMPRESPS/PD	v{k},v	2	2	p5	3	2
VEXPANDPS/PD	v{k},v	2	2	p5	3	2
VGATHERDPS	x,[r+s*x],x	4	4	p0 p1 p23 p5	12	4
VGATHERDPS	y,[r+s*y],y	4	4	p0 p1 p23 p5	13	5

VGATHERQPS	x,[r+s*x],x	4	5	p0 p1 p23 p5		2	
VGATHERQPS	x,[r+s*y],x	4	4	p0 p1 p23 p5		4	
VGATHERDPD	x,[r+s*x],x	4	5	p0 p1 p23 p5		2	
VGATHERDPD	y,[r+s*x],y	4	4	p0 p1 p23 p5		4	
VGATHERQPD	x,[r+s*x],x	4	4	p0 p1 p23 p5		2	
VGATHERQPD	y,[r+s*y],y	4	4	p0 p1 p23 p5		4	
VGATHERDPS	x{k},[r+s*x]	4	4	p0 p1 p23 p5		4	
VGATHERDPS	y{k},[r+s*y]	4	4	p0 p1 p23 p5		5	
VGATHERDPS	z{k},[r+s*z]	4	4	p0 p1 p23 p5		9	
VGATHERDPD	x{k},[r+s*x]	4	5	p0 p1 p23 p5		3	
VGATHERDPD	y{k},[r+s*x]	4	4	p0 p1 p23 p5		4	
VGATHERDPD	z{k},[r+s*y]	4	4	p0 p1 p23 p5		5	
VGATHERQPS	x{k},[r+s*x]	4	5	p0 p1 p23 p5		3	
VGATHERQPS	x{k},[r+s*y]	4	4	p0 p1 p23 p5		4	
VGATHERQPS	y{k},[r+s*z]	4	4	p0 p1 p23 p5		5	
VGATHERQPD	x{k},[r+s*x]	4	5	p0 p1 p23 p5		3	
VGATHERQPD	y{k},[r+s*y]	4	4	p0 p1 p23 p5		4	
VGATHERQPD	z{k},[r+s*z]	4	4	p0 p1 p23 p5		5	
VPSCATTERDPS	[r+s*x]{k},x	19	19			10	
VPSCATTERDPS	[r+s*y]{k},y	27	27			12	
VPSCATTERDPS	[r+s*z]{k},z	43	43		17-20	17	
VPSCATTERDPD	[r+s*x]{k},x	15	15			9	
VPSCATTERDPD	[r+s*x]{k},y	19	19			10	
VPSCATTERDPD	[r+s*y]{k},z	27	27			12	
VPSCATTERQPS	[r+s*x]{k},x	15	15			9	
VPSCATTERQPS	[r+s*y]{k},y	19	19			10	
VPSCATTERQPS	[r+s*z]{k},z	27	27			12	
VPSCATTERQPD	[r+s*x]{k},x	15	15			9	
VPSCATTERQPD	[r+s*y]{k},y	19	9			10	
VPSCATTERQPD	[r+s*z]{k},z	27	27			12	
Conversion							
CVTPD2PS	x,x	2	2	p01 p5	5	1	
VCVTPD2PS	x,y / y,z	2	2	p01 p5	7	1	
CVTPS2PD	X,X	2	2	p01 p5	5	1	
VCVTPS2PD	y,x / z,y	2	2	p01 p5	7	1	
CVTSS2SD	X,X	2	2	p01 p5	5	2	
CVTSD2SS	x,x	2	2	p01 p5	5	1	
CVT(T) PS2DQ	X,X	1	1	p01	4	0.5	
VCVT(T) PS2DQ	y,y	1	1	p01	4	0.5	
VCVT(T) PS2DQ	Z,Z	1	1	p05	4	0.5	
CVTDQ2PS	X,X	1	1	p01	4	0.5	
VCVTDQ2PS	y,y	1	1	p01	4	0.5	
VCVTDQ2PS	Z,Z	1	1	p05	4	0.5	
VCVT(T)PS2QQ	V,V	2	2	p01 p5	7	1	
VCVTQQ2PS	V,V	2	2	p01 p5	7	1	
VCVT(T)PS2UDQ	V,V	1	1	p01	4	0.5-1	
VCVTUDQ2PS	V,V	1	1	p01	4	0.5-1	
VCVT(T)PS2UQQ	V,V	2	2	p01 p5	7	1	
VCVTUQQ2PS	V,V	2	2	p01 p5	7	1	
CVT(T)PS2PI	mm,x	2	2	p0 p5	7	1	
CVTPI2PS	x,mm	2	2	p0 p1	6	2	
CVT(T)SS2SI	r32,x	2	2	2p01	6	1	

			Oltylak			
CVT(T)SS2SI	r64,x	3	3	2p01 p5	7	1 1
CVT(T)SS2SI	r32,m32	3	3	2p01 p23		1 1
CVTSI2SS	x,r32	2	2	p01 p5	6	2
CVTSI2SS	x,r64	3	3	p01 2p5	7	2
CVTSI2SS	x,m32	1	2	p1 p23	,	3
	· ·	2	2		6	
CVT(T)SS2USI	r32,x			2p01		1 1
VCVTPS2PH	x,v,i	2	2	p01 p5	5-7	1
VCVTPS2PH	y,z,i	2	2	p0 p5	5-7	1
VCVTPH2PS	V,X	2	2	p01 p5	5-7	1
VCVTPH2PS	z,y	2	2	p0 p5	5-7	1
CVT(T)PD2DQ	X,X	2	2	p01 p5	5	1
VCVT(T)PD2DQ	x,y	2	2	p01 p5	7	1
VCVT(T)PD2DQ	y,z	2	2	p0 p5	7	1 1
CVTDQ2PD	x,x	2	2	p01 p5	5	1 1
VCVTDQ2PD	y,x	2	2	p01 p5	7	1 1
VCVTDQ2PD	z,y	2	2	p0 p5	7	1
VCVT(T)PD2QQ	V,V	1	1	p01	4	0.5-1
VCVTQQ2PD		1	1	p01	4	0.5-1
	V,V	2	2	•		
VCVT(T)PD2UDQ	X,X			p01 p5	5	1
VCVT(T)PD2UDQ	x,y	2	2	p01 p5	7	1
VCVT(T)PD2UDQ	y,z	2	2	p0 p5	7	1
VCVTUDQ2PD	X,X	2	2	p01 p5	5	1
VCVTUDQ2PD	y,x	2	2	p01 p5	7	1
VCVTUDQ2PD	z,y	2	2	p0 p5	7	1
VCVT(T)PD2UQQ	V,V	1	1	p01	4	0.5-1
VCVTUQQ2PD	V,V	1	1	p01	4	0.5-1
CVT(T) PD2PI	mm,x	2	2	p01 p5	5	1
CVTPI2PD	x,mm	2	2	p01 p5	5	1 1
CVT(T)SD2SI	r32/64,x	2	2	p0 p1	6	1 1
CVTSI2SD	x,r32/64	2	2	p01 p5	6	2
CVT(T)SD2USI	r32/64,x	2	2	p0 p1	6	1
CVTUSI2SD	x,r32/64	2	2	p01 p5	6	1 1
CVTOOIZOD	X,132/04			ροτρο		'
Arithmetic						
ADDSS/D PS/D						
SUBSS/D PS/D	x,x / y,y,y	1	1	p01	4	0.5
VADDPS/D	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	-			-	
VSUBPS/D	Z,Z,Z	1	1	p05	4	0.5-1
ADDSS/D PS/D	_,_,_	-		1 1	-	
SUBSS/D PS/D	x,m / v,v,m	1	2	p01/05 p23		0.5-1
ADDSUBPS/D	x,x / v,v,v	1	1	p01/05	4	0.5-1
ADDSUBPS/D	x,m / v,v,m	1	2	p01/05 p23	•	0.5-1
HADDPS/D	X,111 / V, V,111	'	_	p01/00 p20		0.5-1
HSUBPS/D	x,x / v,v,v	3	3	p01 2p5	6	2
HADDPS/D	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	3		p012p3		
HSUBPS/D	x,m / v,v,m	4	4	p1 2p5 p23		2
MULSS/D PS/D	x,x / v,v,v	1	1	p01	4	0.5-1
VMULPS/D			1 1	•	4	0.5-1
	Z,Z,Z			p05	4	
MULSS/D PS/D	x,m / v,v,m	1	2	p01/05 p23		0.5-1
DIVSS	X,X	1	1	p0	11	3
DIVPS	X,X	1	1	p0	11	3
DIVSS DIVPS	x,m	1	2	p0 p23		3-5
DIVSD	X,X	1	1	p0	13-14	4
DIVPD	x,x	1	1	p0	13-14	4

			,				
DIVSD DIVPD	x,m	1	2	p0 p23		4	
VDIVPS	y,y,y	1	1	p0	11	5	
VDIVPS	Z,Z,Z	3	3	2p0 p5	18	10	
VDIVPS	y,y,m256	1	2	p0 p23		5	
VDIVPD	y,y,y y,y,y	1	1	p0	13-14	8	
VDIVPD		3	3	2p0 p5	24	16	
	Z,Z,Z				24		
VDIVPD	y,y,m256	4	4	p0 p23	_	8	
RCPSS/PS	V,V	1	1	p0	4	1	
RCPSS/PS	v,m	1	2	p0 p23		1	
VRCP14SS/PS	xy,xy	1	1	p0	4	1	
VRCP14SS/PS	Z,Z	3	3	2p0 p5	7	2	
CMPccSS/D							
CMPccPS/D	x,x / v,v,v	1	1	p01	4	0.5	
CMPccSS/D							
CMPccPS/D	x,m / v,v,m	2	2	p01 p23		0.5	
VCMPPS/PD	k,v,v	1	1		3	1	
(U)COMISS/D	x,x	1	1	p0	2	1	
(U)COMISS/D	x,m32/64	2	2	p0 p23		1	
MAXSS/SD/PS/PD	7,11102701	_	_	P0 P20		•	
MINSS/SD/PS/PD	x,x / v,v,v	1	1	p01/05	4	0.5-1	
MAXSS/SD/PS/PD	X,X / V,V,V	•		po 1700	•	0.0 1	
MINSS/SD/PS/PD	x,m / v,v,m	1	2	p01/05 p23		0.5-1	
VRANGESS/SD/PS/PD	V,V,V	1	1	p01/00/p20	4	0.5-1	
		2	2		8		
ROUNDSS/SD/PS/PD	V,V,İ			2p01	0	1	
ROUNDSS/SD/PS/PD	v,m,i	3	3	2p01 p23		1	
VRNDSCALE		•		004/05		_	
SS/PS/SD/PD	v,v,v,i	2	2	2p01/05	8	1	
VREDUCE				04		0.5.4	
SS/PS/SD/PD	v,v,v,i	1	1	p01	4	0.5-1	
DPPS	x,x,i / v,v,v,i	4	4	3p01 p5	13	2	
DPPS	x,m,i / v,v,m,i	6	6	3p01 p23 p5 p6		4	
DPPD	x,x,i	3	3	2p01 p5	9	1	
DPPD	x,m128,i	4	4	2p01 p23 p5		1	
VDBPSADBW	v,v,v,i	1	1	p5	3	1	
VFIXUPIMMSS/SD	x,x,x,i	1	1	p01	4	2	
VFIXUPIMMPS/PD	v,v,v,i	1	1	p01/05	4	2	
VFPCLASSSS/SD	k,x,i	1	1	p5	3	1	
VFPCLASSPS/PD	k,v,i	1	1	p5	3	1	
VFMADD	IX, V,I	'		Po			
(all FMA instr.)	xy,xy,xy	1	1	p01	4	0.5	
VFMADD	, , , , , , , , , , , , , , , , , , ,	'		poi		0.5	
(all FMA instr.)	z,z,z	1	1	p05	4	0.5-1	
VFMADD	2,2,2	'		poo		0.0-1	
(all FMA instr.)	v,v,m	1	2	p01/05 p23		0.5-1	
(all I WA IIIsu.)	V, V, III	'		p01/03 p23		0.5-1	
Math							
Math				0	40	_	
SQRTSS/PS	X,X	1	1	p0	12	3	
VSQRTPS	у,у	1	1	p0	12	6	
VSQRTPS	Z,Z	3	3	p0	20	12	
SQRTSS/PS	x,m128	1	2	p0 p23		3	
VSQRTPS	y,m256	4	4	p0 p23		6	
SQRTSD	x,x	1	1	p0	15-16	4-6	
SQRTPD	x,x	1	1	p0	15-16	4-6	
VSQRTPD	y,y	1	1	p0	15-16	9-12	
,. ogitti b	y,y	•	'	ι ρυ	1 .5 10	J 12	I

			•				
VSQRTPD	Z,Z	3	3	2p0 p5	28-29	18-24	
SQRTSD/PD	x,m128	1	2	p0 p23		4-6	
VSQRTPD	y,m256	4	4	p0 p23		9-12	
RSQRTSS/PS	X,X	1	1	p0	4	1	
RSQRTPS	y,y	1	1	p0	2	1	
VRSQRT14SS/PS	x,x	1	1	p0	4	1	
VRSQRT14PS	y,y	1	1	p0	2	1	
VRSQRT14PS	z,z	3	3	2p0 p5	6	2	
VGETEXPSS/SD	x,x,x	1	1	p01	4	0.5-1	
VGETEXPPS/PD	V,V,V	1	1	p01/05	4	0.5-1	
VGETMANTSS/SD	x,x,x	1	1	p01	4	0.5-1	
VGETMANTPS/PD	V,V,V	1	1	p01/05	4	0.5-1	
VSCALESS/SD/PS/PD	v,v,v	1	1	p01	4	0.5-1	
Lauta							
Logic	_						
AND/ANDN/OR/ XORPS/PD	y y l y y y	1	1	p015	1	0.33	
	x,x / y,y,y	'		ρυ15	I	0.33	
AND/ANDN/OR/ XORPS/PD	z,z,z	1	1	p05	1	0.5	
AND/ANDN/OR/	2,2,2	'		pos	l I	0.5	
XORPS/PD	x,m / y,y,m	1	2	p015 p23		0.5	
AND/ANDN/OR/	X,1117 y, y,111	'	_	p010 p20		0.0	
XORPS/PD	z,z,m	1	2	p05 p23		0.5	
VTESTPS/PD	V,V	1	1	p0		1	
Oth an							
Other VZEROUPPER	_	4	4	none		1	
VZEROUPPER		4	4	none			20 hit made
		25	25 34	p0 p1 p5 p6		12 12	32 bit mode 64 bit mode
VZEROALL	20	34		p0 p1 p5 p6	_	1	64 bit mode
LDMXCSR	m32	4	4	p0 p5 p6 p23	5	3	
STMXCSR	m32	3	4	p0 p4 p6 p237	4	1	
FXSAVE	m	106	136		76	76	32 bit mode
FXSAVE	m	114	136		65	65	64 bit mode
	m					1	
	m					1	
	m						
	m						
	m	199			83	83	32 bit mode
XRSTOR	m	266			116	116	64 bit mode
XSAVEC	m	226			83	83	64 bit mode
XSAVEOPT	m	214			84	84	
	m m m m	266			116	116	64 bit mode

Mask register instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instructions							
KMOVB/W/D/Q	k,k	1	1	p0	1	1	
KMOVB/W/D/Q	k,m	3			3		
KMOVB/W/D/Q	m,k	2	1	p237 p4	4	1	
KMOVB/W/D/Q	k,r	1	1	p5	2	1	

KMOVB/W/D/Q	r,k	1	1	p0	2	1	
VPBROADCASTMB2Q	v,k	2	2	p0 p5	6	1	
VPBROADCASTMW2D	v,k	2	2	p0 p5	6	1	
KUNPCKBW	k,k,k	1	1	p5	4	1	
KUNPCKWD	k,k,k	1	1	p5	4	1	
KUNPCKDQ	k,k,k	1	1	p5	4	1	
VPMOVM2B/W	v,k	1	1	p05	3	0.5	
VPMOVM2D/Q	v,k	1	1	p05	2	0.5	
VPMOVB/W2M	k,v	1	1	p0	3	1	
VPMOVD/Q2M	k,v	1	1	p0	2	1	
Arithmetic instruction	ons						
KADDB/W/D/Q	k,k,k	1	1	p5	4	1	
KSHIFTLB/W/D/Q	k,k,i	1	1	p5	4	1	
KSHIFTRB/W/D/Q	k,k,i	1	1	p5	4	1	
Logic instructions							
KANDB/W/D/Q	k,k,k	1	1	p0	1	1	
KANDNB/W/D/Q	k,k,k	1	1	p0	1	1	
KORB/W/D/Q	k,k,k	1	1	p0	1	1	
KXORB/W/D/Q	k,k,k	1	1	p0	1	1	
KXNORB/W/D/Q	k,k,k	1	1	p0	1	1	
KNOTB/W/D/Q	k,k	1	1	p0	1	1	
KTESTB/W/D/Q	k,k	1	1	p0	2	1	
KORTESTB/W/D/Q	k,k	1	1	p0	2	1	

Intel Coffee Lake

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the same data.

Instructions with and without V name prefix behave the same unless otherwise noted.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm register, mm/

x = mmx or xmm register, y = 256 bit ymm register, v = any vector register (mmx, xmm,

ymm). m = memory operand, m32 = 32-bit memory operand, etc.

μορs fused domain:

The number of µops at the decode, rename and allocate stages in the pipeline. Fused

μops count as one.

μορs unfused domain:

The total number of µops for all execution port. Fused µops count as two. Fused macro-

ops count as one. The instruction has μ op fusion if this number is higher than the number under fused domain. Some operations are not counted here if they do not go to any

execution port or if the counters are inaccurate.

μορs each port: The number of μops for each execution port. p0 means a μop to execution port 0.

p01means a μop that can go to either port 0 or port 1. p0 p1 means two μops going to

port 0 and 1, respectively.

Port 0: Integer, f.p. and vector ALU, mul, div, branch

Port 1: Integer, f.p. and vector ALU

Port 2: Load Port 3: Load Port 4: Store

Port 5: Integer and vector ALU Port 6: Integer ALU, branch Port 7: Store address

Latency:

This is the delay that the instruction generates in a dependency chain. The numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Where hyperthreading is enabled, the use of the same execution units in the other thread leads to inferior performance. Subnormal numbers, NAN's and infinity do not increase the latency. The time unit used is core clock cycles, not the reference clock cycles given by the time stamp acceptor.

ence clock cycles given by the time stamp counter.

Reciprocal throughput:

The average number of core clock cycles per instruction for a series of independent in-

structions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc-							
tions							
MOV	r,i	1	1	p0156		0.25	
MOV	r8/16,r8/16	1	1	p0156	1	0.25	
MOV	r32/64,r32/64	1	1	p0156	0-1	0.25	may be elim.
MOV	r8l,m	1	2	p23 p0156		0.5	
MOV	r8h,m	1	1	p23		0.5	
MOV	r16,m	1	2	p23 p0156		0.5	
MOV	r32/64,m	1	1	p23	2	0.5	all addressing modes
MOV	m,r	1	2	p237 p4	2	1	
MOV	m,i	1	2	p237 p4		1	
MOVNTI	m,r	2	2	p23 p4	~470	2	

MOVSX MOVZX	r,r	1	1	p0156	1	0.25	
MOVSXD	.,.		•	porco	•	0.20	
MOVSX MOVZX	r16,m8	1	2	p23 p0156		0.5	
MOVSX MOVZX MOVSXD	r,m	1	1	p23		0.5	all other combinations
CMOVcc	r,r	1	1	p06	1	0.5	
CMOVcc	r,m	2	2	p06 p23		0.5	
XCHG	r,r	3	3	3p0156	2	1	
XCHG	r,m	8	8		23		implicit lock
XLAT		3	3	p23 2p0156	7	2	
PUSH	r	1	2	p237 p4	3	1	
PUSH	i	1	2	p237 p4		1	
PUSH	m	2	3	p4 2p237		1	
PUSH	stack pointer	2	3	p0156 p237 p4		1	
PUSHF(D/Q)	'	3	4	p1 p4 p237 p06		1	
PUSHA(D)		11	19	P - P - P - 2 - P - 2 -		8	not 64 bit
POP	r	1	1	p23	2	0.5	
POP	stack pointer	3	3	p23 2p0156	_	3	
POP	m	2	3	2p237 p4		1	
POPF(D/Q)	***	9	9	Ζρ207 ρ-		20	
POPA(D)		18	18			8	not 64 bit
LAHF SAHF		10	1	p06	1	1	1101 04 111
SALC		3	3	3p0156	1	1	not 64 bit
LEA	r16,m	2	2	p1 p05	2-4	1	16 or 32 bit
LEA	110,111	2		p i pos	Z -4	ı	address size
LEA	r32/64,m	1	1	p15	1	0.5	1 or 2 compo- nents in address
LEA	r32/64,m	1	1	p1	3	1	3 components in address
LEA	r32/64,m	1	1	p1		1	rip relative address
BSWAP	r32	1	1	p15	1	0.5	addi 555
BSWAP	r64	2	2	p06 p15	2	1	
MOVBE	r16,m16	3	3	2p0156 p23	_	0.5-1	MOVBE
MOVBE	r32,m32	2	2	p15 p23		0.5	MOVBE
MOVBE	r64,m64	3	3	2p0156 p23		0.75	MOVBE
MOVBE	m16,r16	2	3	p06 p237 p4		1	MOVBE
MOVBE	m32,r32	2	3	p15 p237 p4		1	MOVBE
MOVBE	m64,r64	3	4	p06 p15 p237 p4		1	MOVBE
PREFETCHNTA/ 0/1/2	m	1	1	p23		0.5	
PREFETCHW	m	1	1	p23		1	PREFETCHW
LFENCE	""	2		none counted		4	I KEI ETOIW
MFENCE		4	4	p23 p4		33	
SFENCE		2	2	p23 p4 p23 p4		6	
Arithmetic in- structions							
ADD SUB	r,r/i	1	1	p0156	1	0.25	
ADD SUB	r,m	1	2	p0156 p23		0.5	
ADD SUB	m,r/i	2	4	2p0156 2p237 p4	5	1	
ADC SBB	r,r/i	1	1	p06	1	1	

400000	I						1
ADC SBB	r,m	2	2	p06 p23	_	1	
ADC SBB	m,r/i	4	6	3p0156 2p237 p4	5	2	
CMP	r,r/i	1	1	p0156	1	0.25	
CMP	m,r/i	1	2	p0156 p23	1	0.5	
INC DEC NEG NOT	r	1	1	p0156	1	0.25	
INC DEC NOT	m	3	4	p0156 2p237 p4	5-6	1	
NEG	m	2	4	p0156 2p237 p4	5-6	1	
AAA		2	2	p1 p56	4		not 64 bit
AAS		2	2	p1 p056	4		not 64 bit
DAA DAS		3	3	p1 2p056	4		not 64 bit
AAD		3	3	p1 2p056	4		not 64 bit
AAM		11	11	p0 p1 p5 p6	23	7	not 64 bit
MUL IMUL	r8	1	1	p1	3	1	
MUL IMUL	r16	4	4	p1 p0156	4	2	
MUL IMUL	r32	3	3	p1 p0156	4	1	
MUL IMUL	r64	2	2	p1 p6	3	1	
MUL IMUL	m8	1	2	p1 p23	_	1	
MUL IMUL	m16	4	_ 5	p1 3p0156 p23		2	
MUL IMUL	m32	3	4	p1 2p0156 p23		1	
MUL IMUL	m64	2	3	p1 p6 p23		1	
IMUL	r,r	1	1	p1 p1	3	1	
IMUL	r,m	1	2	p1 p23		1	
IMUL	r16,r16,i	2	2	p1 p0156	4	1	
IMUL	r32,r32,i	1	1	p1	3	1	
IMUL	r64,r64,i	1	1	p1	3	1	
IMUL	r16,m16,i	2	3	p1 p0156 p23	0	1	
IMUL	r32,m32,i	1	2	p1 p23		1	
IMUL	r64,m64,i	1	2	p1 p23		1	
MULX	r32,r32,r32	3	3	p1 2p056	4	1	BMI2
MULX	r32,r32,m32	3	4	p1 2p056 p23	4	1	BMI2
MULX	r64,r64,r64	2	2		4	1	BMI2
MULX	r64,r64,m64	2	3	p1 p5 p1 p6 p23	4	1	BMI2
DIV	r8	10	10	p0 p1 p5 p6	23	6	DIVIIZ
DIV	r16	10	10		23	6	
DIV	r32	10	10	p0 p1 p5 p6	23 26	6	
DIV	r64	36	36	p0 p1 p5 p6	35-88	21-83	
IDIV	r8	11	11	p0 p1 p5 p6	24	6	
IDIV	r16	10	10	p0 p1 p5 p6 p0 p1 p5 p6	23	6	
IDIV	r32	10	10		23 26	6	
IDIV	r64			p0 p1 p5 p6	42-95	24-90	
	104	57	57	p0 p1 p5 p6			
CBW		1	1	p0156	1	0.5	
CWDE		1	1	p0156	1	1	
CDQE		1	1	p0156	1	0.5	
CWD		2	2	p0156	1	1	
CDQ		1	1	p06	1	1	
CQO		1	1	p06	1	0.5	00515
POPCNT	r,r	1	1	p1	3	1	SSE4.2
POPCNT	r,m	1	2	p1 p23		1	SSE4.2
CRC32	r,r	1	1	p1	3	1	SSE4.2
CRC32	r,m	1	2	p1 p23		1	SSE4.2

Logic instruc-							
AND OR XOR	r,r/i	1	1	p0156	1	0.25	
AND OR XOR	r,m	1	2	p0156 p23	'	0.5	
AND OR XOR	m,r/i	2	4	2p0156 2p237 p4	5	1	
TEST	r,r/i	1	1	p0156	1	0.25	
TEST		1	2	p0156 p23	1	0.25	
SHR SHL SAR	m,r/i	1	1	l	1	0.5	
SHR SHL SAR	r,i		4	p06	ı		
	m,i	3		2p06 p237 p4	2	2 2	
SHR SHL SAR	r,cl		3	3p06			
SHR SHL SAR	m,cl	5	6	3p06 2p23 p4	_	4	- l u4 f - uu-
ROR ROL	r,1	2	2	2p06	1	1	short form
ROR ROL	r,i	1	1	p06	1	0.5	
ROR ROL	m,i	4	5	2p06 2p237 p4		2	
ROR ROL	r,cl	3	3	3p06	2	2	
ROR ROL	m,cl	5	6	3p06 p23 p4		4	
RCR RCL	r,1	3	3	2p06 p0156	2	2	
RCR RCL	m,1	4	6			3	
RCR RCL	r,i	8	8	p0156	6	6	
RCR RCL	m,i	11	11			6	
RCR RCL	r,cl	8	8	p0156	6	6	
RCR RCL	m,cl	11	11			6	
SHRD SHLD	r,r,i	1	1	p1	3	1	
SHRD SHLD	m,r,i	3	5			2	
SHLD	r,r,cl	4	4	p0156	3	2	
SHRD	r,r,cl	4	4	p0156	4	2	
SHRD SHLD	m,r,cl	5	7			4	
SHLX SHRX SARX	r,r,r	1	1	p06	1	0.5	BMI2
SHLX SHRX SARX	r,m,r	2	2	p06 p23		0.5	BMI2
RORX	r,r,i	1	1	p06	1	0.5	BMI2
RORX	r,m,i	2	2	p06 p23		0.5	BMI2
ВТ	r,r/i	1	1	p06	1	0.5	
ВТ	m,r	10	10			5	
ВТ	m,i	2	2	p06 p23		0.5	
BTR BTS BTC	r,r/i	1	1	p06	1	0.5	
BTR BTS BTC	m,r	10	11	•		5	
BTR BTS BTC	m,i	3	4	p06 p4 p23		1	
BSF BSR	r,r	1	1	p1	3	1	
BSF BSR	r,m	1	2	p1 p23		1	
SETcc	r	1	1	p06	1	0.5	
SETcc	m	2	3	p06 p237 p4		1	
CLC		1	0	none		0.25	
STC		1	1	p0156		0.25	
CMC		1	1	p0156	1	1	
CLD STD		3	3	p15 p6	'	4	
LZCNT	r,r	1	1	p1	3	1	LZCNT
LZCNT	r,m	1	2	p1 p23		1	LZCNT
TZCNT		1	1		3	1	BMI1
TZCNT	r,r	1	2	p1	٥	1	BMI1
ANDN	r,m	1	1	p1 p23	1	0.5	BMI1
ANDN	r,r,r	1	2	p15			
BLSI BLSMSK	r,r,m	1	1	p15 p23	1	0.5	BMI1
BLSI BLSMSK	r,r	1	'	p15	1	0.5	BMI1

BLSI BLSMSK BLSR	r,m	1	2	p15 p23		0.5	BMI1	
BEXTR	r,r,r	2	2	2p0156	2	0.5	BMI1	
BEXTR		3	3	2p0156 p23	2	0.5	BMI1	
BZHI	r,m,r	1	1	p15	1	0.5	BMI2	
BZHI	r,r,r			1 '	1		BMI2	
	r,m,r	1 1	2	p15 p23	0	0.5		
PDEP	r,r,r	1 1	1	p1	3	1	BMI2	
PDEP	r,r,m	1 1	2	p1 p23	•	1	BMI2	
PEXT	r,r,r	1 1	1	p1	3	1	BMI2	
PEXT	r,r,m	1	2	p1 p23		1	BMI2	
Control transfer i	nstructions							
JMP	short/near	1 1	1	p6		1-2		
JMP	r	1 1	1	p6		2		
JMP	m	1 1	2	p23 p6		2		
Conditional jump	short/near	1 1	1	p6		1-2	predicted	
							taken	
Conditional jump	short/near	1	1	p06		0.5-1	predicted not taken	
Fused arithmetic and branch		1	1	p6		1-2	predicted taken	
Fused arithmetic		1	1	p06		0.5-1	predicted not	
and branch				'			taken	
J(E/R)CXZ	short	2	2	p0156 p6		0.5-2		
LOOP	short	7	7	' '		5		
LOOP(N)E	short	11	11			6		
CALL	near	2	3	p237 p4 p6		3		
CALL	r	2	3	p237 p4 p6		2		
CALL	m	3	4	2p237 p4 p6		3		
RET	111	1	2			2		
	:			p237 p6				
RET	İ	4-	2			2		
BOUND	r,m	15	15			8	not 64 bit	
INTO		5	5			6	not 64 bit	
String instruc- tions								
LODSB/W		3	3	2p0156 p23		1		
LODSD/Q		2	2	p0156 p23				
REP LODS		4		p0130 p23		'		
			2	m00 m0456 m4		4		
STOS		3	3	p23 p0156 p4		1		
REP STOS		<2n				~0.5n	worst case	
REP STOS		2.6/32B				1/32B	best case aligned by 32	
MOVS		5	5	2p23 p4 2p0156		4		
REP MOVS		~2n				< 1n	worst case	
REP MOVS		4/32B				1/32B	best case	
							aligned by 32	
SCAS		3	3	p23 2p0156		1		
REP SCAS		≥6n				≥2n		
CMPS		5	5	2p23 3p0156		4		
REP CMPS		≥8n				≥2n		
Synchronization instructions								
XADD	m,r	4	5			5		
, , , , , ,	111,1	1 7 1		1				

LOCK XADD	m,r	9	9			18		
LOCK ADD	m,r	8	8			18		
CMPXCHG	m,r	5	6			6		
LOCK CMPXCHG	m,r	10	10			18		
CMPXCHG8B	m,r	16	16			11		
LOCK CMPXCHG8B	m,r	20	20			19		
CMPXCHG16B	m,r	23	23			16		
LOCK CMPXCHG16B	m,r	25	25			26		
Other								
NOP (90)		1	0	none		0.25		
Long NOP (0F		1	0	none		0.25		
1F)								
PAUSE		4	4	p6				
ENTER	a,0	12	12			8		
ENTER	a,b	~14+7b	~54+7b		~91+2b			
LEAVE		3	3	2p0156 p23		4		
XGETBV		15	15			7	XGETBV	
CPUID		38-120				100-250		
RDTSC		20	20			25		
RDTSCP		22	22			32	RDTSCP	
RDPMC		35	35			40		
RDRAND	r	16	16	p23 15p0156		~420	RDRAND	
RDSEED	r	16	16	p23 15p0156		~420	RDSEED	

Floating point x87 instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc-							
tions							
FLD	r	1	1	p05	1	0.5	
FLD	m32/64	1	1	p23	3	0.5	
FLD	m80	4	4	2p01 2p23	4	2	
FBLD	m80	43	43		46	22	
FST(P)	r	1	1	p05	1	0.5	
FST(P)	m32/m64	1	2	p4 p237	3	1	
FSTP	m80	7	7	3p0156 2p23 2p4	4	5	
FBSTP	m80	244	226		264	266	
FXCH	r	2	2	none	0	0.5	
FILD	m	1	2	p05 p23	5	1	
FIST(P)	m	3	3	p5 p23 p4	7	1	
FISTTP	m	3	3	p1 p23 p4	7	2	SSE3
FLDZ		1	1	p05		1	
FLD1		2	2	2p05		2	
FLDPI FLDL2E				·			
etc.		2	2	2p05		2	
FCMOVcc	r	4	4	p0 p1 p56	3	2	
FNSTSW	AX	2	2	p0 p0156	6	1	
FNSTSW	m16	2	3	p0 p4 p237	6	1	
FLDCW	m16	3	3	p01 p23 p6	7	2	
FNSTCW	m16	2	3	p237 p4 p6	6	1	

				i			
FINCSTP							
FDECSTP		1	1	p05	0	0.5	
FFREE(P)	r	1	1	p05		0.5	
FNSAVE	m	133	133		195	195	
FRSTOR	m	89	89		175	175	
Arithmetic in-							
structions							
FADD(P)							
FSUB(R)(P)	r	1	1	p5	3	1	
FADD(P)				-			
FSUB(R)(P)	m	2	3	p5 p23		1	
FMUL(P)	r	1	1	p0	5	1	
FMUL(P)	m	2	3	p0 p23		1	
FDIV(R)(P)	r	1	1	p0	14-16	4-5	
FDIV(R)(P)	m	1	2	p0 p23		4-5	
FABS		1	1	p0	1	1	
FCHS		1	1	p0	1	1	
FCOM(P) FUCOM	r	1	1	p5	3	1	
FCOM(P) FUCOM	m	1	2	p5 p23		1	
FCOMPP FU-			_	P P P = 0		•	
COMPP		2	2	p0 p5		1	
FCOMI(P)		_		1 1			
FUCOMI(P)	r	3	3	p5		1	
FIADD FISUB(R)	m	3	4	2p5 p23		2	
FIMUL `´	m	2	3	p0 p5 p23		1	
FIDIV(R)	m	2	3	p0 p5 p23			
FICOM(P)	m	2	3	2p5 p23		2	
FTST `		1	1	p5	3	1	
FXAM		2	2	2p5	6	2	
FPREM		31	31		26-30	15	
FPREM1		31	31		30-57	17	
FRNDINT		17	17		21	11	
						• •	
Math							
FSCALE		27	27		11		
FXTRACT		17	17		11	9	
FSQRT		1	1	p0	14-21	4-7	
FSIN		53-105	•		50-120		
FCOS		53-105			50-120		
FSINCOS		55-120			55-150		
F2XM1		40-90			65-80		
FYL2X		40-100			50-100		
FYL2XP1		56			55-80		
FPTAN		50-110			60-150		
FPATAN		30-110			100-150		
LIVION		30-100			100-100		
Other							
FNOP		1	1	p05		0.5	
WAIT		2	2	p05		2	
FNCLEX		5	5	p156		22	
FNINIT		18	18			78	

Integer vector instructions

integer vector			uona.			Recipro-	
		µops fused	µops unfused			cal through	
Instruction	Operands	domain	domain	μops each port	Latency	put	Comments
Move instruc-				proposition process			
tions							
MOVD	r32,mm/x	1	1	р0	2	1	
MOVD	m32,mm/x	1	2	p237 p4	3	1	
MOVD	mm/x,r32	1	1	p5	2	1	
MOVD	mm/x,m32	1	1	p23	2	0.5	
MOVQ	r64,mm/x	1	1	p0	2	1	
MOVQ	mm/x,r64	1	1	p5	2	1	
MOVQ	mm,mm	1		p05	1	0.5	
MOVQ	x,x	1		p015	1	0.33	
MOVQ	mm/x,m64	1	1	p23	2	0.5	
MOVQ	m64, mm/x	1	2	p237 p4	3	1	
MOVDQA/U	x,x	1 1	1	p015	0-1	0.25	may eliminate
MOVDQA/U	x, m128	1	1	p23	2	0.25	may ciiminate
MOVDQA/U	m128, x	1	2	p237 p4	3	1	
VMOVDQA/U	1	1	1	p015	0-1	0.25	may eliminate
VMOVDQA/U	y,y y,m256	1	1	p013	3	0.23	AVX
VMOVDQA/U	1 -	1	2		3	1	AVX
· ·	m256,y	1		p237 p4	3	1	SSE3
LDDQU	x, m128	1	1	p23		0.5	33E3
MOVDQ2Q	mm, x	2	2	p0 p5	2	1	
MOVQ2DQ	x,mm	2	2	p0 p15	2	1	
MOVNTQ	m64,mm	1	2	p237 p4	~420	1	
MOVNTDQ	m128,x	1	2	p237 p4	~400	1	
VMOVNTDQ	m256,y	1	2	p237 p4	~470	1	AVX2
MOVNTDQA	x, m128	2	2	p23 p015	2	0.5	SSE4.1
VMOVNTDQA	y,m256	2	2	p23 p015	3	0.5	AVX2
PACKSSWB/DW PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW				00 O F			
PACKUSWB PACKSSWB/DW	mm,m64	3	3	p23 2p5		2	
PACKUSWB	x,x / y,y,y	1	1	p5	1	1	
PACKSSWB/DW PACKUSWB	x,m / y,y,m	1	2	p23 p5		1	
PACKUSDW	x,x / y,y,y	1 1	1	p5	1	1 1	SSE4.1
PACKUSDW			2	· ·		1	SSE4.1
PUNPCKH/L	x,m / y,y,m	'		p23 p5		l I	33E4.1
BW/WD/DQ	v,v / v,v,v	1	1	p5	1	1	
PUNPCKH/L	V,V / V,V,V	'	!	ρυ		I I	
BW/WD/DQ	v,m / v,v,m	1	2	p23 p5		1	
PUNPCKH/L QDQ	x,x / y,y,y	1	1	p5	1	1	
PUNPCKH/L	,						
QDQ	x,m / y,y,m	1	2	p23 p5		1	
PMOVSX/ZX BW BD BQ DW DQ	x,x	1	1	p5	1	1	SSE4.1
PMOVSX/ZX BW BD BQ DW DQ	x,m	1	2	p23 p5		1	SSE4.1
VPMOVSX/ZX BW BD BQ DW DQ	y,x	1	1	p5	3	1	AVX2

VDMOVOVIZV DM	1 1		l	 			1
VPMOVSX/ZX BW BD BQ DW DQ	y,m		2	p5 p23		1	AVX2
PSHUFB	v,v / v,v,v	1	1	p5	1		SSSE3
PSHUFB	v,m / v,v,m	2	2	p23 p5	•	1 1	SSSE3
PSHUFW	mm,mm,i	1	1	p25 p5	1	1 1	00000
PSHUFW	mm,m64,i	2	2	p23 p5	ı	1 1	
PSHUFD		1	1		1	1 1	
	V,V,İ	2	2	p5	ı	1 1	
PSHUFD	v,m,i	1		p23 p5	4		
PSHUFL/HW	V,V,İ		1	p5	1		
PSHUFL/HW	v,m,i	2	2	p23 p5	4	1	00050
PALIGNR	v,v,i / v,v,v,i	1	1	p5	1	1 1	SSSE3
PALIGNR	v,m,i / v,v,m,i	2	2	p23 p5		1	SSSE3
PBLENDVB	x,x,xmm0	1	1	p015	1	1 1	SSE4.1
PBLENDVB	x,m,xmm0	2	2	p015 p23		1 1	SSE4.1
VPBLENDVB	V,V,V,V	2	2	2p015	2	1	AVX2
VPBLENDVB	v,v,m,v	3	3	2p015 p23		2	AVX2
PBLENDW	x,x,i / v,v,v,i	1	1	p5	1	1	SSE4.1
PBLENDW	x,m,i / v,v,m,i	2	2	p23 p5		1 1	SSE4.1
VPBLENDD	v,v,v,i	1	1	p015	1	0.33	AVX2
VPBLENDD	v,v,m,i	2	2	p015 p23		0.5	AVX2
VPERMD	y,y,y	1	1	p5	3	1	AVX2
VPERMD	y,y,m	1	2	p5 p23		1 1	AVX2
VPERMQ	y,y,i	1	1	p5	3	1	AVX2
VPERMQ	y,m,i	2	2	p5 p23		1	AVX2
VPERM2I128	y,y,y,i	1	1	p5	3	1 1	AVX2
VPERM2I128	y,y,m,i	2	2	p5 p23		1 1	AVX2
MASKMOVQ	mm,mm	4	4	p0 p4 2p23	~400	2	
MASKMOVDQU	x,x	10	10	4p04 2p56 4p23	~400	6	
VPMASKMOVD/Q	v,v,m	2	2	p23 p015	4	0.5	AVX2
VPMASKMOVD/Q	m,v,v	3	3	p0 p4 p23	~8	1	AVX2
PMOVMSKB	r,v	1	1	p0	2	1	,
PEXTRB/W/D/Q	r,x,i	2	2	p0 p5	3	1	SSE4.1
PEXTRB/W/D/Q	m,x,i	2	3	p23 p4 p5	ŭ	1	SSE4.1
VEXTRACTI128	x,y,i	1	1	p5	3	1	AVX2
VEXTRACTI128		2	2	p23 p4	4	1 1	AVX2
PINSRB	m,y,i x,r32,i	2	2	2p5	3	2	SSE4.1
PINSRB	x,n8,i	2	2	p23 p5	3	1	SSE4.1
PINSRW	mm/x,r32,i	2	2		3	2	33E4.1
PINSRW	mm/x,m16,i	2	2	p5	3	1	
PINSRD/Q		2	2	p23 p5	3	2	SSE4.1
	x,r32,i	2	2	2p5	3		SSE4.1 SSE4.1
PINSRD/Q	x,m32,i			p23 p5	2	1 1	
VINSERTI128	y,y,x,i	1	1	p5	3	1	AVX2
VINSERTI128	y,y,m,i	2	2	p015 p23	3	0.5	AVX2
VPBROADCAST B/W/D/Q	x,x	1	1	p5	1	1	AVX2
VPBROADCAST	^,^	Ī	!	ρυ	į	Į.	AVAZ
B/W	x,m8/16	2	2	p23 p5	7	1	AVX2
VPBROADCAST	74,6, 10	_	_	P=0 P0	·		,
D/Q	x,m32/64	1	1	p23	4	0.5	AVX2
VPBROADCAST							
B/W/D/Q	y,x	1	1	p5	3	1	AVX2
VPBROADCAST	0445	_			_		A) 0/6
B/W	y,m8/16	2	2	p23 p5	7	1	AVX2
VPBROADCAST	v m32/64	1	1	n22	3	0.5	AVX2
D/Q	y,m32/64	I	ļ I	p23	S	0.5	AVAZ

POSTHERDD X, [res*x], X 4 4 p0 p1 p23 p5 5 AVX2					1			,
PORTHEROD Y,[r+s*y],	VBROADCASTI128	y,m128	1	1	p23	3	0.5	AVX2
VPGATHERQD X, r=s* X , X 4 5 p0 p1 p23 p5 2 AVX2	VPGATHERDD	x,[r+s*x],x	4	4	p0 p1 p23 p5		4	AVX2
VPGATHERQD X, r=s* X , X 4 5 p0 p1 p23 p5 2 AVX2	VPGATHERDD	y,[r+s*y],y	4	4	p0 p1 p23 p5		5	AVX2
VPGATHEROD X, [r+s*y], X 4 4 5 p0 p1 p23 p5 2 AVX2	VPGATHERQD		4	5				AVX2
VPGATHERDQ X, [r+s*x], X 4 4 5 p0 p1 p23 p5 2 AVX2 VPGATHERQQ X, [r+s*x], Y 4 4 5 p0 p1 p23 p5 4 AVX2 VPGATHERQQ X, [r+s*x], Y 4 4 5 p0 p1 p23 p5 2 AVX2 VPGATHERQQ X, [r+s*x], Y 4 4 5 p0 p1 p23 p5 2 AVX2 AVX2 VPGATHERQQ X, [r+s*y], Y 4 4 4 p0 p1 p23 p5 4 AVX2 A	· ·							
VPGATHERQQ								
VPGATHERQQ X, [r+s*x], X 4 5 p0 p1 p23 p5 2 AVX2								
VPGATHERQQ V, [r+s*y], y 4								
Arithmetic instructions PADD/SUB(S,US) BW/D/O PADD/SUB BW/D/O Q PADD/SUB BW/D/O Q PADD/SUB BW/D/O Q PADD/SUB BW/D/O Q PADD/SUB BW/D/O Q PADD/SUB BW/D/O Q PADD/SUB BW/D/O Q V,v/v,v,v 1 1 1 p015 1 0.33 PADD/SUB BW/D/O Q V,v/v,v,v 1 1 1 p01 1 0.5 PADD/SUB BW/D/O PHADD(S)W/D PHADD(S)W/D PHADD(S)W/D PHADD(S)W/D PHADD(S)W/D PHADD(S)W/D PHSUB(S)W/D PHSUB(S)W/D PCMPEQBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERBW/D PCMPERQQ V,v/v,v,v 1 1 p01 1 0.5 PCMPERQQ V,v/v,v,v 1 1 p01 1 0.5 PCMPERQQ V,v/v,v,v 1 1 p01 1 0.5 PCMPERQQ V,v/v,v,v 1 1 p01 1 0.5 PCMPERQQ V,v/v,v,v 1 1 p01 1 0.5 PCMPERQQ V,v/v,v,v 1 1 p01 1 0.5 PCMPERQQ V,v/v,v,v 1 1 p01 5 0.5 PCMPERTO PMULL/HW PM								
Structions	VPGATHERQQ	y,[r+s~y],y	4	4	pu p i p23 p5		4	AVXZ
Structions								
PADDISUB(S, US)								
BW/ID/Q								
PADD/SUB B/W/D/Q PADD/SUB(S,US) B/W/D/Q PADD/SUBS/US B/M/D/Q PADD/SUBS/US B/M/D/C/C/C/C PADD/SUBS/US B/M/D/C/C/C/C PADD/SUBS/US B/M/D/C/C/C/C PADD/SUBS/US B/D/C/C/C/C/C/C PADD/SUBS/US B/M/D/C/C/C/C/C/C/C/C/C/C/C/C/C/C/C/C/C/C	PADD/SUB(S,US)	mm mm	4	1	205	1	0.5	
Q		111111,111111	1		pus	ľ	0.5	
PADD/SUB(S,US) B/W/D/Q V,m / v,v,m 1	1_	v v / v v v	1	1	n015	1	0.33	
BMVIDQ		v, v / v, v, v		'	poro		0.00	
PADD/SUBS/US B/ V,V / V,V,V		v.m / v.v.m	1	2	p015 p23		0.5	
W/D/Q		.,, .,.,		_	μοιο μ=ο		0.0	
PHADD(S)W/D		v,v / v,v,v	1	1	p01	1	0.5	
PHSUB(S)W/D PHADD(S)W/D PHSUB(S)(S)W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEGB/W/D PCMPGTB/W/D v,v/v,v,v 3 3 p01 2p5 p23 2 SSSE3 PCMPEQB/W/D PCMPEQB/W/D PCMPEGB/W/D PCMPGTB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQQ v,m/v,v,w 1 1 p0 1 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 1 1 p0 5 SSE4.1 p0 1 SSE4.2 p0 p0 5 SSE4.1 p0 5 1 p0 5 1 p0 5 1 p0 5 1 p0 5 1 p0 5 1 p0 5	PHADD(S)W/D				•			
PHADD(S)W/D PHSUB(S)W/D PCMPEQBW/D PCMPEQBW/D PCMPGTB/W/D PCMPGTB/W/D PCMPGTB/W/D PCMPGTB/W/D PCMPGTB/W/D PCMPGBW/D PCMPGBW/D PCMPGBW/D PCMPGBW/D PCMPGBW/D PCMPGBW/D PCMPGBW/D PCMPEQQ V,m/v,v,v 1 1 p0 1 2 2 1 2 3 3 3 3		v,v / v,v,v	3	3	p01 2p5	3	2	SSSE3
PHSUB(S)W/D PCMPEQBW/D PCMPEQBW/D PCMPEQBW/D PCMPGTBW/D PCMPGTBW/D PCMPEQBW/D PCMPEQBW/D PCMPEQBW/D PCMPEQBW/D PCMPEQBW/D PCMPGTBW/D X,x/y,y,y 1 1 p01 1 0.5 PCMPEQBW/D X,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PCMPEQQ V,v/v,v,v 1 1 p01 1 0.5 SSE4.1 PCMPEQQ V,v/v,v,v 1 1 p5 3 1 SSE4.2 PCMPGTQ V,v/v,v,m 1 2 p5 p23 1 SSE4.2 PCMPGTQ V,m/v,v,m 1 2 p5 p23 1 SSE4.2 PCMPGTQ V,m/v,v,m 1 2 p5 p23 1 SSE4.2 PMULL/HW PMULHUW mm,mm 1 1 p0 5 1 PMULL/HW PMULHUW X,x/y,y,y 1 1 p01 5 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,x/y,y,y 1 1 p01 5 0.5 SSSE3 PMULHRSW x,x/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/	` '							
PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQQ V,V/v,v 1		v,m / v,v,m	4	4	p01 2p5 p23		2	SSSE3
PCMPGTB/W/D PCMPEQB/W/D PCMPGTB/W/D PCMPGTB/W/D PCMPGTB/W/D PCMPGTB/W/D PCMPGTB/W/D PCMPGDQ PCMPEQQ PCMPEQQ PCMPEQQ V,v/v,v,v 1 1 p01 1 1 PCMPEQB/W/D PCMPGTB/W/D PCMPEQQ PCMPEQQ V,v/v,v,v 1 1 p01 p23 0.5 SSE4.1 PCMPEQQ PCMPGTQ PCMPGTQ PCMPGTQ PMULL/HW PMULL/HW PMULL/HW PMULLHW PMULLHW PMULLHW PMULLHW PMULLHW PMULHRSW PMULHRSW PMULHRSW PMULHRSW PMULHRSW Mm,mm 1 1 p01 p23 0.5 SSE4.2 PMULHRSW PMULHRSW PMULHRSW PMULHRSW PMULLD PMULDQ X,x/y,y,y 1 1 p01 p23 0.5 SSE3 PMULLD PMULDQ X,x/y,y,y 1 1 p01 p23 0.5 SSE3 PMULDQ X,x/y,y,y 2 2 p01 p23 0.5 SSE3 PMULDDQ X,x/y,y,y 1 1 p01 p23 0.5 SSE4.1 PMULDQ X,x/y,y,y 1 1 p01 p23 1 SSE4.1 PMULDQ X,x/y,y,y 1 1 p01 p23 0.5 SSE4.1 PMULDQ X,x/y,y,y 1 1 p01 p23 0.5 SSE4.1 PMULUDQ PMULUDQ X,x/y,y,y 1 1 p01 p23	` '							
PCMPEQB/W/D PCMPGTB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQB/W/D PCMPEQQ x,x/y,y,y 1 1 p01 1 0.5 PCMPEQB/W/D PCMPEQB/W/D PCMPEQQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PCMPEQQ v,m/v,v,m 1 2 p01 p23 0.5 SSE4.1 PCMPEQQ v,m/v,v,m 1 2 p01 p23 0.5 SSE4.1 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PMULL/HW PMULLHUW mm,mm 1 1 p0 5 1 PMULLHRSW x,m/y,y,m 1 2 p01 p23 0.5 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSE3 PMULDR x,x/y,y,y 2 2 2p01 p23 0.5 SSE4.1 PMULDD x,x/y,y,y 1 2 p01 p23 0.5 SSE4.1 PMU		mm,mm	1	1	p0	1	1	
PCMPGTB/W/D PCMPEQB/W/D PCMPGTB/W/D PCMPGTB/W/D PCMPGTB/W/D x,x/y,y,y 1 1 p01 1 0.5 PCMPEQB/W/D PCMPGTB/W/D PCMPEQQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PCMPEQQ v,v/v,v,v 1 1 p01 1 0.5 SSE4.1 PCMPGTQ v,v/v,v,v 1 1 p5 3 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PMULL/HW PMULL/HW PMULHUW mm,mm 1 1 p0 5 1 PMULHRSW mm,mm 1 1 p01 5 0.5 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSE3 PMULLD x,x/y,y,y 1 1 p01 5 1 SSE3 PMULLD x,x/y,y,y 1 1 p01 5 0.5 SSE3 PMULDQ x,x/y,y,y 1 1 p01 5 0.		,			•			
PCMPEQB/W/D PCMPETB/WID x,m/y,y,m 1 2 p01 p23 0.5 PCMPEQQ PCMPEQQ v,v/v,v,v 1 1 p01 1 0.5 SSE4.1 PCMPEQQ PCMPETQ v,w/v,v,v 1 1 p01 p23 0.5 SSE4.1 PCMPGTQ v,w/v,v,v 1 1 p5 3 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PMULL/HW PMULHUW mm,mm 1 1 p0 5 1 PMULLHWW PMULHUW x,x/y,y,y 1 1 p01 5 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 1 1 p01 5 0.5 SSSE3 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDQ <td></td> <td>x,x / y,y,y</td> <td>1</td> <td>1</td> <td>p01</td> <td>1</td> <td>0.5</td> <td></td>		x,x / y,y,y	1	1	p01	1	0.5	
PCMPGTB/W/D x,m/y,y,m 1 2 p01 p23 0.5 PCMPEQQ v,v/v,v,v 1 1 p01 1 0.5 SSE4.1 PCMPEQQ v,m/v,v,m 1 2 p01 p23 0.5 SSE4.1 PCMPGTQ v,v/v,v,v 1 1 p5 3 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PCMULHW mm,mm 1 1 p0 5 1 SSE4.2 PMULLHW x,x/y,y,y 1 1 p01 5 0.5 SSSE3 PMULHRSW mm,mm 1 1 p01 p23 0.5 SSSE3 PMULLD x,m/y,y,m	PCMPEQB/W/D	, 3,3,5			•			
PCMPEQQ v,v/v,v,w 1 1 p01 1 0.5 SSE4.1 PCMPEQQ v,m/v,v,m 1 2 p01 p23 0.5 SSE4.1 PCMPGTQ v,v/v,v,w 1 1 p5 3 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PMULL/HW PMULL/HW mm,mm 1 1 p0 5 1 PMULL/HW pMULLHUW x,x/y,y,y 1 1 p01 5 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 1 1 p01 p23 0.5 SSE4.1 PMULDD x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1		x,m / y,y,m	1	2	p01 p23		0.5	
PCMPEQQ v,m/v,v,m 1 2 p01 p23 0.5 SSE4.1 PCMPGTQ v,v/v,v,v 1 1 p5 3 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PMULL/HW PMULL/HW pMULL/HW p00 5 1 PMULL/HW pMULHUW x,m/y,y,m 1 2 p01 p23 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULDQ x,m/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m	PCMPEQQ	v,v / v,v,v	1	1		1	0.5	SSE4.1
PCMPGTQ v,v/v,v,v 1 1 p5 3 1 SSE4.2 PCMPGTQ v,m/v,v,m 1 2 p5 p23 1 SSE4.2 PMULL/HW mm,mm 1 1 p0 5 1 PMULL/HW pMULHUW x,x/y,y,y 1 1 p01 5 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULD x,x/y,y,y 1 1 p01 5 0.5 SSE3 PMULDQ x,m/y,y,m 3 2p01 p23 1 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y	PCMPEQQ	v,m / v,v,m	1	2	p01 p23		0.5	SSE4.1
PCMPGTQ PMULL/HW PMULHUW v,m / v,v,m 1 2 p5 p23 1 SSE4.2 PMULL/HW PMULHUW mm,mm 1 1 p0 5 1 PMULL/HW PMULHUW x,x / y,y,y 1 1 p01 5 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSE3 PMULHRSW mm,mm 1 1 p01 5 0.5 SSSE3 PMULHRSW x,x / y,y,y 1 1 p01 5 0.5 SSSE3 PMULLD x,x / y,y,y 1 2 p01 p23 0.5 SSE4.1 PMULDD x,x / y,y,m 1 2 p01 p23 1 SSE4.1 PMULDQ x,x / y,y,y 1 1 p01 5 0.5 SSE4.1 PMULUDQ x,m / y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1	PCMPGTQ		1			3	1 1	SSE4.2
PMULL/HW PMULHUW mm,mm 1 1 p0 5 1 PMULL/HW PMULHUW x,x/y,y,y 1 1 p01 5 0.5 PMULL/HW PMULHRW x,m/y,y,m 1 2 p01 p23 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULDD x,m/y,y,m 3 2p01 p23 1 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 D.5 PMADDWD x,m/y,y,m 1 2	· ·		1				1	
PMULHUW mm,mm 1 1 p0 5 1 PMULL/HW x,x/y,y,y 1 1 p01 5 0.5 PMULHUW x,m/y,y,m 1 2 p01 p23 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,x/y,y,y 1 1 p01 5 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULDD x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y		.,, .,.,		_	P			
PMULL/HW PMULHUW x,x/y,y,y 1 1 p01 5 0.5 PMULL/HW PMULHUW x,m/y,y,m 1 2 p01 p23 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,x/y,y,y 1 1 p01 5 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULDD x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD <t< td=""><td>- '</td><td>mm.mm</td><td>1</td><td>1</td><td>0g</td><td>5</td><td>1</td><td></td></t<>	- '	mm.mm	1	1	0g	5	1	
PMULHUW x,x/y,y,y 1 1 p01 5 0.5 PMULL/HW x,m/y,y,m 1 2 p01 p23 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,x/y,y,y 1 1 p01 5 0.5 SSSE3 PMULLD x,x/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULDD x,x/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,m/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD x,x/y,y,y 1 1 <		,			1 -			
PMULL/HW PMULHUW x,m/y,y,m 1 2 p01 p23 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,x/y,y,y 1 1 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 p23 1 SSE4.1 PMULDD x,m/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,m/y,y,y 1 1 p01 p23 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ mm,mm 1 1 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMADDWD mm,mm 1 1 p01 p23 0.5 SSE4.1 PMADDWD x,m/y,y,m 1 2 <td></td> <td>x.x / v.v.v</td> <td>1</td> <td>1</td> <td>p01</td> <td>5</td> <td>0.5</td> <td></td>		x.x / v.v.v	1	1	p01	5	0.5	
PMULHUW x,m/y,y,m 1 2 p01 p23 0.5 PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,x/y,y,y 1 1 p01 p23 0.5 SSSE3 PMULHD x,x/y,y,y 2 2 2p01 p23 1 SSE4.1 PMULLD x,x/y,y,y 1 1 p01 p23 1 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 p23 0.5 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 p23 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ x,x/y,y,y 1 1 p01 p2 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 D PMADDWD mm,mm 1 1 p01 p2 5 1 PMADDWD x,m/y,y,m 1 2 p01 p		, . , , , , ,						
PMULHRSW mm,mm 1 1 p0 5 1 SSSE3 PMULHRSW x,x/y,y,y 1 1 p01 5 0.5 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULDQ x,m/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ mm,mm 1 1 p0 5 1 PMULUDQ x,x/y,y,y 1 1 p01 p23 0.5 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 p23 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5		x.m / v.v.m	1	2	p01 p23		0.5	
PMULHRSW x,x/y,y,m 1 1 p01 5 0.5 SSSE3 PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULDD x,m/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ mm,mm 1 1 p01 5 0.5 PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD x,m/y,y,m 1 2						5		SSSE3
PMULHRSW x,m/y,y,m 1 2 p01 p23 0.5 SSSE3 PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULDD x,m/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ mm,mm 1 1 p0 5 1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDWB x,m/y,y,m 1 2 p01 p23 0.5 PMADDWB mm,mm 1 1 p01 p23 0.5					·		0.5	
PMULLD x,x/y,y,y 2 2 2p01 10 1 SSE4.1 PMULLD x,m/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ mm,mm 1 1 p0 5 1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDWB mm,mm 1 1 p01 p23 0.5 PMADDWB mm,mm 1 1 p01 p23 0.5					· ·			
PMULLD x,m/y,y,m 3 3 2p01 p23 1 SSE4.1 PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ mm,mm 1 1 p0 5 1 PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1						10		
PMULDQ x,x/y,y,y 1 1 p01 5 0.5 SSE4.1 PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ mm,mm 1 1 p0 5 1 PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1			I		•	10		
PMULDQ x,m/y,y,m 1 2 p01 p23 0.5 SSE4.1 PMULUDQ mm,mm 1 1 p0 5 1 PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3						5		
PMULUDQ mm,mm 1 1 p0 5 1 PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3						5		
PMULUDQ x,x/y,y,y 1 1 p01 5 0.5 PMULUDQ x,m/y,y,m 1 2 p01 p23 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x/y,y,y 1 1 p01 5 0.5 PMADDWD x,m/y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3						_		55E4.1
PMULUDQ x,m / y,y,m 1 2 p01 p23 0.5 PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x / y,y,y 1 1 p01 5 0.5 PMADDWD x,m / y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3								
PMADDWD mm,mm 1 1 p0 5 1 PMADDWD x,x / y,y,y 1 1 p01 5 0.5 PMADDWD x,m / y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3					· ·	5		
PMADDWD x,x / y,y,y 1 1 p01 5 0.5 PMADDWD x,m / y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3		x,m / y,y,m	1		p01 p23		0.5	
PMADDWD x,m / y,y,m 1 2 p01 p23 0.5 PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3			1		· ·			
PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3			1			5	0.5	
PMADDUBSW mm,mm 1 1 p0 5 1 SSSE3	PMADDWD	x,m / y,y,m	1	2	p01 p23		0.5	
	PMADDUBSW		1	1		5	1	SSSE3
	PMADDUBSW	x,x / y,y,y	1	1	·		0.5	SSSE3

1	1		1	ı	1	1	ı	
PMADDUBSW	x,m / y,y,m	1	2	p01 p23		0.5	SSSE3	
PAVGB/W	mm,mm	1	1	p0	1	1		
PAVGB/W	x,x / y,y,y	1	1	p01	1	0.5		
PAVGB/W	x,m / y,y,m	1	2	p01 p23		0.5		
PMIN/PMAX	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	•	_	μο. μ=σ				
SB/SW/SD								
UB/UW/UD	mm,mm	1	1	р0	1	1	SSE4.1	
PMIN/PMAX				Po	•	'	0024.1	
SB/SW/SD								
UB/UW/UD	x,x / y,y,y	1	1	p01	1	0.5	SSE4.1	
PMIN/PMAX	^,^ / y,y,y			ροι	'	0.5	00L4.1	
SB/SW/SD								
UB/UW/UD	x,m / y,y,m	1	2	p01 p23		0.5	SSE4.1	
PHMINPOSUW		1	1		4		SSE4.1	
	X,X			p0	4	1		
PHMINPOSUW	x,m128	1	2	p0 p23		1	SSE4.1	
PABSB/W/D	mm,mm	1	1	p0	1	1	SSSE3	
PABSB/W/D	x,x / y,y	1	1	p01	1	0.5	SSSE3	
PABSB/W/D	x,m / y,m	1	2	p01 p23		0.5	SSSE3	
PSIGNB/W/D	mm,mm	1	1	р0	1	1	SSSE3	
PSIGNB/W/D	x,x / y,y,y	1	1	p01	1	0.5	SSSE3	
PSIGNB/W/D	x,m / y,y,m	1	2	p01 p23		0.5	SSSE3	
PSADBW	v,v / v,v,v	1	1	p51 p20	3	1	00020	
PSADBW			2					
	v,m / v,v,m	1	1	p5 p23		1	00544	
MPSADBW	x,x,i / v,v,v,i	2	2	2p5	4	2	SSE4.1	
MPSADBW	x,m,i / v,v,m,i	3	3	2p5 p23		2	SSE4.1	
Logic instruc-								
tions								
PAND PANDN								
POR PXOR	mm,mm	1	1	p05	1	0.5		
PAND PANDN								
POR PXOR	x,x / y,y,y	1	1	p015	1	0.33		
PAND PANDN								
POR PXOR	v,m / v,v,m	1	2	p015 p23		0.5		
PTEST	V,V	2	2	p0 p5	3	1	SSE4.1	
PTEST	v,m	2	3	p0 p5 p23		1	SSE4.1	
PSLLW/D/Q				P				
PSRLW/D/Q								
PSRAW/D/Q	mm,mm	1	1	р0	1	1		
PSLLW/D/Q		•						
PSRLW/D/Q								
PSRAW/D/Q	mm,m64	2	2	p0 p23		1		
PSLLW/D/Q	,	_	_	PO P-0				
PSRLW/D/Q								
PSRAW/D/Q	x,x / v,v,x	2	2	p01 p5	1	1		
PSLLW/D/Q	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	_	_	ρο. ρο				
PSRLW/D/Q								
PSRAW/D/Q	x,m / v,v,m	2	2	p01 p23		0.5		
PSLLW/D/Q	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	_	_	ρο: μ2ο		0.0		
PSRLW/D/Q								
PSRAW/D/Q	mm,i	1	1	p0	1	1		
PSLLW/D/Q	''''',1	'		ρυ	'	'		
PSRLW/D/Q								
PSRAW/D/Q	x,i / y,y,i	1	1	p01	1	0.5		
	^,ı / y,y,ı	ı	1	μοι	'	0.5	l	I

VPSLLVD/Q							
VPSRAVD							
VPSRLVD/Q	V,V,V	2	2	p01 p5	2	1	AVX2
VPSLLVD/Q							
VPSRAVD				04 00		0.5	11.00
VPSRLVD/Q	v,v,m	2	2	p01 p23		0.5	AVX2
PSLLDQ PSRLDQ	v: / v v :	1	4	, F	4	4	
PSRLDQ	x,i / v,v,i	1	1	p5	1	1	
String instruc- tions							
PCMPESTRI	x,x,i	8	8	6p05 2p16	4	4	SSE4.2
PCMPESTRI	x,m128,i	8	8	3p0 2p16 2p5 p23		4	SSE4.2
PCMPESTRM	x,x,i	9	9	3p0 2p16 4p5	9	9	SSE4.2
PCMPESTRM	x,m128,i	9	9	6p05 2p16 p23		5	SSE4.2
PCMPISTRI	x,x,i	3	3	3p0	3	3	SSE4.2
PCMPISTRI	x,m128,i	4	4	3p0 p23		3	SSE4.2
PCMPISTRM	x,x,i	3	3	3p0	9	9	SSE4.2
PCMPISTRM	x,m128,i	4	4	3p0 p23		3	SSE4.2
Encryption instru	uctions						
PCLMULQDQ	x,x,i	1	1	p5	7	1	CLMUL
PCLMULQDQ	x,m,i	2	2	p5 p23		1	CLMUL
AESDEC,							
AESDECLAST,							
AESENC,							
AESENCLAST	X,X	1	1	p0	4	1	AES
AESDEC,							
AESDECLAST,							
AESENC, AESENCLAST	x,m	2	2	p0 p23		1.5	AES
AESIMC	X,111 X,X	2	2	2p0	8	2	AES
AESIMC		3	3	2p0 p23	O	2	AES
AESKEYGENAS	x,m	3	3	Ζρυ μ23			ALS
SIST	x,x,i	13	13	p0 p5	12	12	AES
AESKEYGENAS	۸,۸,۱	'0		ρο ρο	12	12	ALO
SIST	x,m,i	13	13			12	AES
	,,,,,,					. <u> </u>	,
Other							
EMMS		10	10	p05		6	

Floating point XMM and YMM instructions

Instruction	Operands	μορs fused domain	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instruc- tions							
MOVAPS/D	X,X	1	1	p015	0-1	0.25	may eliminate
VMOVAPS/D	y,y	1	1	p015	0-1	0.25	may eliminate
MOVAPS/D MOVUPS/D	x,m128	1	1	p23	2	0.5	
VMOVAPS/D VMOVUPS/D MOVAPS/D	y,m256	1	1	p23	3	0.5	AVX
MOVAPS/D	m128,x	1	2	p237 p4	3	1	

\text{VMOVUPS/ID} \text{ m256.y} \text{ 1 } 2 \text{ p237 p4 } 3 \text{ 1 } 1 \text{ p5 } 1 \text{ 1 } 1 \text{ m0VSS/D} \text{ m32/64 } 1 \text{ 1 } 1 \text{ p5 } 2 \text{ 2 } 0.5 \text{ m0VSS/D} \text{ m32/64 x } 1 \text{ 2 } p237 p4 \text{ 3 } 1 \text{ 1 } 1 \text{ m0VSS/D} \text{ m32/64 x } 1 \text{ 2 } p237 p4 \text{ 3 } 1 \text{ 1 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p23 p5 \text{ 4 } 1 \text{ 1 } \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 3 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 3 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 3 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 3 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 3 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 3 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 3 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 3 } 1 \text{ m0VHPS/D} \text{ m64 x } 1 \text{ 2 } p4 p237 \text{ 400 } 1 \text{ m0VHPS/D} \text{ m258, y } 1 \text{ 2 } p4 p237 \text{ 400 } 1 \text{ m0VMOVMS/PS/D} \text{ m258, y } 1 \text{ 2 } p4 p237 \text{ 400 } 1 \text{ m4VX} \text{ m0VMOVMSPS/D} \text{ m258, y } 1 \text{ 2 } p4 p237 \text{ 400 } 1 \text{ m4VX} \text{ m14 p5 } 1 \text{ 1 } 1 \text{ p5 } 1 \text{ 1 } 1 m5 m2 m2 m2 m2 m2 m2 m2 m2 m2 m2 m2 m2 m2	h	I	I	ı	l	I	ı	l I
MOVSS/D x,x 1 1 p5 1 1 1 MOVSS/D x,m32/64,x 1 1 p23 2 0.5 MOVHPS/D x,m64 1 2 p23 p5 4 1 MOVHPS/D m64,x 1 2 p23 p5 4 1 MOVLPS/D m64,x 1 2 p4 p237 3 1 MOVHPS/D m23,x 1 1 p6 1 1 MOVNTPS/D m128,x 1 1 p0 4 1 VMOVMSKPS/D m22,y 1 1 p0 4 1 VMOVNTPS/D m128,x 1 2 p4 p237 ~400 1 AVX <td>VMOVAPS/D</td> <td>m256 v</td> <td>4</td> <td>2</td> <td>n227 n4</td> <td>2</td> <td>4</td> <td>A\/\</td>	VMOVAPS/D	m256 v	4	2	n227 n4	2	4	A\/\
MOVSS/D x,m32/64 1 1 p23 2 0.5 MOVHPS/D x,m64 1 2 p237 p4 3 1 MOVHPS/D m64,x 1 2 p223 p5 4 1 MOVLPS/D m64,x 1 2 p4 p237 3 1 MOVLPS/D m64,x 1 2 p4 p237 3 1 MOVLPS/D m64,x 1 2 p4 p237 3 1 MOVHPS/D m64,x 1 2 p4 p237 3 1 MOVHPS/D x,x 1 p5 1 1 MOVMSKPS/D r32,x 1 1 p0 2 1 MOVNTPS/D m128,x 1 2 p4 p237 ~400 1 AVX VMOVMSKPS/D m25b,y 1 2 p4 p237 ~400 1 AVX VMOVNTPS/D m25b,y 1 2 p5 p23 1 1			· ·				-	AVA
MOVSS/D m32/64.x 1 2 p23 p5 4 1 MOVHPS/D m64.x 1 2 p23 p5 4 1 MOVLPS/D m64.x 1 2 p4 p237 3 1 MOVLPS/D m64.x 1 2 p4 p237 3 1 MOVLPS/D m64.x 1 2 p4 p237 3 1 MOVLPS/D m64.x 1 2 p4 p237 3 1 MOVHPS/D m28.x 1 1 p5 1 1 MOVMSKPS/D m32.x 1 1 p0 4 1 VMOVMSKPS/D m256.y 1 2 p4 p237 ~400 1 VMOVNTPS/D m128.x 1 2 p4 p237 ~400 1 AVX SHUFPS/D m256.y 1 2 p4 p237 ~400 1 AVX VPERMILPS/PD v.x.i/v.v.mi 1 p5 1			· •					
MOVHPS/ID				-				
MOVHPS/ID m64 x m64 x m64 movtles/id 1 movtles/id 2 m64 x m64 movtles/id 3 m64 x mm0 med x movtles/id 1 movtles/id 3 movtles/id 4 movt		· ·						
MOVLPS/ID x,m64 1 2 p23 p5 4 1 1 AVX <td></td> <td>· ·</td> <td></td> <td></td> <td></td> <td></td> <td>•</td> <td></td>		· ·					•	
MOVLPS/D m64,x 1 2 p4 p237 3 1 MOVHLPS x,x 1 1 p5 1 1 MOVMSKPS/D r32,x 1 1 p5 1 1 MOVMSKPS/D r32,y 1 1 p0 4 1 MOVNTPS/D m128,x 1 2 p4 p237 ~400 1 MOVNTPS/D m256,y 1 2 p4 p237 ~400 1 SHUFPS/D x,xi,i /v,v,vi 1 1 p6 1 1 VPERMILPS/PD v,xi 1 1 p5 1 1 AVX VPERMILPS/PD v,v,v 1 1 p5 1 1 AVX VPERMILPS/PD v,v,v 1 1 p5 1 1 AVX VPERMILPS/PD v,v,v 1 1 p5 3 1 AVX VPERMBP128 y,y,y,i 1 1								
MOVHLPS x,x 1 1 p5 1 1 1 MOVHSKPS/D r32,x 1 1 p0 2 1 VMOWMSKPS/D r32,x 1 1 p0 2 1 VMOVNTPS/D m128,x 1 2 p4 p237 ~400 1 SHUFPS/D x,x,i / v,v,v,i 1 1 p5 1 1 SHUFPS/D x,x,i / v,v,v,i 1 1 p5 1 1 VPERMILPS/PD v,v,i 1 1 p5 1 1 AVX VPERMILPS/PD v,v,v 1 1 p5 1 1 AVX VPERMILPS/PD v,v,m 2 2 p5 p23 1 AVX VPERMILPS/PD v,v,m 1 1 p5 1 1 AVX VPERMILPS/PD v,v,m 2 2 p5 p23 1 AVX VPERMILPS/PD y,y,y,m 1 <t< td=""><td></td><td>· ·</td><td> 1</td><td></td><td></td><td></td><td> 1</td><td></td></t<>		· ·	1				1	
MOVLHPS x,x 1 1 p5 1 1 1 MOVMSKPS/D f32,x 1 1 p0 2 1 VMOVMSKPS/D m25,y 1 1 p0 4 1 MOVMTPS/D m128,x 1 2 p4 p237 ~400 1 AVX SHUFPS/D x,x,i /v,v,v,i 1 1 p6 1 1 AVX SHUFPS/D x,x,i /v,v,v,i 1 1 p5 1 1 AVX VPERMILPS/PD v,v,i 1 1 p5 1 1 AVX VPERMILPS/PD v,v,i 1 1 p5 1 1 AVX VPERMILPS/PD v,v,v 1 1 p5 1 1 AVX VPERMILPS/PD v,v,v 1 1 p5 3 1 AVX VPERMILPS/PD v,v,v 1 1 p5 3 1 AVX VPERMILPS/PD v,v,v 1 1 p5 3 1 AVX VP		m64,x	1			3	1	
MOVMSKPS/D r32,x 1 1 p0 2 1 4 MOVMONSKPS/D MOVMTPS/D r32,y 1 1 p0 4 1 MOVMTPS/D m268,x 1 2 p4 p237 ~400 1 AVX SHUFPS/D x,x,i /v,v,v,i 1 2 p4 p237 ~400 1 AVX SHUFPS/D x,x,i /v,v,v,i 1 1 p5 1 1 SSED VPERMILPS/PD v,v,i 1 1 p5 1 1 AVX VPERMILPS/PD v,v,m,i 2 2 p5 p23 1 AVX VPERMILPS/PD v,v,m,i 2 2 p5 p23 1 AVX VPERMILPS/PD v,v,m,i 2 2 p5 p23 1 AVX VPERMILPS/PD v,v,m,i 2 2 p5 p23 1 AVX VPERMPS y,y,y,m 1 1 p5 3 1 AVX VPERMPS <td></td> <td>X,X</td> <td> 1</td> <td>1</td> <td>p5</td> <td>1</td> <td>1</td> <td></td>		X,X	1	1	p5	1	1	
\text{VMOVMSKPS/D} \text{VMOVMTPS/D} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text{VMID} \text{VMOVMTPS/D} \text{VMID} \text	MOVLHPS	X,X	1	1	p5	1	1	
MOVNTPS/D	MOVMSKPS/D	r32,x	1	1	p0	2	1	
VMOVNTPS/D SHUFPS/D SHUFPS/D VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VV,V,V 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	VMOVMSKPS/D	r32,y	1	1	p0	4	1	
SHUFPS/D SHUFPS/D SHUFPS/D SHUFPS/D SHUFPS/D SHUFPS/D VV,V,I 1 1 1 p5 1 1 1 AVX VPERMILPS/PD VV,V,I VPERMILPS/PD V,V,I 1 1 1 p5 1 1 1 AVX VPERMILPS/PD V,V,W VPERMILPS/PD V,V,W 1 1 1 p5 1 1 AVX VPERMILPS/PD V,V,W VPERMILPS/PD V,V,W 1 1 1 p5 3 1 AVX VPERMILPS/PD VPERMILPS/PD V,V,W 1 1 1 p5 3 1 AVX VPERMIPS/PD VPERMIPS/D VPERMIPS/PD SHUPS/PD SHUPS/PD SHUPS/PD SHUPS/PD SHUPS/PD SHUPS/PD SHUPS/PD SHUPS/PD SHUPS/PD SHUPS/PD SHUPS/PD V,V,V,V,V,V,V,V,V,V,V,V,V,V,V,V,V,V,V,	MOVNTPS/D	m128,x	1	2	p4 p237	~400	1	
SHUFPS/ID X,m,i / v,v,m,i 2 2 p5 p23 1 AVX	VMOVNTPS/D	m256,y	1	2	p4 p237	~400	1	AVX
SHUFPS/ID X,m,i / v,v,m,i 2 2 p5 p23 1 AVX	SHUFPS/D	x,x,i / v,v,v,i	1	1	p5	1	1	
VPERMILPS/PD v,v,i 1 1 p5 p23 1 1 AVX VPERMILPS/PD v,m,i 2 2 p5 p23 1 AVX VPERMILPS/PD v,v,v 1 1 p5 1 1 AVX VPERMSP128 y,y,y,i 1 1 p5 3 1 AVX VPERMSP128 y,y,m,i 2 2 p5 p23 1 AVX VPERMPS y,y,m,i 2 2 p5 p23 1 AVX VPERMPS y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,m,i 2 2 p5 p23 1 AVX2 VPERMPD y,m,i 1 1 p015 1 0.33 SSE4.1 BLENDVPS/PD	SHUFPS/D		2	2			1	
VPERMILPS/PD V,m,i 2 2 p5 p23 1 AVX VPERMILPS/PD V,v,v 1 1 p5 1 1 AVX VPERMILPS/PD V,v,m 2 2 p5 p23 1 AVX VPERM2F128 Y,y,y,mi 2 2 p5 p23 1 AVX VPERMPS Y,y,y,m 1 2 p5 p23 1 AVX VPERMPB Y,y,m 1 2 p5 p23 1 AVX2 VPERMPD Y,y,i 1 1 p5 3 1 AVX2 VPERMPD Y,mi 2 2 p5 p23 1 AVX2 VPERMPD Y,mi 2 2 p5 p23 1 AVX2 VPERMPD Y,mi 1 1 p015 1 0.33 SSE4.1 BLENDPS/PD X,x,i /v,v,i 1 1 p015 1 0.33 SSE4.1 BLENDVPS/PD X,x,i X,	VPERMILPS/PD		1	1		1	1	AVX
VPERMILPS/PD V,V,V 1 1 p5 1 1 AVX VPERMZF128 V,V,M 2 2 p5 p23 1 AVX VPERMZF128 Y,Y,y,J 1 1 p5 3 1 AVX VPERMPS Y,Y,J 1 1 p5 3 1 AVX2 VPERMPS Y,Y,J 1 1 p5 3 1 AVX2 VPERMPD Y,Y,I 1 1 p5 3 1 AVX2 VPERMPD Y,Y,I 1 1 p5 3 1 AVX2 VPERMPD Y,Y,II 1 1 p5 3 1 AVX2 VPERMPD Y,Y,II 1 1 p5 3 1 AVX2 VPERMPD X,X,I/V,V,V,II 1 1 p015 1 0.33 SSE4.1 BLENDVPS/PD X,X,xmm0 1 1 p015 1 1 SSE4.1<			2	2	•		1	
VPERMILPS/PD V,v,m 2 2 p5 p23 1 AVX VPERMZF128 y,y,y,i 1 1 p5 3 1 AVX VPERMPS y,y,mi 2 2 p5 p23 1 AVX VPERMPS y,y,y 1 1 p5 3 1 AVX2 VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,mi 1 2 p5 p23 1 AVX2 VPERMPD y,mi 1 1 p5 3 1 AVX2 VPERMPD y,mi 1 1 p015 1 0.33 SSE4.1 BLENDPS/PD x,x,i'/v,v,mi 2 2 p015 p23 0.5 SSE4.1 BLENDVPS/PD x,m,xmm0 1 1 p015 1 1 SSE4.1 VBLENDVPS/PD v,v,m,v 2 2 2p015 p23 1 AVX VBLENDVPS/PD </td <td></td> <td></td> <td>1</td> <td></td> <td></td> <td>1</td> <td>1</td> <td></td>			1			1	1	
VPERMZF128 y,y,y,i 1 1 p5 3 1 AVX VPERMPS y,y,m,i 2 2 p5 p23 1 AVX VPERMPS y,y,y 1 1 p5 3 1 AVX2 VPERMPD y,y,i 1 1 p5 p23 1 AVX2 VPERMPD y,y,i 1 1 p5 p23 1 AVX2 VPERMPD y,m,i 2 2 p5 p23 1 AVX2 VPERMPD y,m,i 2 2 p5 p23 1 AVX2 VPERMPD y,m,i 2 2 p5 p23 1 AVX2 VPERMPD y,m,i 2 2 p015 p23 0.5 SSE4.1 BLENDPS/PD x,m,xmm0 1 1 p015 1 1 SSE4.1 VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX WBLENDVPS/PD v,v,m,v 3 3 <td></td> <td></td> <td>2</td> <td></td> <td></td> <td></td> <td>1</td> <td></td>			2				1	
VPERMPF128 y,y,m,i 2 2 p5 p23 1 AVX VPERMPS y,y,y 1 1 p5 3 1 AVX2 VPERMPB y,y,y 1 1 p5 3 1 AVX2 VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,m,i 2 2 p5 p23 1 AVX2 VPERMPD y,m,i 2 2 p5 p23 1 AVX2 BLENDPS/PD x,x,x/mm0 1 1 p015 1 0.33 SSE4.1 BLENDVPS/PD x,x,xmm0 1 1 p015 1 1 SSE4.1 BLENDVPS/PD x,x,xmm0 1 1 p015 2 1 AVX VBLENDVPS/PD x,x,x,mm0 1 1 p2 2 2 2015 AVX <td></td> <td></td> <td></td> <td></td> <td></td> <td>3</td> <td>1</td> <td></td>						3	1	
VPERMPS y,y,y 1 1 p5 3 1 AVX2 VPERMPS y,y,m 1 2 p5 p23 1 AVX2 VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,mi 2 2 p5 p23 1 AVX2 BLENDPS/PD x,x,i'/v,v,v,i 1 1 p015 1 0.33 SSE4.1 BLENDVPS/PD x,x,xmm0 1 1 p015 1 1 SSE4.1 BLENDVPS/PD x,x,xmm0 1 1 p015 p23 1 SSE4.1 BLENDVPS/PD x,v,v,w,v 2 2 2p015 p23 1 SSE4.1 BLENDVPS/PD x,v,v,m,w 3 3 2p015 p23 1 AVX VBLENDVPS/PD x,v,m,m 1 1 p5 1 1 SSE3								
VPERMPS y,y,m 1 2 p5 p23 1 AVX2 VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,m,i 2 2 p5 p23 1 AVX2 VPERMPD x,x,i / v,v,v,i 1 1 p015 1 0.33 SSE4.1 BLENDPS/PD x,m,i / v,v,m,i 2 2 p015 p23 0.5 SSE4.1 BLENDVPS/PD x,m,xmm0 1 1 p015 1 1 SSE4.1 BLENDVPS/PD x,m,xmm0 2 2 p015 p23 1 SSE4.1 VBLENDVPS/PD v,v,m,v 2 2 2p015 2 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,m,v 1 1 p23 3 0.5 SSE3						3	-	
VPERMPD y,y,i 1 1 p5 3 1 AVX2 VPERMPD y,m,i 2 2 p5 p23 1 AVX2 BLENDPS/PD x,x,i/v,v,v,i 1 1 p015 1 0.33 SSE4.1 BLENDVPS/PD x,m,mi/v,v,m,i 2 2 p015 p23 0.5 SSE4.1 BLENDVPS/PD x,m,xmm0 1 1 p015 1 1 SSE4.1 BLENDVPS/PD x,m,xmm0 2 2 p015 p23 1 SSE4.1 VBLENDVPS/PD v,v,v,v 2 2 2p015 2 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX			· ·				-	
VPERMPD y,m,i 2 2 p5 p23 1 AVX2 BLENDPS/PD x,x,i / v,v,v,i 1 1 p015 1 0.33 SSE4.1 BLENDVPS/PD x,m,i / v,v,m,i 2 2 p015 p23 0.5 SSE4.1 BLENDVPS/PD x,m,xmm0 1 1 p015 1 1 SSE4.1 BLENDVPS/PD x,m,xmm0 2 2 p015 p23 1 SSE4.1 VBLENDVPS/PD x,m,xmm0 2 2 p015 p23 1 AVX VBLENDVPS/PD v,v,v,v 2 2 2p015 2 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,m,m 1 1 p5 1 1 SSE3 MOVDDUP v,v,m 1 1 p23 3 0.5 AVX						2		
BLENDPS/PD x,x,i / v,v,v,i 1 1 p015 1 0.33 SSE4.1 BLENDPS/PD x,m,i / v,v,m,i 2 2 p015 p23 0.5 SSE4.1 BLENDVPS/PD x,x,xmm0 1 1 p015 p23 1 SSE4.1 VBLENDVPS/PD v,v,m,mm0 2 2 p015 p23 1 SSE4.1 VBLENDVPS/PD v,v,w,v 2 2 2p015 2 1 AVX WBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX WBLENDVPS/PD v,v,w,v 2 2 2p015 2 1 AVX WBLENDVPS/PD v,v,w,v 3 3 2p015 p23 1 AVX WBLENDVPS/PD v,v,v,v 2 2 2p015 2 1 AVX MOVDUP v,v,m 1 1 p23 3 0.5 SSE3 VBROADCASTSS y,m32 1 1 p23 3 0.5 <td></td> <td></td> <td></td> <td></td> <td>•</td> <td>3</td> <td> '</td> <td></td>					•	3	'	
BLENDPS/PD x,m,i / v,v,m,i 2 2 p015 p23 0.5 SSE4.1 BLENDVPS/PD x,x,xmm0 1 1 p015 1 1 SSE4.1 BLENDVPS/PD x,x,xmm0 1 1 p015 p23 1 SSE4.1 VBLENDVPS/PD v,v,v,v 2 2 2p015 p23 1 AVX VBLENDVPS/PD v,v,w,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,w,v 1 1 p5 1 AVX VBLENDVPS/PD v,v,w,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,w,v 1 1 p5 1 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX VBLENDVPS/PD v,v,m 1 1 p5 1 1 XX MOVDUP v,v 1 1 p23 3 0.5 AVX VBROAD		_				4	U 33	
BLENDVPS/PD x,x,xmm0 1 1 p015 1 1 SSE4.1 BLENDVPS/PD x,m,xmm0 2 2 p015 p23 1 SSE4.1 VBLENDVPS/PD v,v,v,v 2 2 2p015 2 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX MOVDDUP v,v 1 1 p5 1 1 SSE3 MOVDDUP v,m 1 1 p23 3 0.5 SSE3 VBROADCASTSS x,m32 1 1 p23 2 0.5 AVX VBROADCASTSS y,m32 1 1 p5 1 1 AVX2 VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p23 3 0.5 AVX VBROADCASTSD y,x 1 1 p5 3 1						I I		
BLENDVPS/PD x,m,xmm0 2 2 p015 p23 1 SSE4.1 VBLENDVPS/PD v,v,v,v 2 2 2p015 2 1 AVX VBLENDVPS/PD v,v,m,v 3 3 2p015 p23 1 AVX MOVDDUP v,v 1 1 p5 1 1 SSE3 VBROADCASTSS x,m32 1 1 p23 2 0.5 AVX VBROADCASTSS y,m32 1 1 p5 1 1 AVX2 VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p5 3 1 AVX2 VBROADCASTF128 y,m128 1 1 p5 1 1 <t< td=""><td></td><td></td><td></td><td></td><td></td><td>4</td><td></td><td></td></t<>						4		
VBLENDVPS/PD V,V,V,V 2 2 2p015 2 1 AVX VBLENDVPS/PD V,V,m,v 3 3 2p015 p23 1 AVX MOVDDUP V,v 1 1 p5 1 1 SSE3 MOVDDUP V,m 1 1 p23 3 0.5 SSE3 VBROADCASTSS X,m32 1 1 p23 2 0.5 AVX VBROADCASTSS Y,m32 1 1 p5 1 1 AVX2 VBROADCASTSS Y,x 1 1 p5 3 1 AVX2 VBROADCASTSD Y,m64 1 1 p5 3 1 AVX2 VBROADCASTSD Y,x 1 1 p5 3 1 AVX2 VBROADCASTSD Y,x 1 1 p5 3 1 AVX2 VBROADCASTSD Y,m128 1 1 p23 3 0.5						I I		
VBLENDVPS/PD V,V,m,V 3 3 2p015 p23 1 AVX MOVDDUP v,v 1 1 p5 1 1 SSE3 MOVDDUP v,m 1 1 p23 3 0.5 SSE3 VBROADCASTSS x,m32 1 1 p23 2 0.5 AVX VBROADCASTSS y,m32 1 1 p5 1 1 AVX2 VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTF128 y,m128 1 1 p5 3 1 AVX2 VBROADCASTF128 y,m128 1 1 p5 1 1 SSE3 MOVSH/LDUP v,m 1 1 p5 1 1			I			_	-	
MOVDDUP v,v 1 1 p5 1 1 SSE3 MOVDDUP v,m 1 1 p23 3 0.5 SSE3 VBROADCASTSS x,m32 1 1 p23 2 0.5 AVX VBROADCASTSS y,m32 1 1 p5 1 1 AVX2 VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m128 1 1 p23 3 0.5 AVX MOVSH/LDUP v,m 1 1 p5 1 1			I		•		-	
MOVDDUP v,m 1 1 p23 3 0.5 SSE3 VBROADCASTSS x,m32 1 1 p23 2 0.5 AVX VBROADCASTSS y,m32 1 1 p23 3 0.5 AVX VBROADCASTSS x,x 1 1 p5 1 1 AVX2 VBROADCASTSD y,m64 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p5 3 1 AVX2 VBROADCASTSD y,m128 1 1 p5 3 1 AVX2 VBROADCASTSD y,m128 1 1 p23 3 0.5 AVX VBROADCASTSD y,m128 1 1 p23 3 0.5 AVX VBROADCASTSD y,m128 1 1 p23 3 0.5 AVX MOVSH/LDUP v,m 1 1 p5 1							1	
VBROADCASTSS x,m32 1 1 p23 2 0.5 AVX VBROADCASTSS y,m32 1 1 p23 3 0.5 AVX VBROADCASTSS x,x 1 1 p5 1 1 AVX2 VBROADCASTSD y,m64 1 1 p23 3 0.5 AVX VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m128 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p5 3 1 AVX2 VBROADCASTF128 y,m128 1 1 p23 3 0.5 AVX MOVSH/LDUP v,m 1 1 p5 1			· •				1	
VBROADCASTSS y,m32 1 1 p23 3 0.5 AVX VBROADCASTSS x,x 1 1 p5 1 1 AVX2 VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX VBROADCASTSD y,x 1 1 p5 3 1 AVX MOVSH/LDUP y,m 1 1 p5 1 1			'					
VBROADCASTSS x,x 1 1 p5 1 1 AVX2 VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p23 3 0.5 AVX VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX MOVSH/LDUP y,m 1 1 p5 1 1 SSE3 UNPCKH/LPS/D x,m / v,v,w 1 1 p5 3 1			1	Ī .	•			
VBROADCASTSS y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,m64 1 1 p23 3 0.5 AVX VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTSD y,x 1 1 p23 3 0.5 AVX VBROADCASTSD y,x 1 1 p5 1 1 AVX MOVSH/LDUP y,m128 1 1 p5 1 1 SSE3 UNPCKH/LDS/D x,x,r/v,v,v 1 1 p5 1 1 SSE3 UNPCKH/LPS/D x,m/v,v,m 1 2 p5 p23 1		-	1		•			
VBROADCASTSD y,m64 1 1 p23 3 0.5 AVX VBROADCASTSD y,x 1 1 p5 3 1 AVX2 VBROADCASTF128 y,m128 1 1 p23 3 0.5 AVX MOVSH/LDUP v,v 1 1 p5 1 1 SSE3 MOVSH/LDUP v,m 1 1 p23 3 0.5 SSE3 UNPCKH/LPS/D v,m 1 1 p5 1 1 SSE3 UNPCKH/LPS/D x,x / v,v,v 1 2 p5 p23 1 SSE3 EXTRACTPS r32,x,i 2 2 p0 p5 3 1 SSE4.1 EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 </td <td></td> <td></td> <td>1</td> <td></td> <td></td> <td>-</td> <td></td> <td></td>			1			-		
VBROADCASTSD VBROADCASTF128 y,x 1 1 p5 3 1 AVX2 VBROADCASTF128 y,m128 1 1 p23 3 0.5 AVX MOVSH/LDUP v,v 1 1 p5 1 1 SSE3 MOVSH/LDUP v,m 1 1 p23 3 0.5 SSE3 UNPCKH/LPS/D x,x / v,v,v 1 1 p5 1 1 SSE3 UNPCKH/LPS/D x,m / v,v,m 1 2 p5 p23 1 SSE3 EXTRACTPS r32,x,i 2 2 p0 p5 3 1 SSE4.1 VEXTRACTF128 m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX VEXTRACTF128 x,x,i 1 1 p5 1 1 AVX INSERTPS x,x,i 1 1 p5 </td <td></td> <td>-</td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td>		-	1					
VBROADCASTF128 y,m128 1 1 p23 3 0.5 AVX MOVSH/LDUP v,v 1 1 p5 1 1 SSE3 MOVSH/LDUP v,m 1 1 p23 3 0.5 SSE3 UNPCKH/LPS/D x,x / v,v,v 1 1 p5 1 1 SSE3 UNPCKH/LPS/D x,m / v,v,m 1 2 p5 p23 1 SSE3 EXTRACTPS r32,x,i 2 2 p0 p5 3 1 SSE4.1 EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5		y,m64	1	1	•		0.5	
MOVSH/LDUP v,v 1 1 p5 1 1 SSE3 MOVSH/LDUP v,m 1 1 p23 3 0.5 SSE3 UNPCKH/LPS/D x,x / v,v,v 1 1 p5 1 1 SSE3 UNPCKH/LPS/D x,m / v,v,m 1 2 p5 p23 1 SSE3 EXTRACTPS r32,x,i 2 2 p0 p5 3 1 SSE4.1 EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1	VBROADCASTSD	_	1	1	· ·			
MOVSH/LDUP v,m 1 1 p23 3 0.5 SSE3 UNPCKH/LPS/D x,x / v,v,v 1 1 p5 1 1 SSE3 UNPCKH/LPS/D x,m / v,v,m 1 2 p5 p23 1 SSE3 EXTRACTPS r32,x,i 2 2 p0 p5 3 1 SSE4.1 EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1		y,m128	1	1	•	3	0.5	
UNPCKH/LPS/D x,x / v,v,v 1 1 p5 1 1 SSE3 UNPCKH/LPS/D x,m / v,v,m 1 2 p5 p23 1 SSE3 EXTRACTPS r32,x,i 2 2 p0 p5 3 1 SSE4.1 EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1	MOVSH/LDUP	V,V	1	1	p5	1	1	SSE3
UNPCKH/LPS/D x,m / v,v,m 1 2 p5 p23 1 SSE3 EXTRACTPS r32,x,i 2 2 p0 p5 3 1 SSE4.1 EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1	MOVSH/LDUP	v,m	1	1	p23	3	0.5	SSE3
EXTRACTPS r32,x,i 2 2 p0 p5 3 1 SSE4.1 EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1	UNPCKH/LPS/D	x,x / v,v,v	1	1	p5	1	1	SSE3
EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1	UNPCKH/LPS/D	x,m / v,v,m	1		p5 p23		1	SSE3
EXTRACTPS m32,x,i 2 3 p4 p5 p23 4-5 1 SSE4.1 VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1	EXTRACTPS	r32,x,i	2	2		3	1	SSE4.1
VEXTRACTF128 x,y,i 1 1 p5 3 1 AVX VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1	EXTRACTPS		2	3		4-5	1	SSE4.1
VEXTRACTF128 m128,y,i 2 2 p23 p4 5 1 AVX INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1	VEXTRACTF128		1	1			1	AVX
INSERTPS x,x,i 1 1 p5 1 1 SSE4.1 INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1			2	2	· ·	5	1	
INSERTPS x,m32,i 2 2 p23 p5 3-4 1 SSE4.1		_					1	
			2		· ·	3-4	1	
	VINSERTF128	y,y,x,i			p5		1	AVX

VINOEDTE400	400 :			04500	1 45	0.5	l AV.04 l
VINSERTF128	y,y,m128,i	2	2	p015 p23	4-5	0.5	AVX
VMASKMOVPS/D	v,v,m	2	2	p015 p23	3	0.5	AVX
VMASKMOVPS/D	m128,x,x	4	4	p0 p4 p23	13	1	AVX
VMASKMOVPS/D	m256,y,y	4	4	p0 p4 p23	14	1	AVX
VGATHERDPS	x,[r+s*x],x	4	4	p0 p1 p23 p5	12	4	AVX2
VGATHERDPS	y,[r+s*y],y	4	4	p0 p1 p23 p5	13	5	AVX2
VGATHERQPS	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
VGATHERQPS	x,[r+s*y],x	4	4	p0 p1 p23 p5		4	AVX2
VGATHERDPD	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
VGATHERDPD	y,[r+s*x],y	4	4	p0 p1 p23 p5		4	AVX2
VGATHERQPD	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
VGATHERQPD	y,[r+s*y],y	4	4	p0 p1 p23 p5		4	AVX2
Conversion							
CVTPD2PS	X,X	2	2	p01 p5	5	1	
CVTPD2PS	x,m128	2	3	p01 p5 p23		1	
VCVTPD2PS	x,y	2	2	p01 p5	7	1	AVX
VCVTPD2PS	x,m256	2	3	p01 p5 p23		1	AVX
CVTSD2SS	X,X	2	2	p01 p5	5	1	
CVTSD2SS	x,m64	2	3	p01 p5 p23		1	
CVTPS2PD	X,X	2	2	p01 p5	5	1	
CVTPS2PD	x,m64	1	2	p01 p23		0.5	
VCVTPS2PD	y,x	2	2	p01 p20	7	1	AVX
VCVTPS2PD	y,m128	1	2	p01 p23	,	0.5	AVX
CVTSS2SD	x,x	2	2	p01 p20	5	2	/\\
CVTSS2SD	x,m32	1	2	p01 p5 p23		2	
CVTDQ2PS	X,11132 X,X		1	p01 p3 p23	4	0.5	
CVTDQ2PS	x,m128		2	p01 p23	4	0.5	
VCVTDQ2PS			1	p01 p23	4	0.5	AVX
VCVTDQ2F3	y,y v m256		2	p01 p23	4	0.5	AVX
	y,m256	-	1		4		AVA
CVT(T) PS2DQ	X,X	1		p01	4	0.5	
CVT(T) PS2DQ	x,m128	1	2	p01 p23		0.5	A) /)/
VCVT(T) PS2DQ	у,у	1	1	p01	4	0.5	AVX
VCVT(T) PS2DQ	y,m256	1	2	p01 p23	_	0.5	AVX
CVTDQ2PD	X,X	2	2	p01 p5	5	1	
CVTDQ2PD	x,m64	1	2	p01 p23	_	0.5	1100
VCVTDQ2PD	у,х	2	2	p01 p5	7	1	AVX
VCVTDQ2PD	y,m128	1	2	p01 p23	_	0.5	AVX
CVT(T)PD2DQ	X,X	2	2	p01 p5	5	1	
CVT(T)PD2DQ	x,m128	3	3	p01 p23 p5	_	1	
VCVT(T)PD2DQ	x,y	2	2	p01 p5	7	1	AVX
VCVT(T)PD2DQ	x,m256	2	3	p01 p23 p5		1	AVX
CVTPI2PS	x,mm	2	2	p01 p1	6	2	
CVTPI2PS	x,m64	1	2	p01 p23		2	
CVT(T)PS2PI	mm,x	2	2	p0 p5	7	1	
CVT(T)PS2PI	mm,m128	2	2	p0 p23		1	
CVTPI2PD	x,mm	2	2	p01 p5	5	1	
CVTPI2PD	x,m64	1	2	p01 p23		0.5	
CVT(T) PD2PI	mm,x	2	2	p01 p5	5	1	
CVT(T) PD2PI	mm,m128	2	3	p01 p23 p5		1	
CVTSI2SS	x,r32	2	2	p01 p5	6	2	
CVTSI2SS	x,r64	3	3	p01 2p5	7	2	
CVTSI2SS	x,m32	1	2	p01 p23		2	

CVT(T)SS2SI	r32,x	2	2	2p01	6	1	
CVT(T)SS2SI	r64,x	3	3	2p01 p5	7	1	
CVT(T)SS2SI	r32,m32	3	3	2p01 p23		1	
CVTSI2SD	x,r32/64	2	2	p01 p5	6	2	
CVTSI2SD	x,m32	1	2	p01 p23		2	
CVT(T)SD2SI	r32/64,x	2	2	p0 p1	6	1	
CVT(T)SD2SI	r32,m64	3	3	2p01 p23	,	1	
VCVTPS2PH	x,v,i	2	2	p01 p5	5-7	1	F16C
VCVTPS2PH	m,v,i	3	3	p01 p4 p23	0 /	1	F16C
VCVTPH2PS	V,X	2	2	p01 p4 p20	5-7	1	F16C
VCVTPH2PS	v,x v,m	1	2	p01 p23	J-1	0.5	F16C
VCVIFIIZES	V,111	ļ !		p01 p23		0.5	1 100
Arithmetic							
ADDSS/D PS/D							
SUBSS/D PS/D	x,x / v,v,v	1	1	p01	4	0.5	
ADDSS/D PS/D	^,^ / V,V,V	ļ !	'	poi	7	0.5	
SUBSS/D PS/D	x,m / v,v,m	1	2	p01 p23		0.5	
ADDSUBPS/D	x,x / v,v,v	1	1	p01 p23	4	0.5	SSE3
ADDSUBPS/D	x,m / v,v,m	1	2	p01 p23	7	0.5	SSE3
HADDPS/D	X,111 / V,V,111	ı		p01 p23		0.5	3323
HSUBPS/D	x,x / v,v,v	3	3	p01 2p5	6	2	SSE3
HADDPS/D	X,X / V,V,V	٥	٥	pu i zpo	O	2	SSES
HSUBPS/D	x,m / v,v,m	4	4	p1 2p5 p23		2	SSE3
MULSS/D PS/D	x,x / v,v,v	1	1	p1 2p3 p23	4	0.5	3323
		1	2	•	4		
MULSS/D PS/D	x,m / v,v,m			p01 p23	44	0.5	
DIVSS	X,X	1	1	p0	11	3	
DIVPS	X,X	1	1	p0	11	3	
DIVSS DIVPS	x,m	1	2	p0 p23	40.44	3	
DIVSD	X,X	1	1	p0	13-14	4	
DIVPD	X,X	1	1	p0	13-14	4	
DIVSD DIVPD	x,m	1	2	p0 p23		4	
VDIVPS	y,y,y	1	1	p0	11	5	AVX
VDIVPS	y,y,m256	1	2	p0 p23		5	AVX
VDIVPD	y,y,y	1	1	p0	13-14	8	AVX
VDIVPD	y,y,m256	1	2	p0 p23		8	AVX
RCPSS/PS	V,V	1	1	p0	4	1	
RCPSS/PS	v,m	1	2	p0 p23		1	
CMPccSS/D							
CMPccPS/D	x,x / v,v,v	1	1	p01	4	0.5	
CMPccSS/D							
CMPccPS/D	x,m / v,v,m	2	2	p01 p23		0.5	
(U)COMISS/D	X,X	1	1	p0		1	
(U)COMISS/D	x,m32/64	2	2	p0 p23		1	
MAXSS/D PS/D							
MINSS/D PS/D	x,x / v,v,v	1	1	p01	4	0.5	
MAXSS/D PS/D	_						
MINSS/D PS/D	x,m / v,v,m	1	2	p01 p23		0.5	
ROUNDSS/D PS/D	v,v,i	2	2	2p01	8	1	SSE4.1
ROUNDSS/D PS/D	v,m,i	3	3	2p01 p23		1	SSE4.1
DPPS	x,x,i / v,v,v,i	4	4	3p01 p5	13	1.5	SSE4.1
DPPS	x,m,i / v,v,m,i	6	6	3p01 p23 p5 p6		4	SSE4.1
DPPD	x,x,i	3	3	2p01 p5	9	1	SSE4.1
DPPD	x,m128,i	4	4	2p01 p23 p5		1	SSE4.1

VFMADD							
(all FMA instr.)	V,V,V	1	1	p01	4	0.5	FMA
VFMADD				-			
(all FMA instr.)	v,v,m	1	2	p01 p23		0.5	FMA
Math							
SQRTSS/PS	X,X	1	1	p0	12	3	
SQRTSS/PS	x,m128	1	2	p0 p23		3	
VSQRTPS	y,y	1	1	p0	12	6	AVX
VSQRTPS	y,m256	4	4	p0 p23		6	AVX
SQRTSD	x,x	1	1	p0	15-16	4-6	
SQRTPD	X,X X,X	1	1	p0	15-16	4-6	
SQRTSD/PD	x,m128	1	2	p0 p23	10 10	4-6	
VSQRTPD	y,y	1	1	p0	15-16	9-12	AVX
VSQRTPD	y,m256	4	4	p0 p23	10 10	9-12	AVX
RSQRTSS/PS	V,V	1	1	p0	4	1	
RSQRTSS/PS	v,m	1	2	p0 p23	•	1	
	,,,,,	-	_	F - F - 5			
Logic							
AND/ANDN/OR/							
XORPS/PD	x,x / v,v,v	1	1	p015	1	0.33	
AND/ANDN/OR/		_		04500		0.5	
XORPS/PD	x,m / v,v,m	1	2	p015 p23		0.5	
Other							
VZEROUPPER		4	4			1	AVX
VZEROALL		25	25	p0 p1 p5 p6		12	AVX, 32 bit
VZEROALL		34	34	p0 p1 p5 p6		12	AVX, 64 bit
LDMXCSR	m32	4	4	p0 p5 p6 p23	5	3	
STMXCSR	m32	3	4	p0 p4 p6 p237	4	1	
FXSAVE	m4096	106	120		60	60	32 bit mode
FXSAVE	m4096	114	136		65	65	64 bit mode
FXRSTOR	m4096	105	105		76	76	32 bit mode
FXRSTOR	m4096	121	121		77	77	64 bit mode
XSAVE		247	272		107	107	32 bit mode
XSAVE		271	304		107	107	64 bit mode
XRSTOR		190	190		79	79	
XSAVEOPT	m	214	221		84	84	

Intel Pentium 4

List of instruction timings and uop breakdown

This list is measured for a Pentium 4, model 2. Timings for model 3 may be more like the values for P4E, listed on the next sheet

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB,

JNE, etc.

i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit Operands:

> mmx register, xmm = 128 bit xmm register, sr = segment register, m = any memory operand including indirect operands, m64 means 64-bit memory op-

erand, etc.

Number of µops issued from instruction decoder and stored in trace cache. μops:

Microcode: Number of additional uops issued from microcode ROM.

This is the delay that the instruction generates in a dependency chain if the Latency:

next dependent instruction starts in the same execution unit. The numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latency of moves to and from memory cannot be measured accurately because of the problem with memory intermediates explained

above under "How the values were measured".

Additional latency: This number is added to the latency if the next dependent instruction is in a

different execution unit. There is no additional latency between ALU0 and

ALU1.

This is also called issue latency. This value indicates the number of clock cy-Reciprocal throughput:

cles from the execution of an instruction begins to a subsequent independent

instruction can begin to execute in the same execution subunit. A value of

0.25 indicates 4 instructions per clock cycle in one thread.

The port through which each upp goes to an execution unit. Two independent Port:

uops can start to execute simultaneously only if they are going through differ-

Execution unit: Use this information to determine additional latency. When an instruction with

more than one µop uses more than one execution unit, only the first and the

last execution unit is listed.

Throughput measures apply only to instructions executing in the same sub-Execution subunit:

unit.

Instruction set Indicates the compatibility of an instruction with other 80x86 family micropro-

cessors. The instruction can execute on microprocessors that support the in-

struction set indicated.

Integer instructions

Instruction	Operands	sdorl	Micr	Latency	Addi	Reci put	Port	Exec	Subunit	Instr	Notes
		0,	Microcode	ncy	Additional latency	Reciprocal through- put		Execution unit	ınit	Instruction set	Š
					laten	thro		unit		set	
					cy	ugh-					
Move instructions											
MOV	r,r	1	0	0.5	0.5-1		0/1	alu0/1		86	С
MOV	r,i	1	0	0.5	0.5-1	0.25	0/1	alu0/1		86	
MOV	r32,m	1	0	2	0	1	2	load		86	
MOV	r8/16,m	2	0	3	0	1	2	load		86	
MOV	m,r	1	0	1		2	0	store		86	b, c
MOV	m,i	3	0			2	0,3	store		86	
MOV	r,sr	4	2			6				86	
MOV	sr,r/m	4	4	12	0	14				86	a, q
MOVNTI	m,r32	2	0			≈33				sse2	
MOVZX	r,r	1	0	0.5	0.5-1	0.25	0/1	alu0/1		386	С
MOVZX	r,m	1	0	2	0	1	2	load		386	
MOVSX	r,r	1	0	0.5	0.5-1	0.5	0	alu0		386	С
MOVSX	r,m	2	0	3	0.5-1	1	2,0			386	
CMOVcc	r,r/m	3	0	6	0	3				ppro	a, e
XCHG	r,r	3	0	1.5	0.5-1	1	0/1	alu0/1		86	
XCHG	r,m	4	8	>100						86	
XLAT		4	0	3						86	
PUSH	r	2	0	1		2				86	
PUSH	i	2	0	1		2				186	
PUSH	m	3	0			2				86	
PUSH	sr	4	4			7				86	
PUSHF(D)		4	4			10				86	
PUSHA(D)		4	10			19				186	
POP	r	2	0	1	0	1				86	
POP	m	4	8			14				86	
POP	sr	4	5			13				86	
POPF(D)		4	8			52				86	
POPA(D)		4	16			14				186	
LEA	r,[r+r/i]	1	0	0.5	0.5-1	0.25	0/1	alu0/1		86	
LEA	r,[r+r+i]	2	0	1	0.5-1		0/1	alu0/1		86	
LEA	r,[r*i]	3	0	4	0.5-1	1	1	int,alu		386	
LEA	r,[r+r*i]	2	0	4	0.5-1		1	int,alu		386	
LEA	r,[r+r*i+i]	3	0	4	0.5-1	1	1	int,alu		386	
LAHF	.,[]	1	0	4	0	4	1	int		86	
SAHF		1	0	0.5	0.5-1	0.5	0/1	alu0/1		86	d
SALC		3	0	5	0	1	1	int		86	
LDS, LES,	r,m	4	7			15	-			86	
BSWAP	r	3	0	7	0	2		int,alu		486	
IN, OUT	r,r/i	8	64			- >100	0	,	86		
PREFETCHNTA	m	4	2			6	- 			sse	
PREFETCHT0/1/2	m	4	2			6				sse	
SFENCE		4	2			40				sse	
LFENCE		4	2			38				sse2	
MFENCE		4	2			100				sse2	
IVII LINOL	1	"	_	1		100			1	3362	

				1							
Arithmetic instructions											
ADD, SUB	r,r	1	0	0.5	0.5-1	0.25	0/1	alu0/1		86	С
ADD, SUB	r,m	2	0	1	0.5-1	1				86	С
ADD, SUB	m,r	3	0	≥ 8		≥ 4				86	С
ADC, SBB	r,r	4	4	6	0	6	1	int,alu		86	
ADC, SBB	r,i	3	0	6	0	6	1	int,alu		86	
ADC, SBB	r,m	4	6	8	0	8	1	int,alu		86	
ADC, SBB	m,r	4	7	≥ 9		8	•	,		86	
CMP	r,r	1	0	0.5	0.5-1	_	0/1	alu0/1		86	С
CMP	r,m	2	0	1	0.5-1	1	0, .	G.1.G.07 .		86	С
INC, DEC	r	2	0	0.5	0.5-1		0/1	alu0/1		86	_
INC, DEC	m .	4	0	4		≥ 4	0, .	G.1.G.07 .		86	
NEG	r	1	0	0.5	0.5-1	0.5	0	alu0		86	
NEG	m .	3	0	0.0	0.0 .	≥ 3		aido		86	
AAA, AAS		4	27	90		- 0				86	
DAA, DAS		4	57	100						86	
AAD		4	10	22			1	int	fpmul	86	
AAM		4	22	56			1	int	fpdiv	86	
MUL, IMUL	r8/32	4	6	16	0	8	1	int	fpmul	86	
MUL, IMUL	r16	4	7	17	0	8	1	int	fpmul	86	
MUL, IMUL	m8/32	4	7-8	16	0	8	1	int	fpmul	86	
MUL, IMUL	m16	4	10	16	0	8	1	int	fpmul	86	
IMUL	r32,r	4	0	14	0	4.5	1	int	fpmul	386	
IMUL	r32,(r),i	4	0	14	0	4.5	1	int	fpmul	386	
IMUL	r16,r	4	5	16	0	9	1	int	fpmul	386	
IMUL	r16,r,i	4	5	15	0	8	1	int	fpmul	186	
IMUL	r16,m16	4	7	15	0	10	1	int	fpmul	386	
IMUL	r32,m32	4	0	14	0	8	1	int	fpmul	386	
IMUL	r,m,i	4	7	14	0	10	1	int	fpmul	186	
DIV	r8/m8	4	20	61	0	24	1	int	fpdiv	86	а
DIV	r16/m16	4	18	53	0	23	1	int	fpdiv	86	а
DIV	r32/m32	4	21	50	0	23	1	int	fpdiv	386	
IDIV	r8/m8	4	24	61	0	24	1	int	fpdiv	86	а
IDIV	r16/m16	4	22	53	0	23	1	int	fpdiv	86	а
IDIV	r32/m32	4	20	50	0	23	1	int	fpdiv	386	а
CBW		2	0	1	0.5-1		0	alu0	•	86	
CWD, CDQ		2	0	1	0.5-1		0/1	alu0/1		86	
CWDE		1	0	0.5	0.5-1		0	alu0		386	
Logic instructions											
AND, OR, XOR	r,r	1	0	0.5	0.5-1	0.5	0	alu0		86	С
AND, OR, XOR	r,m	2	0	≥ 1	0.5-1	≥ 1				86	С
AND, OR, XOR	m,r	3	0	≥ 8		≥ 4				86	С
TEST	r,r	1	0	0.5	0.5-1	0.5	0	alu0		86	С
TEST	r,m	2	0	≥ 1	0.5-1	≥ 1				86	С
NOT	r	1	0	0.5	0.5-1	0.5	0	alu0		86	
NOT	m	4	0			≥ 4				86	
SHL, SHR, SAR	r,i	1	0	4	1	1	1	int	mmxsh	186	
SHL, SHR, SAR	r,CL	2	0	6	0	1	1	int	mmxsh	86	d
ROL, ROR	r,i	1	0	4	1	1	1	int	mmxsh	186	d
ROL, ROR	r,CL	2	0	6	0	1	1	int	mmxsh	86	d
RCL, RCR	r,1	1	0	4	1	1	1	int	mmxsh	86	d

RCL, RCR	r,i	4	15	16	0	15	1	int	mmxsh	186	d
RCL, RCR	r,CL	4	15	16	0	14	1	int	mmxsh	86	d
SHL,SHR,SAR,ROL,											
ROR	m,i/CL	4	7-8	10	0	10	1	int	mmxsh	86	d
RCL, RCR	m,1	4	7	10	0	10	1	int	mmxsh	86	d
RCL, RCR	m,i/CL	4	18	18-28	}	14	1	int	mmxsh	86	d
SHLD, SHRD	r,r,i/CL	4	14	14	0	14	1	int	mmxsh	386	
SHLD, SHRD	m,r,i/CL	4	18	14	0	14	1	int	mmxsh	386	
BT	r,i	3	0	4	0	2	1	int	mmxsh	386	d
BT	r,r	2	0	4	0	1	1	int	mmxsh	386	d
ВТ	m,i	4	0	4	0	2	1	int	mmxsh	386	d
ВТ	m,r	4	12	12	0	12	1	int	mmxsh	386	d
BTR, BTS, BTC	r,i	3	0	6	0	2	1	int	mmxsh	386	
BTR, BTS, BTC	r,r	2	0	6	0	4	1	int	mmxsh	386	
BTR, BTS, BTC	m,i	4	7	18	0	8	1	int	mmxsh	386	
BTR, BTS, BTC	m,r	4	15	14	0	14	1	int	mmxsh	386	
BSF, BSR	r,r	2	0	4	0	2	1	int	mmxsh	386	
BSF, BSR	r,m	3	0	4	0	3	1	int	mmxsh	386	
SETcc	r	3	0	5	0	1	1	int		386	
SETcc	m	4	0	5	0	3	1	int		386	
CLC, STC		3	0	10	0	2				86	d
CMC		3	0	10	0	2				86	
CLD		4	7	52	0	52				86	
STD		4	5	48	0	48				86	
CLI		4	5	35		35				86	
STI		4	12	43		43				86	
Control transfer instruct	tions										
Control transfer instruct	tions short/near	1	0	0	0	1	0	alu0	branch	86	
		1 4	0 28	0 118	0	1 118	0	alu0	branch	86 86	
JMP	short/near	_	-	-	0		-	alu0 alu0	branch branch		
JMP JMP	short/near far	4	28	118	0	118	0			86	
JMP JMP	short/near far r	4 3	28 0	118 4	0	118 4	0	alu0	branch	86 86	
JMP JMP JMP JMP	short/near far r m(near)	4 3 3	28 0 0	118 4 4	0	118 4 4	0 0	alu0	branch	86 86 86	
JMP JMP JMP JMP JCC	short/near far r m(near) m(far)	4 3 3 4	28 0 0 31	118 4 4 11	0	118 4 4 11	0 0 0 0	alu0 alu0	branch branch	86 86 86	
JMP JMP JMP JMP JMP	short/near far r m(near) m(far) short/near	4 3 3 4 1	28 0 0 31 0	118 4 4 11 0	0	118 4 4 11 2-4	0 0 0 0	alu0 alu0 alu0	branch branch	86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ	short/near far r m(near) m(far) short/near short	4 3 3 4 1 4	28 0 0 31 0 4	118 4 4 11 0	0	118 4 4 11 2-4 2-4	0 0 0 0 0 0	alu0 alu0 alu0 alu0	branch branch branch branch	86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP	short/near far r m(near) m(far) short/near short short	4 3 3 4 1 4 4	28 0 0 31 0 4 4	118 4 4 11 0 0	0	118 4 4 11 2-4 2-4 2-4	0 0 0 0 0 0	alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch	86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL	short/near far r m(near) m(far) short/near short short near	4 3 3 4 1 4 4 3	28 0 0 31 0 4 4	118 4 4 11 0 0	0	118 4 4 11 2-4 2-4 2-4	0 0 0 0 0 0	alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch	86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL	short/near far r m(near) m(far) short/near short short near far	4 3 3 4 1 4 4 4 3 4	28 0 0 31 0 4 4 0 34	118 4 4 11 0 0 0	0	118 4 4 11 2-4 2-4 2-4	0 0 0 0 0 0 0 0 0	alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch	86 86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL	short/near far r m(near) m(far) short/near short short near far r	4 3 3 4 1 4 4 3 4	28 0 0 31 0 4 4 0 34 4	118 4 4 11 0 0 0 2	0	118 4 4 11 2-4 2-4 2-4	0 0 0 0 0 0 0 0 0 0	alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch	86 86 86 86 86 86 86	
JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL	short/near far r m(near) m(far) short/near short short near far r m(near)	4 3 3 4 1 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4	118 4 4 11 0 0 0 2	0	118 4 4 11 2-4 2-4 2-4		alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch	86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL CALL	short/near far r m(near) m(far) short/near short short near far r m(near)	4 3 3 4 1 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 4 38	118 4 4 11 0 0 0 2 8 9	0	118 4 4 11 2-4 2-4 2-4		alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL CALL CALL CAL	short/near far r m(near) m(far) short/near short short near far r m(near) m(far)	4 3 3 4 1 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0	118 4 4 11 0 0 0 2 8 9	0	118 4 4 11 2-4 2-4 2-4		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL CALL CALL RETN RETN	short/near far r m(near) m(far) short/near short short near far r m(near) m(far)	4 3 3 4 1 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0	118 4 4 11 0 0 0 2 8 9	0	118 4 4 11 2-4 2-4 2-4		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL CALL RETN RETN RETF	short/near far r m(near) m(far) short/near short near far r m(near) m(far)	4 3 3 4 1 4 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0 0 33	118 4 4 11 0 0 0 2 8 9	0	118 4 4 11 2-4 2-4 2-4		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL CALL CALL RETN RETN RETF RETF	short/near far r m(near) m(far) short/near short near far r m(near) m(far)	4 3 3 4 1 4 4 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0 0 33 33	118 4 4 11 0 0 0 2 8 9 2 2 11 11	0	118 4 4 11 2-4 2-4 2-4		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL RETN RETN RETF RETF IRET	short/near far r m(near) m(far) short/near short near far r m(near) m(far) i	4 3 3 4 1 4 4 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0 0 33 33 48	118 4 4 11 0 0 0 2 8 9 2 2 11 11 24 26	0	118 4 4 11 2-4 2-4 2-4 2		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL RETN RETN RETF RETF IRET ENTER	short/near far r m(near) m(far) short/near short short near far r m(near) m(far) i	4 3 3 4 1 4 4 4 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0 0 33 33 48 12	118 4 4 11 0 0 0 2 8 9 2 2 11 11 24 26	0	118 4 4 11 2-4 2-4 2 2		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL RETN RETN RETF RETF IRET ENTER ENTER	short/near far r m(near) m(far) short/near short short near far r m(near) m(far) i	4 3 3 4 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0 0 33 33 48 12 45+2	118 4 4 11 0 0 0 2 8 9 2 2 11 11 24 26 4n	0	118 4 4 11 2-4 2-4 2 2 2 128+		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86 86 86 86 86	
JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL RETN RETN RETF RETF RETF IRET ENTER ENTER LEAVE	short/near far r m(near) m(far) short/near short near far r m(near) m(far) i i i,0 i,n	4 3 3 4 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0 0 33 33 48 12 45+2 0	118 4 4 11 0 0 0 2 8 9 2 2 11 11 24 26 4n 3	0	118 4 4 11 2-4 2-4 2-4 2 2 128+ 3		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86 86 86 86 86 8	
JMP JMP JMP JMP JMP JCC J(E)CXZ LOOP CALL CALL CALL CALL CALL RETN RETN RETF RETF IRET ENTER ENTER LEAVE BOUND	short/near far r m(near) m(far) short/near short near far r m(near) m(far) i i i,0 i,n	4 3 3 4 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	28 0 0 31 0 4 4 0 34 4 4 38 0 0 33 33 48 12 45+2 0 14	118 4 4 11 0 0 0 2 8 9 2 2 11 11 24 26 4n 3 14	0	118 4 4 11 2-4 2-4 2-4 2 2 128+ 3 14		alu0 alu0 alu0 alu0 alu0 alu0 alu0 alu0	branch branch branch branch branch branch branch branch	86 86 86 86 86 86 86 86 86 86 86 86 86 8	

String instructions										
LODS	4	3	6		6				86	
REP LODS	4	5n	≈ 4n+	-36			86			
STOS	4	2	6		6				86	
REP STOS	4	2n+3	≈ 3n+	-10			86			
MOVS	4	4	6		4				86	
REP MOVS	4	≈163·	+1.1n				86			
SCAS	4	3			6				86	
REP SCAS	4	≈ 40+	-6n	≈4n				86		
CMPS	4	5			8				86	
REP CMPS	4	≈ 50+	-8n	≈4n				86		
Other										
NOP (90)	1	0	0		0.25	0/1	alu0/1		86	
Long NOP (0F 1F)	1	0	0		0.25	0/1	alu0/1		ppro	
PAUSE	4	2							sse2	
CPUID	4	39-81	İ	200-5	00		p5			
RDTSC	4	7			80				p5	

Notes:

a) Add 1 µop if source is a memory operand.

b) Uses an extra μop (port 3) if SIB byte used. A SIB byte is needed if the mem-

ory operand has more than one pointer register, or a scaled index, or ESP is

used as base pointer.

c) Add 1 µop if source or destination, but not both, is a high 8-bit register (AH,

BH, CH, DH).

d) Has (false) dependence on the flags in most cases.

e) Not available on PMMX

q) Latency is 12 in 16-bit real or virtual mode, 24 in 32-bit protected mode.

Floating point x87 instructions

Instruction	Operands	hobs	Microcode	Latency	Additional latency	Reciprocal through put	Port	Execution unit	Subunit	Instruction set	Notes
Move instructions						7					
FLD	- r	1	0	6	0	1	0	mov		87	
FLD	m32/64	1	0	≈ 7	0	1	2	load		87	
FLD	m80	3	4			6	2	load		87	
FBLD	m80	3	75			90	2	load		87	
FST(P)	r	1	0	6	0	1	0	mov		87	
FST(P)	m32/64	2	0	≈ 7		2-3	0	store		87	
FSTP	m80	3	8			8	0	store		87	
FBSTP	m80	3	311			400	0	store		87	
FXCH	r	1	0	0	0	1	0	mov		87	
FILD	m16	3	3	≈ 10		6	2	load		87	
FILD	m32/64	2	0	≈ 10		1	2	load		87	
FIST	m16	3	0	≈ 10		2-4	0	store		87	
FIST	m32/64	2	0	≈ 10		2-3	0	store		87	

									1		
FISTP	m	3	0	≈ 10		2-4	0	store		87	
FLDZ		1	0			2	0	mov		87	
FLD1		2	0			2	0	mov		87	
FCMOVcc	st0,r	4	0	2-4	1	4	1	fp		PPro	е
FFREE	r	3	0			4	0	mov		87	
FINCSTP, FDECSTP	·	1	0	0	0	1	0	mov		87	
FNSTSW	AX	4	0	11	0	3	1	11101		287	
			_		-		-				
FSTSW	AX	6	0	11	0	3	1			287	
FNSTSW	m16	4	4			6	0			87	
FNSTCW	m16	4	4			6	0			87	
FLDCW	m16	4	7	(3)		(8)	0,2			87	f
Arithmetic instructions											
FADD(P),FSUB(R)(P)	r	1	0	5	1	1	1	fp	add	87	
FADD,FSUB(R)	m	2	0	5	1	1	1	fp	add	87	
FIADD, FISUB(R)	m16	3	4	6	0	6	1	fp	add	87	
FIADD,FISUB(R)	m32	3	0	5	1	2	1	fp	add	87	
FMUL(P)	r	1	0	7	1	2	1	fp	mul	87	
` '		2	_	7		2		-			
FMUL	m		0		1		1	fp	mul	87	
FIMUL	m16	3	4	7	1	6	1	fp	mul	87	
FIMUL	m32	3	0	7	1	2	1	fp	mul	87	
FDIV(R)(P)	r	1	0	43	0	43	1	fp	div	87	g, h
FDIV(R)	m	2	0	43	0	43	1	fp	div	87	g, h
FIDIV(R)	m16	3	4	43	0	43	1	fp	div	87	g, h
FIDIV(R)	m32	3	0	43	0	43	1	fp	div	87	g, h
FABS	_	1	0	2	1	1	1	fp	misc	87	J,
FCHS		1	0	2	1	1	1	fp	misc	87	
FCOM(P), FUCOM(P)	r	1	0	2	0	1	1		misc	87	
			_		-	_		fp			
FCOM(P)	m	2	0	2	0	1	1	fp	misc	87	
FCOMPP, FUCOMPP		2	0	2	0	1	1	fp	misc	87	
FCOMI(P)	r	3	0	10	0	3	0,1	fp	misc	PPro	
FICOM(P)	m16	4	4			6	1	fp	misc	87	
FICOM(P)	m32	3	0	2	0	2	1,2	fp	misc	87	
FTST		1	0	2	0	1	1	fp	misc	87	
FXAM		1	0	2	0	1	1	fp	misc	87	
FRNDINT		3	15	23	0	15	0,1	•		87	
FPREM		6	84	212			1	fp		87	
FPREM1		6	84	212			1	fp		387	
							•				
Math											
FSQRT		1	0	43	0	43	1	fp	div	87	g, h
FLDPI, etc.		2	0			3	1	fp		87	
FSIN		6	≈150	≈180		≈170	1	fp		387	
FCOS		6		≈207		≈207	1	fp		387	
FSINCOS		7		~207 ≈216						387	
						≈211	1	fp			
FPTAN		6		≈230		≈200	1	fp		87	
FPATAN		3		≈187		≈153		fp		87	
FSCALE		3	24	57		66	1	fp		87	
FXTRACT		3	15	20		20	1	fp		87	
F2XM1		3	45	≈165		63	1	fp		87	
FYL2X		3	60	≈200		90	1	fp		87	
FYL2XP1		11		≈242		≈220	1	fp		87	
		· •						· I=			
1	1		I	ı 1		1	I		1		1 1

Other									
FNOP	1	0	1	0	1	0	mov	87	
(F)WAIT	2	0	0	0	1	0	mov	87	
FNCLEX	4	4			96	1		87	
FNINIT	6	29			172			87	
FNSAVE	4	174	456		420	0,1		87	
FRSTOR	4	96	528		532			87	
FXSAVE	4	69	132		96			sse	i
FXRSTOR	4	94	208		208			sse	i

Notes:

e) Not available on PMMX

The latency for FLDCW is 3 when the new value loaded is the same as the value of the control word before the preceding FLDCW, i.e. when alternating

between the same two values. In all other cases, the latency and reciprocal

throughput is 143.

g) Latency and reciprocal throughput depend on the precision setting in the F.P.

control word. Single precision: 23, double precision: 38, long double precision

(default): 43.

h) Throughput of FP-MUL unit is reduced during the use of the FP-DIV unit.

i) Takes 6 μops more and 40-80 clocks more when XMM registers are disabled.

Integer MMX and XMM instructions

Instruction	Operands	pops	Microcode	Latency	Additional latency	Reciprocal through-	Port	Execution unit	Subunit	Instruction set	Notes
					ÿ	igh-					
Move instructions				_			_	_			
MOVD	r32, mm	2	0	5	1	1	0	fp		mmx	
MOVD	mm, r32	2	0	2	0	2	1	mmx	alu	mmx	
MOVD	mm,m32	1	0	≈ 8	0	1	2	load		mmx	
MOVD	r32, xmm	2	0	10	1	2	0	fp		sse2	
MOVD	xmm, r32	2	0	6	1	2	1	mmx	shift	sse2	
MOVD	xmm,m32	1	0	≈ 8	0	1	2	load		sse2	
MOVD	m32, r	2	0	≈ 8		2	0,1			mmx	
MOVQ	mm,mm	1	0	6	0	1	0	mov		mmx	
MOVQ	xmm,xmm	1	0	2	1	2	1	mmx	shift	sse2	
MOVQ	r,m64	1	0	≈ 8		1	2	load		mmx	
MOVQ	m64,r	2	0	≈ 8		2	0	mov		mmx	
MOVDQA	xmm,xmm	1	0	6	0	1	0	mov		sse2	
MOVDQA	xmm,m	1	0	≈ 8		1	2	load		sse2	
MOVDQA	m,xmm	2	0	≈ 8		2	0	mov		sse2	
MOVDQU	xmm,m	4	0			2	2	load		sse2	k
MOVDQU	m,xmm	4	6			2	0	mov		sse2	k
MOVDQ2Q	mm,xmm	3	0	8	1	2	0,1	mov-mmx	sse2		
MOVQ2DQ	xmm,mm	2	0	8	1	2	0,1	mov-mmx	sse2		
MOVNTQ	m,mm	3	0			75	0	mov		sse	
MOVNTDQ	m,xmm	2	0			18	0	mov		sse2	

DACKECIA/D/DIA/	1 1						1				1 1
PACKSSWB/DW PACKUSWB	mm,r/m	1	0	2	1	1	1	mmx	shift	mmx	а
PACKSSWB/DW	111111,1/111	'	U		'	'	'	111111	SHILL	1111117	а
PACKUSWB	xmm,r/m	1	0	4	1	2	1	mmx	shift	mmx	а
PUNPCKH/LBW/WD/	AIIIII,1/111	'	U	_	'	_	'	1111111	Sillit	1111117	a
DQ	mm,r/m	1	0	2	1	1	1	mmx	shift	mmx	а
PUNPCKHBW/WD/DQ/	111111,17111	'			'	'	'	111111	Silit	1111111	a
QDQ	xmm,r/m	1	0	4	1	2	1	mmx	shift	sse2	а
PUNPCKLBW/WD/DQ/	Z,,,,,,,,	·		•	•	_	•	IIIIIX	Ormic	0002	u
QDQ	xmm,r/m	1	0	2	1	2	1	mmx	shift	sse2	а
PSHUFD	xmm,xmm,i	1	0	4	1	2	1	mmx	shift	sse2	
PSHUFL/HW	xmm,xmm,i	1	0	2	1	2	1	mmx	shift	sse2	
PSHUFW	mm,mm,i	1	0	2	1	1	1	mmx	shift	mmx	
MASKMOVQ	mm,mm	4	4	_	'	7	Ö	mov	Sillit	sse	
MASKMOVDQU		4	6			10	0			sse2	
	xmm,xmm	2	0	7	4	3	_	mov	000	5562	
PMOVMSKB	r32,r		_	_	1		0,1	mmx-alu0	sse		
PEXTRW	r32,mm,i	3	0	8	1	2	1	mmx-int	sse		
PEXTRW	r32,xmm,i	3	0	9	1	2	1	mmx-int	sse2		
PINSRW	mm,r32,i	2	0	3	1	2	1	int-mmx	sse		
PINSRW	xmm,r32,i	2	0	4	1	2	1	int-mmx	sse2		
Arithmetic instructions											
PADDB/W/D											
PADD(U)SB/W	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
PSUBB/W/D											
PSUB(U)SB/W	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
PADDQ, PSUBQ	mm,r/m	1	0	2	1	1	1	mmx	alu	sse2	a
PADDQ, PSUBQ	xmm,r/m	1	0	4	1	2	1	fp	add	sse2	а
PCMPEQB/W/D											
PCMPGTB/W/D	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
PMULLW PMULHW	r,r/m	1	0	6	1	1,2	1	fp	mul	mmx	a,j
PMULHUW	r,r/m	1	0	6	1	1,2	1	fp	mul	sse	a,j
PMADDWD	r,r/m	1	0	6	1	1,2	1	fp	mul	mmx	a,j
PMULUDQ	r,r/m	1	0	6	1	1,2	1	fp	mul	sse2	a,j
PAVGB/W	r,r/m	1	0	2	1	1,2	1	mmx	alu	sse	a,j
PMIN/MAXUB	r,r/m	1	0	2	1	1,2	1	mmx	alu	sse	a,j
PMIN/MAXSW	r,r/m	1	0	2	1	1,2	1	mmx	alu	sse	a,j
PSADBW	r,r/m	1	0	4	1	1,2	1	mmx	alu	sse	a,j
Logic	<u> </u>		_								
PAND, PANDN	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
POR, PXOR	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
PSLL/RLW/D/Q,			_	_	_		_				
PSRAW/D	r,i/r/m	1	0	2	1	1,2	1	mmx	shift	mmx	a,j
PSLLDQ, PSRLDQ	xmm,i	1	0	4	1	2	1	mmx	shift	sse2	а
Other											
EMMS	1	4	11	12		12	0			mmx	
		•						l			

Notes:

Add 1 µop if source is a memory operand. a)

j) k) Reciprocal throughput is 1 for 64 bit operands, and 2 for 128 bit operands.

It may be advantageous to replace this instruction by two 64-bit moves

Floating point XMM instructions

Instruction	Operands	sdon	Microcode	Latency	Additional latency	Reciprocal through- put	Port	Execution unit	Subunit	Instruction set	Notes
Move instructions											
MOVAPS/D	r,r	1	0	6	0	1	0	mov		sse	
MOVAPS/D	r,m	1	0	≈ 7	0	1	2			sse	
MOVAPS/D	m,r	2	0	≈ 7		2	0			sse	
MOVUPS/D	r,r	1	0	6	0	1	0	mov		sse	
MOVUPS/D	r,m	4	0			2	2			sse	k
MOVUPS/D	m,r	4	6			8	0			sse	k
MOVSS	r,r	1	0	2	0	2	1	mmx	shift	sse	
MOVSD	r,r	1	0	2	1	2	1	mmx	shift	sse	
MOVSS, MOVSD	r,m	1	0	≈ 7	0	1	2			sse	
MOVSS, MOVSD	m,r	2	0			2	0			sse	
MOVHLPS	r,r	1	0	4	0	2	1	mmx	shift	sse	
MOVLHPS	r,r	1	0	2	0	2	1	mmx	shift	sse	
MOVHPS/D, MOVLPS/D	','	'		-		_			3		
VIO VIII O/D, IVIO VEI O/D	r,m	3	0			4	2			sse	
MOVHPS/D, MOVLPS/D	,										
	m,r	2	0			2	0			sse	
MOVNTPS/D	m,r	2	0			4	0			sse/2	
MOVMSKPS/D	r32,r	2	0	6	1	3	1	fp		sse	
SHUFPS/D	r,r/m,i	1	0	4	1	2	1	mmx	shift	sse	
UNPCKHPS/D	r,r/m	1	0	4	1	2	1	mmx	shift	sse	
UNPCKLPS/D	r,r/m	1	0	2	1	2	1	mmx	shift	sse	
Conversion											
CVTPS2PD	r,r/m	4	0	7	1	4	1	mmx	shift	sse2	а
CVTPD2PS	r,r/m	2	0	10	1	2	1	fp-mmx	sse2	а	
CVTSD2SS	r,r/m	4	0	14	1	6	1	mmx	shift	sse2	а
CVTSS2SD	r,r/m	4	0	10	1	6	1	mmx	shift	sse2	а
CVTDQ2PS	r,r/m	1	0	4	1	2	1	fp		sse2	а
CVTDQ2PD	r,r/m	3	0	9	1	4	1	mmx-fp	sse2	а	
CVT(T)PS2DQ	r,r/m	1	0	4	1	2	1	fp		sse2	а
CVT(T)PD2DQ	r,r/m	2	0	9	1	2	1	fp-mmx	sse2	а	
CVTPI2PS	xmm,mm	4	0	10	1	4	1	mmx		sse	а
CVTPI2PD	xmm,mm	4	0	11	1	5	1	fp-mmx	sse2	а	
CVT(T)PS2PI	mm,xmm	3	0	7	0	2	0,1	fp-mmx	sse	а	
CVT(T)PD2PI	mm,xmm	3	0	11	1	3	0,1	fp-mmx	sse2	а	
CVTSI2SS	xmm,r32	3	0	10	1	3	1	fp-mmx	sse	a	
CVTSI2SD	xmm,r32	4	0	15	1	6	1	fp-mmx	sse2	a	
CVT(T)SD2SI	r32,xmm	2	0	8	1	2.5	1	fp		sse2	а
CVT(T)SS2SI	r32,xmm	2	0	8	1	2.5	1	fp		sse	a
Arithmetic											
ADDPS/D ADDSS/D	r,r/m	1	0	4	1	2	1	fp	add	sse	а
SUBPS/D SUBSS/D	r,r/m	1	0	4	1	2	1	fp	add	sse	a

MULPS/D MULSS/D	r,r/m	1	0	6	1	2	1	fp	mul	sse	а
DIVSS	r,r/m	1	0	23	0	23	1	fp	div	sse	a,h
DIVPS	r,r/m	1	0	39	0	39	1	fp	div	sse	a,h
DIVSD	r,r/m	1	0	38	0	38	1	fp	div	sse2	a,h
DIVPD	r,r/m	1	0	69	0	69	1	fp	div	sse2	a,h
RCPPS RCPSS	r,r/m	2	0	4	1	4	1	mmx		sse	а
MAXPS/D MAXSS/											
DMINPS/D MINSS/D	r,r/m	1	0	4	1	2	1	fp	add	sse	а
CMPccPS/D											
CMPccSS/D	r,r/m	1	0	4	1	2	1	fp	add	sse	а
COMISS/D UCOMISS/D	r,r/m	2	0	6	1	3	1	fp	add	sse	а
Logic											
ANDPS/D ANDNPS/D											
ORPS/D XORPS/D	r,r/m	1	0	2	1	2	1	mmx	alu	sse	а
Math											
SQRTSS	r,r/m	1	0	23	0	23	1	fp	div	sse	a,h
SQRTPS	r,r/m	1	0	39	0	39	1	fp	div	sse	a,h
SQRTSD	r,r/m	1	0	38	0	38	1	fp	div	sse2	a,h
SQRTPD	r,r/m	1	0	69	0	69	1	fp	div	sse2	a,h
RSQRTSS	r,r/m	2	0	4	1	3	1	mmx		sse	а
RSQRTPS	r,r/m	2	0	4	1	4	1	mmx		sse	а
Other											
LDMXCSR	m	4	8	98		100	1			sse	
STMXCSR	m	4	4			6	1			sse	

Notes:

a) Add 1 µop if source is a memory operand.

h) Throughput of FP-MUL unit is reduced during the use of the FP-DIV unit.

k) It may be advantageous to replace this instruction by two 64-bit moves.

Intel Pentium 4 w. EM64T (Prescott)

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Instruction name. cc means any condition code. For example, Jcc can be JB,

JNE, etc.

Operands: i = immediate constant, r = any register, r32 = 32-bit register, etc., mm = 64 bit

mmx register, xmm = 128 bit xmm register, sr = segment register, m = any memory operand including indirect operands, m64 means 64-bit memory oper-

and, etc., mabs = memory operand with 64-bit absolute address.

μορs: Number of μορs issued from instruction decoder and stored in trace cache.

Microcode: Number of additional μops issued from microcode ROM.

Latency: This is the delay that the instruction generates in a dependency chain if the next

dependent instruction starts in the same execution unit. The numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latency of moves to and from memory cannot be measured accurately because of the problem with memory intermediates explained above under

"How the values were measured".

Additional latency: This number is added to the latency if the next dependent instruction is in a dif-

ferent execution unit. There is no additional latency between ALU0 and ALU1.

ReciprocalThis is also called issue latency. This value indicates the number of clock cycles throughput:
from the execution of an instruction begins to a subsequent independent in-

from the execution of an instruction begins to a subsequent independent instruction can begin to execute in the same execution subunit. A value of 0.25

indicates 4 instructions per clock cycle in one thread.

Port: The port through which each μop goes to an execution unit. Two independent

μops can start to execute simultaneously only if they are going through different

ports.

Execution unit: Use this information to determine additional latency. When an instruction with

more than one µop uses more than one execution unit, only the first and the

last execution unit is listed.

Execution subunit: Throughput measures apply only to instructions executing in the same subunit.

Indicates the compatibility of an instruction with other 80x86 family micropro-

cessors. The instruction can execute on microprocessors that support the in-

struction set indicated.

Integer instructions

Instruction	Operands	pops	Microcode	Latency	Additional latency	Reciprocal through- put	Port	Execution unit	Subunit	Instruction set	Notes
Move instructions											
MOV	r,r	1	0	1	0	0.25	0/1	alu0/1		86	С
MOV	r8/16/32,i	1	0	1	0	0.25	0/1	alu0/1		86	
MOV	r64,i32	1	0		0	0.5	0/1	alu0/1		x64	
MOV	r64,i64	2	0		0	1	1	alu1		x64	

				10000	J. (.						
MOV	r8/16,m	2	0	3	0	1	2	load		86	
MOV	r32/64,m	1	0	2	0	1	2	load		86	
MOV	m,r	1	0			2	0	store		86	b,c
MOV	m,i	2	0			2	0,3	store		86	
MOV	m64,i32	2	0			2	0,3	store		x64	
MOV	r,sr	1	2			8	- , -			86	
MOV	sr,r/m	1	8			27				86	a,q
MOV	r,mabs	3	0			1				x64	","
MOV	mabs,r	3	0			2				x64	i
MOVNTI	m,r32	2	0			2				sse2	'
MOVZX	r,r	1	0	1	0	0.25	0/1	alu0/1		386	С
MOVZX	r16,r8	2	0	2	0	1	0/1	alu0/1		386	C
MOVZX	r,m	1	0	2	0	1	2	load		386	
MOVSX	r16,r8	2	0	2	0	1	0	alu0		386	a,c,o
MOVSX	r32/64,r8/16	1	0	1	0	0.5	0	alu0 alu0		386	
MOVSX		2	0	3	0	1	2	load		386	a,c,o
MOVSXD	r,m	1	0	1	0	0.5	0			x64	
	r64,r32		_				U	alu0			a
CMOVcc	r,r/m	3	0	9.5	0	3	0/4	-10/4		PPro	a,e
XCHG	r,r	3	0	2	0	1	0/1	alu0/1		86	
XCHG	r,m	2	6	≈100						86	
XLAT		4	0	6						86	
PUSH	r	2	0	2		2				86	
PUSH	i	2	0	2		2				186	
PUSH	m	3	0	2		2				86	
PUSH	sr	1	3			9				86	
PUSHF(D/Q)		1	3			9				86	
PUSHA(D)		1	9			16				186	m
POP	r	2	0	1	0	1				86	
POP	m	2	6			10				86	
POP	sr	1	8			30				86	
POPF(D/Q)		1	8			70				86	
POPA(D)		2	16			15				186	m
LEA	r,[m]	1	0			0.25	0/1	alu0/1		86	р
LEA	r,[r+r/i]	1	0	2.5	0	0.25	0/1	alu0/1		86	
LEA	r,[r+r+i]	2	0	3.5	0	0.5	0/1	alu0/1		86	
LEA	r,[r*i]	3	0	3.5	0	1	1	alu		386	
LEA	r,[r+r*i]	2	0	3.5	0	1	0,1	alu0,1		386	
LEA	r,[r+r*i+i]	3	0	3.5	0	1	1	alu		386	
LAHF		1	0	4	0		1	int		86	n
SAHF		1	0	5	0		0/1	alu0/1		86	d,n
SALC		2	0		0	1	1	int		86	m
LDS, LES,	r,m	2	10			28				86	m
LODS		1	3	8		8				86	
REP LODS		1	5n	≈ 4n-	-50			86			
STOS		1	2	8		8				86	
REP STOS		1	2.5n	≈ 3n				86			
MOVS		1	4	8		8				86	
REP MOVSB		9	≈.3n					86			
REP MOVSW		1	≈.5-1.1r		1			86			
REP MOVSD		1	≈1.1n					86			
REP MOVSQ		1	≈1.1n					x64			
BSWAP	r	1	0	1	0	1		alu		486	
IN, OUT	r,r/i	1	52			>100	0		86		
1 '	· ' !		1	1	I	1		ı	ı	1	1 1

	I	١.	۱ ـ	ı	I		ı		1 1		1 1
PREFETCHNTA	m	1	0			1				sse	
PREFETCHT0/1/2	m	1	0			1				sse	
SFENCE		1	2			50				sse	
LFENCE		1	2			50				sse2	
MFENCE		1	4			124				sse2	
Arithmetic instructions					_						
ADD, SUB	r,r	1	0	1	0	0.25	0/1	alu0/1		86	С
ADD, SUB	r,m	2	0	1	0	1				86	С
ADD, SUB	m,r	3	0	5	_	2				86	С
ADC, SBB	r,r/i	3	0	10	0	10	1	int,alu		86	
ADC, SBB	r,m	2	5	10	0	10	1	int,alu		86	
ADC, SBB	m,r	2	6	20		10				86	
ADC, SBB	m,i	3	5	22		10				86	
CMP	r,r	1	0	1	0	0.25	0/1	alu0/1		86	С
CMP	r,m	2	0	1	0	1				86	С
INC, DEC	r	2	0	1	0	0.5	0/1	alu0/1		86	
INC, DEC	m	4	0	5		3				86	
NEG	r	1	0	1	0	0.5	0	alu0		86	
NEG	m	3	0	5		3				86	
AAA, AAS		1	10	26						86	m
DAA, DAS		1	16	29						86	m
AAD		2	5	13			1	int	mul	86	m
AAM		2	17	71			1	int	fpdiv	86	m
MUL, IMUL	r8	1	0	10	0		1	int	mul	86	
MUL, IMUL	r16	4	0	11	0		1	int	mul	86	
MUL, IMUL	r32	3	0	11	0		1	int	mul	86	
MUL, IMUL	r64	1	5	11	0		1	int	mul	x64	
MUL, IMUL	m8	2	0	10	0		1	int	mul	86	
MUL, IMUL	m16	2	5	11	0		1	int	mul	86	
MUL, IMUL	m32	3	0	11	0		1	int	mul	86	
MUL, IMUL	m64	2	6	11	0		1	int	mul	x64	
IMUL	r16,r16	1	0	10	0	2.5	1	int	mul	386	
IMUL	r16,r16,i	2	0	11	0	2.5	1	int	mul	186	
IMUL	r32,r32	1	0	10	0	2.5	1	int	mul	386	
IMUL	r32,(r32),i	1	0	10	0	2.5	1	int	mul	386	
IMUL	r64,r64	1	0	10	0	2.5	1	int	mul	x64	
IMUL	r64,(r64),i	1	0	10	0	2.5	1	int	mul	x64	
IMUL	r16,m16	2	0	10	0	2.5	1	int	mul	386	
IMUL	r32,m32	2	0	10	0	2.5	1	int	mul	386	
IMUL	r64,m64	2	0	10	0	2.5	1	int	mul	x64	
IMUL	r,m,i	3	0	10	0	1-2.5	1	int	mul	186	
DIV	r8/m8	1	20	74	0	34	1	int	fpdiv	86	а
DIV	r16/m16	1	19	73	0	34	1	int	fpdiv	86	а
DIV	r32/m32	1	21	76	0	34	1	int	fpdiv	386	а
DIV	r64/m64	1	31	63	0	52	1	int	fpdiv	x64	а
IDIV	r8/m8	1	21	76	0	34	1	int	fpdiv	86	a
IDIV	r16/m16	1	19	79	0	34	1	int	fpdiv	86	a
IDIV	r32/m32	1	19	79	0	34	1	int	fpdiv	386	a
IDIV	r64/m64	1	58	96	0	91	1	int	fpdiv	x64	a
CBW		2	0	2	0	1	0	alu0	'	86	
CWD		2	0	2	0	1	0/1	alu0/1		86	
CDQ		1	0	1	0	1	0/1	alu0/1		386	
	I	1 -	-	1 -		1 -			1		1 I

CQO CWDE CDQE SCAS REP SCAS CMPS REP CMPS		1 2 1 1 1 1	0 0 0 3 ≈ 54+6 5 ≈ 81+8		0 0 0 0 ≈ 4n ≈ 5n	1 1 1 8 8 10	0/1 0/1 0/1	alu0/1 alu0/1 alu0/1	86 86	x64 386 x64 86	
AND, OR, XOR AND, OR, XOR AND, OR, XOR TEST TEST NOT NOT SHL SHR, SAR SHR, SAR SHR, SAR	r,r r,m m,r r,r r,m r m r,i r8/16/32,i r64,i r,CL	1 2 3 1 2 1 3 1 1 1 2 2 2	0 0 0 0 0 0 0 0	1 1 5 1 1 1 5 1 7 2	0 0 0 0 0 0	0.5 1 2 0.5 1 0.5 2 0.5 0.5 2 2 2	0 0 1 1 1 1 1 1 1	alu0 alu0 alu0 alu1 alu1 alu1 alu1 alu1		86 86 86 86 86 186 186 x64 86	C C C C
SHR, SAR ROL, ROR ROL, ROR ROL, ROR ROL, RCR RCL, RCR RCL RCR SHL, SHR, SAR ROL. ROR SHL, SHR, SAR ROL. ROR RCL, RCR RCL, RCR RCL, RCR RCL, RCR RCL, RCR RCL, RCR RCL, RCR SHLD, SHRD	r64,CL r8/16/32,i r64,i r8/16/32,CL r64,CL r,1 r,i r,CL r,CL m8/16/32,i m8/16/32,cl m8/16/32,cl m8/16/32,cl m8/16/32,ri m8/16/32,ri	2 1 1 2 2 1 1 2 2 1 1 3 3 2 2 2 3 2 3 2	0 0 0 0 0 11 11 11 6 6 6 6 5 13	8 1 7 2 8 7 31 25 31 25 10 10 10 27 38 37 8		1 7 2 8 7 31 25 31 25 25 31 25	1 1 1 1 1 1 1 1 1 1 1 1 1	alu1 alu1 alu1 alu1 alu1 alu1 alu1 alu1		x64 186 x64 86 x64 86 186 186 86 86 86 86 86 86 86	d d d d d d d d d d
SHLD, SHRD SHRD SHRD SHLD, SHRD SHLD, SHRD SHLD, SHRD SHLD, SHRD SHLD, SHRD BT BT BT BT BT BT BT BT, BTS, BTC BTR, BTS, BTC BTR, BTS, BTC	r6/10/32,r,1 r64,r64,i r64,r64,i r8/16/32,r,cl r64,r64,cl r64,r64,cl m,r,i m,r,CL r,i r,r m,i m,r r,i r,r	3 4 3 4 4 3 3 2 1 2 3 2 1 2 3	5 7 0 5 8 8 8 0 0 7 0 6	10 10 9 14 12 20 20 8 9 8 10 8 9 28	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	8 10 10 8 9 8 10 8 9	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	alu 1 alu1 alu1 alu1 alu1 alu1 alu1 alu1 alu		386 x64 x64 x64 x64 386 386 386 386 386 386 386 386	d d d

BTR, BTS, BTC	m,r	2	10	14	0	14	1	alu1		386	
BSF, BSR	r,r/m	2	0	16	0	4	1	alu1		386	
SETcc	r	2	0	9	0	1	1	int		386	
SETcc	m .	3	0	9	0	2	1	int		386	
CLC, STC		2	0	0	0	8				86	d
CMC		3	0	15	0					86	٦ <u> </u>
		1	8	13	0	53				86	
CLD, STD		I	0		U	55				00	
Control transfer instruct	l tions										
JMP	short/near	1	0	0	0	1	0	alu0	branch	86	
JMP	far	2	25			154	0			86	m
JMP	r	3	0			15	0	alu0	branch	86	
JMP	m(near)	3	0			10	0	alu0	branch	86	
JMP	m(far)	2	28			157	0	5.1.5.5		86	
Jcc	short/near	1	0			2-4	0	alu0	branch	86	
J(E)CXZ	short	4	0			4	0	alu0	branch	86	
LOOP	short	4	0			4	0	alu0	branch	86	
CALL	near	3	0			7	0	alu0	branch	86	
CALL	far	3	29			160	0	aido	branon	86	m
CALL	r	4	0			7	0	alu0	branch	86	
CALL	m(near)	4	0			9	0	alu0	branch	86	
CALL	m(far)	2	32			160	0	aido	branon	86	
RETN	in(idi)	4	0			7	0	alu0	branch	86	
RETN	i	4	0			7	0	alu0	branch	86	
RETF		1	30			160	0	aido	branon	86	
RETF	i	2	30			160	0			86	
IRET		1	49			325	0			86	
BOUND	m	2	11			12	U			186	m
INT	m i	2	67			470				86	111
INTO	l l	1	4			26				86	m
		'	-			20				00	'''
Other											
NOP (90)		1	0	0		0.25	0/1	alu0/1		86	
Long NOP (0F 1F)		1	0	0		0.25	0/1	alu0/1		ppro	
PAUSE		1	2	U		50	0/1	aldo/ i		sse2	
LEAVE		4	0	5		5				186	
CLI		1	5	3		52				86	
STI		1	11			64				86	
CPUID			49-90		300-5	1		n.E		00	
RDTSC			12		300-0			p5		n.E	
	4	27	12		100	100			n.E	p5	
RDPMC (bit 31 = 1)	1	37			100				p5		
RDPMC (bit 31 = 0)	4	154			240				p5	(0)	
MONITOR										(sse3)	
MWAIT										(sse3)	

Notes:

a) Add 1 μop if source is a memory operand.

b) Uses an extra μop (port 3) if SIB byte used.

c) Add 1 μop if source or destination, but not both, is a high 8-bit register (AH, BH, CH, DH).

d) Has (false) dependence on the flags in most cases.

e) Not available on PMMX

Move accumulator to/from memory with 64 bit absolute address (opcode A0 - A3).

m) Not available in 64 bit mode.

Not available in 64 bit mode on some processors. n)

o) MOVSX uses an extra uop if the destination register is smaller than the biggest register size available. Use a 32 bit destination register in 16 bit and 32 bit

mode, and a 64 bit destination register in 64 bit mode for optimal performance.

LEA with a direct memory operand has 1 µop and a reciprocal throughput of 0.25. This also applies if there is a RIP-relative address in 64-bit mode. A signextended 32-bit direct memory operand in 64-bit mode without RIP-relative address takes 2 µops because of the SIB byte. The throughput is 1 in this case.

You may use a MOV instead.

These values are measured in 32-bit mode. In 16-bit real mode there is 1 miq)

crocode µop and a reciprocal throughput of 17.

Floating point x87 instructions

p)

Instruction	Operands	pops	Microcode	Latency	Addition	Reciproc put	Port	Execution unit	Subunit	Instruction set	Notes
			de		Additional latency	Reciprocal through- put		n unit		on set	
Move instructions											
FLD	r	1	0	7	0	1	0	mov		87	
FLD	m32/64	1	0		0	1	2	load		87	
FLD	m80	3	3			8	2	load		87	
FBLD	m80	3	74			90	2	load		87	
FST(P)	r	1	0	7	0	1	0	mov		87	
FST(P)	m32/64	2	0	7		2	0	store		87	
FSTP	m80	3	6			10	0	store		87	
FBSTP	m80	3	311			400	0	store		87	
FXCH	r	1	0	0	0	1	0	mov		87	
FILD	m16	3	2			8	2	load		87	
FILD	m32/64	2	0			2	2	load		87	
FIST(P)	m	3	0			2.5	0	store		87	
FISTTP	m	3	0			2.5	0	store		sse3	
FLDZ		1	0			2	0	mov		87	
FLD1		2	0			2	0	mov		87	
FCMOVcc	st0,r	4	0	5	1	4	1	fp		PPro	е
FFREE	r	3	0			3	0	mov		87	
FINCSTP, FDECSTP		1	0	0	0	1	0	mov		87	
FNSTSW	AX	4	0		0	3	1			287	
FSTSW	AX	6	0		0	3	1			287	
FNSTSW	m16	2	3			8	0			87	
FNSTCW	m16	4	0			3	0			87	
FLDCW	m16	3	6			10	0,2			87	f
Arithmetic instructions											
FADD(P),FSUB(R)(P)	r	1	0	6	1	1	1	fp	add	87	
FADD,FSUB(R)	m	2	0	6	1	1	1	fp	add	87	
FIADD,FISUB(R)	m16	3	3	7	1	6	1	fp	add	87	
FIADD,FISUB(R)	m32	3	0	6	1	2	1	fp	add	87	

FMUL(P) FMUL FIMUL FIMUL FDIV(R)(P) FDIV(R) FIDIV(R) FIDIV(R) FABS FCHS FCOM(P), FUCOM(P) FCOM(P) FCOMPP, FUCOMPP FCOMI(P) FICOM(P) FICOM(P) FICOM(P) FICOM(P) FTST FXAM FRNDINT FPREM FPREM1	r m16 m32 r m16 m32 r m	1 2 3 3 1 1 2 2 3 3 3 1 1 3 8 9	0 0 3 0 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0	8 8 8 45 45 45 45 3 3 3 3 3 28 220 220	1 1 1 1 1 1 1 1 0 0	2 8 3 45 45 45 1 1 1 1 3 8 2 1 1 16	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	fp fp fp fp fp fp fp fp fp fp fp fp	mul mul mul div div div div misc misc misc misc misc misc misc misc	87 87 87 87 87 87 87 87 87 87 87 87 87 8	g,h g,h g,h
		9	92	220	ı		I	ιρ		367	
Math FSQRT FLDPI, etc. FSIN, FCOS FSINCOS FPTAN FPATAN FSCALE FXTRACT F2XM1 FYL2X FYL2XP1 Other		1 2 3 5 8 4 3 4 3 3 3	0 0 ≈100 ≈150 ≈170 97 25 16 190 63 58	45 ≈200 ≈200 ≈270 ≈250 96 27 ≈270 ≈170 ≈170		45 2 ≈200 ≈200 ≈270 ≈250	1 1 1 1 1 1 1 1 1	fp fp fp fp fp fp fp	div	87 87 387 387 87 87 87 87 87	g,h
FNOP (F)WAIT FNCLEX FNINIT FNSAVE FRSTOR FXSAVE FXRSTOR		1 2 1 1 2 2 2 2	0 0 4 30 181 96 121 118	1 0 500 570	0	1 1 120 200 160 244	0 0 1 0,1		mov mov	87 87 87 87 87 87 sse sse	i i

Notes:

e) Not available on PMMX

The latency for FLDCW is 3 when the new value loaded is the same as the value of the control word before the preceding FLDCW, i.e. when alternating between the same two values. In all other cases, the latency and reciprocal throughput is > 100.

g)
Latency and reciprocal throughput depend on the precision setting in the F.P. control word. Single precision: 32, double precision: 40, long double precision (default): 45.

h) Throughput of FP-MUL unit is reduced during the use of the FP-DIV unit.

Takes fewer microcode μops when XMM registers are disabled, but the throughput is the same.

Integer MMX and XMM instructions

Instruction	Operands	sdon	Mic	Lat	Adc	Rec	Port	Exe	Sub	Inst	Notes
		Š	Microcode	Latency	Additional latency	Reciprocal through- put	_	Execution unit	Subunit	Instruction set	es
			<u>6</u>		al la	al t				on s	
					ten	hro		≓		e	
					cy	ugh					
Move instructions						'					
MOVD	r32, mm	2	0	6	1	1	0	fp		mmx	
MOVD	mm, r32	1	0	3	1	1	1	mmx	alu	mmx	
MOVD	mm,m32	1	0		'	1	2	load	aiu	mmx	
MOVD	r32, xmm	1	0	7	1	1	0	fp		sse2	
MOVD	xmm, r32	2	0	4		2	1	mmx	shift	sse2	
MOVD	xmm,m32	1	0	*	'	1	2	load	Sillit	sse2	
MOVD	m32, r	2	0			2	0,1	loau		mmx	
MOVQ		1	0	7	0	1	0,1	mov		mmx	
MOVQ	mm,mm		0	2	1	2	1	mmx	shift	sse2	
MOVQ	xmm,xmm r,m64		0	-	'	1	2		Sillit		
MOVQ	m64,r	2	0			2	0	load		mmx	
MOVDQA	1	1	0	7	0	1	0	mov		mmx	
MOVDQA	xmm,xmm		0	'	0	1	-	mov		sse2	
	xmm,m	1	-			-	2	load		sse2	
MOVDQA	m,xmm	2	0			2 23	0	mov		sse2	l.
MOVDQU	xmm,m	4	0				2	load		sse2	k
MOVDQU	m,xmm	4	2			8	0	mov		sse2	k
LDDQU	xmm,m	4	0	40	١,	2.5	2	load		sse3	
MOVDQ2Q	mm,xmm	3	0	10	1	2	0,1	mov-mmx	sse2		
MOVQ2DQ	xmm,mm	2	0	10	1	2	0,1	mov-mmx	sse2		
MOVNTQ	m,mm	3	0			4	0	mov		sse	
MOVNTDQ	m,xmm	2	0			4	0	mov		sse2	
MOVDDUP	xmm,xmm	1	0	2	1	2	1	mmx	shift	sse3	
MOVSHDUP					١,				16		
MOVSLDUP	xmm,xmm	1	0	4	1	2	1	mmx	shift	sse3	
PACKSSWB/DW		,			4		_		- 1- :61		_
PACKUSWB	mm,r/m	1	0	2	1	2	1	mmx	shift	mmx	а
PACKSSWB/DW	vmm r/m	1	0	1	1	4	4	mmy	ohift	mmy	
PACKUSWB	xmm,r/m		0	4	1	4	1	mmx	shift	mmx	а
PUNPCKH/LBW/WD/ DQ	mm,r/m	1	0	2	1	2	1	mmx	shift	mmx	а
PUNPCKHBW/WD/DQ/	111111,17111	'	0	-	'		'	IIIIIA	Sillit	1111117	a
QDQ	xmm,r/m	1	0	4	1	4	1	mmx	shift	sse2	а
PUNPCKLBW/WD/DQ/	XIIIII,1/111			-	'	-	'	IIIIIX	Jiiiit	3302	u
QDQ	xmm,r/m	1	0	2	1	2	1	mmx	shift	sse2	а
PSHUFD	xmm,xmm,i	1	0	4	1	2	1	mmx	shift	sse2	
PSHUFL/HW	xmm,xmm,i	1	0	2	1	2	1	mmx	shift	sse	
PSHUFW	mm,mm,i	1	0	2	1	1	1	mmx	shift	sse	
MASKMOVQ	mm,mm	1	4	-	'	10	0	mov	31111	sse	
MASKMOVDQU	xmm,xmm		6			12	0	mov		sse2	
PMOVMSKB	r32,r	2	0	7		3	0,1		655	3362	
I MO MINISKD	132,1	4	l U	'	1	3	υ, ι	mmx-alu0	sse		1

PEXTRW	r32,mm,i	2	0	7		2	1	mmx-int	sse		
PEXTRW	r32,xmm,i	2	0	7		3	1	mmx-int	sse2		
PINSRW	r,r32,i	2	0	4		2	1	int-mmx	sse		
	1,102,1	_				_	•		555		
Arithmetic instructions											
PADDB/W/D											
PADD(U)SB/W	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
PSUBB/W/D											
PSUB(U)SB/W	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
PADDQ, PSUBQ	mm,r/m	1	0	2	1	1	1	mmx	alu	sse2	а
PADDQ, PSUBQ	xmm,r/m	1	0	5	1	2	1	fp	add	sse2	а
PCMPEQB/W/D											
PCMPGTB/W/D	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
PMULLW PMULHW	r,r/m	1	0	7	1	1,2	1	fp	mul	mmx	a,j
PMULHUW	r,r/m	1	0	7	1	1,2	1	fp	mul	sse	a,j
PMADDWD	r,r/m	1	0	7	1	1,2	1	fp	mul	mmx	a,j
PMULUDQ	r,r/m	1	0	7	1	1,2	1	fp	mul	sse2	a,j
PAVGB/W	r,r/m	1	0	2	1	1,2	1	mmx	alu	sse	a,j
PMIN/MAXUB	r,r/m	1	0	2	1	1,2	1	mmx	alu	sse	a,j
PMIN/MAXSW	r,r/m	1	0	2	1	1,2	1	mmx	alu	sse	a,j
PSADBW	r,r/m	1	0	4	1	1,2	1	mmx	alu	sse	a,j
Logic											
PAND, PANDN	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
POR, PXOR	r,r/m	1	0	2	1	1,2	1	mmx	alu	mmx	a,j
PSLL/RLW/D/Q,											
PSRAW/D	r,i/r/m	1	0	2	1	1,2	1	mmx	shift	mmx	a,j
PSLLDQ, PSRLDQ	xmm,i	1	0	4	1	2	1	mmx	shift	sse2	
Other											
EMMS		10	10			12	0			mmx	

Notes:

a) Add 1 µop if source is a memory operand.

Reciprocal throughput is 1 for 64 bit operands, and 2 for 128 bit operands. j) k)

It may be advantageous to replace this instruction by two 64-bit moves or LD-

DQU.

Floating point XMM instructions

Instruction	Operands	sdon	Microcode	Latency	Additional latency	Reciprocal through- put	Port	Execution unit	Subunit	Instruction set	Notes
Move instructions											
MOVAPS/D	r,r	1	0	7	0	1	0	mov		sse	
MOVAPS/D	r,m	1	0		0	1	2			sse	
MOVAPS/D	m,r	2	0			2	0			sse	
MOVUPS/D	r,r	1	0	7	0	1	0	mov		sse	
MOVUPS/D	r,m	4	0			2	2			sse	k
MOVUPS/D	m,r	4	2			8	0			sse	k

MOVSS	r,r	1	0	2	1	2	1	mmx	shift	sse	
MOVSD	r,r	1	0	4	1	2	1	mmx	shift	sse	
MOVSS, MOVSD	r,m	1	0		0	1	2			sse	
MOVSS, MOVSD	m,r	2	0			2	0			sse	
MOVHLPS	r,r	1	0	4	1	2	1	mmx	shift	sse	
MOVLHPS	r,r	1	0	2	1	2	1	mmx	shift	sse	
MOVHPS/D, MOVLPS/D	r,m	2	0	-	•	2	2			sse	
MOVHPS/D, MOVLPS/D	m,r	2	0			2	0			sse	
MOVSH/LDUP	r,r	1	0	4	1	2	1			sse3	
MOVDDUP	r,r	1	0	2	1	2	1			sse3	
MOVNTPS/D	m,r	2	0	-		4	0			sse	
MOVMSKPS/D	r32,r	2	0	5	1	3	1	fp		sse	
SHUFPS/D	r,r/m,i	1	0	4	1	2	1	mmx	shift	sse	
UNPCKHPS/D	r,r/m	2	0	4	1	2	1	mmx	shift	sse	
UNPCKLPS/D	r,r/m	1	0	2	1	2		mmx	shift	sse	
ON ONE OF	1,1/111	'			'	_	'	IIIIIX	Siliit	330	
Conversion											
CVTPS2PD	r,r/m	1	0	4	1	4	1	mmx	shift	sse2	а
CVTPD2PS	r,r/m	2	0	10	1	2	1	fp-mmx	sse2	а	
CVTSD2SS	r,r/m	3	0	14	1	6	1	mmx	shift	sse2	а
CVTSS2SD	r,r/m	2	0	8	1	6	1	mmx	shift	sse2	а
CVTDQ2PS	r,r/m	1	0	5	1	2	1	fp		sse2	а
CVTDQ2PD	r,r/m	3	0	10	1	4	1	mmx-fp	sse2	а	
CVT(T)PS2DQ	r,r/m	1	0	5	1	2	1	fp		sse2	а
CVT(T)PD2DQ	r,r/m	2	0	11	1	2	1	fp-mmx	sse2	а	
CVTPI2PS	xmm,mm	4	0	12	1	6	1	mmx		sse	а
CVTPI2PD	xmm,mm	4	0	12	1	5	1	fp-mmx	sse2	а	
CVT(T)PS2PI	mm,xmm	3	0	8	0	2	0,1	fp-mmx	sse	а	
CVT(T)PD2PI	mm,xmm	4	0	12	1	3	0,1	fp-mmx	sse2	а	
CVTSI2SS	xmm,r32	3	0	20	1	4	1	fp-mmx	sse	а	
CVTSI2SD	xmm,r32	4	0	20	1	5	1	fp-mmx	sse2	а	
CVT(T)SD2SI	r32,xmm	2	0	12	1	4	1	fp		sse2	а
CVT(T)SS2SI	r32,xmm	2	0	17	1	4	1	fp		sse	а
Arithmetic		١.	_	_		_		_			
ADDPS/D ADDSS/D	r,r/m	1	0	5	1	2	1	fp	add	sse	а
SUBPS/D SUBSS/D	r,r/m	1	0	5	1	2	1	fp	add	sse	а
ADDSUBPS/D	r,r/m	1	0	5	1	2	1	fp	add	sse3	а
HADDPS/D HSUBPS/D	r,r/m	3	0	13	1	5-6	1	fp	add	sse3	а
MULPS/D MULSS/D	r,r/m	1	0	7	1	2	1	fp	mul	sse	a
DIVSS	r,r/m	1	0	32	1	23	1	fp	div 	sse	a,h
DIVPS	r,r/m	1	0	41	1	41	1	fp	div	sse	a,h
DIVSD	r,r/m	1	0	40	1	40	1	fp	div	sse2	a,h
DIVPD	r,r/m	1	0	71	1	71	1	fp	div	sse2	a,h
RCPPS RCPSS	r,r/m	2	0	6	1	4	1	mmx		sse	а
MAXPS/D MAXSS/	,							_			
DMINPS/D MINSS/D	r,r/m	1	0	5	1	2	1	fp	add	sse	а
CMPccPS/D	r r/m	4	0	_	1	2	4	fn	244	600	
CMPccSS/D COMISS/D UCOMISS/D	r,r/m r,r/m	1 2	0	5 6	1	2	1	fp fp	add add	sse	a
COMISS/D OCOMISS/D	1,1/1/1	~	0	0		٥	'	ıρ	auu	sse	а
Logic											

ANDPS/D ANDNPS/D ORPS/D XORPS/D	r,r/m	1	0	2	1	2	1	mmx	alu	sse	а
Math											
SQRTSS	r,r/m	1	0	32	1	32	1	fp	div	sse	a,h
SQRTPS	r,r/m	1	0	41	1	41	1	fp	div	sse	a,h
SQRTSD	r,r/m	1	0	40	1	40	1	fp	div	sse2	a,h
SQRTPD	r,r/m	1	0	71	1	71	1	fp	div	sse2	a,h
RSQRTSS	r,r/m	2	0	5	1	3	1	mmx		sse	а
RSQRTPS	r,r/m	2	0	6	1	4	1	mmx		sse	а
Other											
LDMXCSR	m	2	11			13	1			sse	
STMXCSR	m	3	0			3	1			sse	

Notes:

a) Add 1 µop if source is a memory operand.

h) Throughput of FP-MUL unit is reduced during the use of the FP-DIV unit.

k) It may be advantageous to replace this instruction by two 64-bit moves or LDDQU.

Intel Atom

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Instruction name, cc means any condition code. For example, Jcc can be JB,

JNE, etc.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, xmm = 128 bit

xmm register, mm/x = mmx or xmm register, sr = segment register, m =

memory, m32 = 32-bit memory operand, etc.

μορs: The number of μops from the decoder or ROM.

Unit: Tells which execution unit is used. Instructions that use the same unit cannot

execute simultaneously.

ALU0 and ALU1 means integer unit 0 or 1, respectively.

ALU0/1 means that either unit can be used. ALU0+1 means that both units

are used.

Mem means memory in/out unit.

FP0 means floating point unit 0 (includes multiply, divide and other SIMD in-

structions).

FP1 means floating point unit 1 (adder).

MUL means multiplier, shared between FP and integer units. DIV means divider, shared between FP and integer units.

np means not pairable: Cannot execute simultaneously with any other in-

struction.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give

a similar delay.

Reciprocal throughput: The average number of clock cycles per instruction for a series of indepen-

dent instructions of the same kind in the same thread.

Integer instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOV	r,r	1	ALU0/1	1	1/2	
MOV	r,i	1	ALU0/1	1	1/2	
MOV	r,m	1	ALU0, Mem	1-3	1	All addr. modes
MOV	m,r	1	ALU0, Mem	1	1	All addr. modes
MOV	m,i	1	ALU0, Mem		1	
MOV	r,sr	1		1	1	
MOV	m,sr	2			5	
MOV	sr,r	7			21	
MOV	sr,m	8			26	
MOVNTI	m,r	1	ALU0, Mem		2.5	
MOVSX MOVZX MOVSXD	r,r/m	1	ALU0	1	1	
CMOVcc	r,r	1	ALU0+1	2	2	
CMOVcc	r,m	1			3	
XCHG	r,r	3		6	6	
XCHG	r,m	4		6	6	Implicit lock
XLAT		3		6	6	
PUSH	r	1	np	1	1	

Atom

PUSH	i	1	np		1	
PUSH	m	2			5	
PUSH	sr	3			6	
PUSHF(D/Q)		14			12	
PUSHA(D)		9			11	Not in x64 mode
POP	r	1	nn	1	1	Not in xo4 mode
		-	np			
POP	(E/R)SP	1	np	1	1	
POP	m m	3			6	
POP	sr	7			31	
POPF(D/Q)		19			28	
POPA(D)		16			12	Not in x64 mode
LAHF		1	ALU0+1	2	2	
SAHF		1	ALU0/1	1	1/2	
SALC		2		7	5	Not in x64 mode
57.25		_		•		4 clock latency
LEA	r,m	1	AGU1	1-4	1	on input register
BSWAP	r	1	ALU0	1	1	on input regioter
LDS LES LFS LGS LSS	m '	10	ALOU	30	30	
PREFETCHNTA		10	Mem	30		
	m 		1		1	
PREFETCHT0/1/2	m m	1	Mem		1	
LFENCE		1			1/2	
MFENCE		1			1	
SFENCE		1			1	
Arithmetic instructions						
ADD SUB	r,r/i	1	ALU0/1	1	1/2	
ADD SUB	r,m	1	ALU0/1, Mer	n	1	
ADD SUB	m,r/i	1		2	1	
ADC SBB	r,r/i	1		2	2	
ADC SBB	r,m	1		2	2	
ADC SBB	m,r/i	1		2	2	
CMP	r,r/i	1	ALU0/1	1	1/2	
CMP	· ·	1	ALOU/I	'	1	
	m,r/i		AL LIO/4	4		
INC DEC NEG NOT	r	1	ALU0/1	1	1/2	
INC DEC NEG NOT	m m	1		1		
AAA		13		16		Not in x64 mode
AAS		13		12		Not in x64 mode
DAA		20		20		Not in x64 mode
DAS		21		25		Not in x64 mode
AAD		4		7		Not in x64 mode
AAM		10		24		Not in x64 mode
MUL IMUL	r8	3	ALU0, Mul	7	7	
MUL IMUL	r16	4	ALU0, Mul	6	6	
MUL IMUL	r32	3	ALU0, Mul	6	6	
MUL IMUL	r64	8	ALU0, Mul	14	14	
IMUL	r16,r16	2	ALUO, Mul	6	5	
IMUL	r32,r32	1	ALU0, Mul	5	2	
IMUL	r64,r64	6	ALU0, Mul	13	11	
IMUL	r16,r16,i	2	ALU0, Mul	5	5	
IMUL	r32,r32,i	1	ALU0, Mul	5	2	
IMUL	r64,r64,i	7	ALU0, Mul	14	14	
MUL IMUL	m8	3	ALU0, Mul	6		
MUL IMUL	m16	5	ALU0, Mul	7		
•	•				•	,

Atom

MUL IMUL MUL IMUL DIV DIV DIV DIV IDIV IDIV IDIV IDIV CBW CWDE CDQE CWD CDQ CQO	m32 m64 r/m8 r/m16 r/m32 r/m 64 r/m8 r/m16 r/m32 r/m64	4 8 9 12 12 38 26 29 29 60 2 1 1 2	ALUO, Mul ALUO, Div ALUO, Div ALUO, Div ALUO, Div ALUO, Div ALUO, Div ALUO, Div ALUO ALUO ALUO ALUO ALUO ALUO ALUO	7 14 22 33 49 183 38 45 61 207 5 1 1 5	22 33 49 183 38 45 61 207	
Logic instructions AND OR XOR AND OR XOR AND OR XOR TEST TEST TEST SHR SHL SAR SHR SHL SAR ROR ROL ROR ROL RCR RCL SHLD SHLD SHLD SHLD SHLD SHLD SHLD SHL	r,r/i r,m m,r/i r,r/i m,r/i r,r/i m,r/i r,i/cl m,i/cl r,i/cl m,i/cl r,1 r/m,i/cl r16,r16,i r32,r32,i r64,r64,i r16,r16,cl r32,r32,cl r64,r64,cl r16,r16,cl r32,r32,cl r64,r64,cl r16,r16,cl r32,r32,cl r64,r64,cl r,r/i m,r m,i r,r/i m,r m,i r,r/m r m	1 1	ALU0/1 ALU0/1, Mer ALU0/1, Mer ALU0/1, Mer ALU0 ALU0 ALU0 ALU0 ALU0 ALU0 ALU0 ALU0	1 1	1/2 1 1/2 1 1 1 1 1 1	1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem 1-2 more if mem

Atom

CMC CLD		1 5		2	2 7	
STD		6			25	
Control transfer instruction	ns					
JMP	short/near	1	ALU1		2	
JMP	far	29			66	Not in x64 mode
JMP	r	1			4	
JMP	m(near)	2			7	
JMP	m(far)	30			78	
Conditional jump	short/near	1	ALU1		2	
J(E/R)CXZ	short	3			7	
LÒOP	short	8			8	
LOOP(N)E	short	8			8	
CALL	near	1			3	
CALL	far	37			65	Not in x64 mode
CALL	r	1			18	
CALL	m(near)	2			20	
CALL	m(far)	38			64	
RETN	()	1	np		6	
RETN	i	1	np		6	
RETF		36			80	
RETF	i	36			80	
BOUND	r,m	11			10	Not in x64 mode
INTO	,	4			6	Not in x64 mode
String instructions						
LODS		3		6		
REP LODS		5n+11		3n+50		
STOS		2		5		
REP STOS		3n+10		2n+4		
MOVS		4		6		
REP MOVS		4n+11		2n - 4n		fastest for high n
SCAS		3		6		
REP SCAS		5n+16		3n+60		
CMPS		5		7		
REP CMPS		6n+16		4n+40		
Other						
NOP (90)		1	ALU0/1		1/2	
Long NOP (0F 1F)		1	ALU0/1		1/2	
PAUSE		5		24		
ENTER	a,0	14		23		
ENTER	a,b	20+6b				
LEAVE		4			6	
CPUID		40-80		100-170		
RDTSC		16		29		
RDPMC		24		48		

Floating point x87 instructions

· · · · · · · · · · · · · · · · · · ·								
	Operands	μops	Unit	Latency	Reciprocal	Remarks		
	•			_	throughput			

Move instructions]
FLD	r	1		1	1	
FLD	m32/m64	1		3	1	
		· -				
FLD	m80	4		9	10	
FBLD	m80	52		92	92	
FST(P)	r	1		1	1	
FST(P)	m32/m64	3		7	9	
FSTP	m80	8		12	13	
FBSTP	m80	189		221	221	
FXCH	r	1		1	1	
FILD	m	1		7	6	
FIST(P)	m	3		11	9	
FISTTP	m	3		11	9	SSE3
FLDZ	111	1		11		3323
I					1	
FLD1		2			8	
FLDPI FLDL2E etc.		2			10	
FCMOVcc	r	3		9	9	
FNSTSW	AX	4			10	
FNSTSW	m16	4			10	
FLDCW	m16	2			8	
FNSTCW	m16	3			9	
FINCSTP FDECSTP		1		1	1	
FFREE(P)		1			1	
FNSAVE	m	166		321	321	
	m 					
FRSTOR	m	83		177	177	
Arithmetic instructions						
FADD(P) FSUB(R)(P)	r/m	1		5	1	
FMUL(P)	r/m	1	Mul	5	2	
FDIV(R)(P)	r/m	1	Div	71	71	
FABS	1/111	1	Div	1	1	
				· .	-	
FCHS	,	1		1	1	
FCOM(P) FUCOM	r/m	1		1	1	
FCOMPP FUCOMPP		1		1	1	
FCOMI(P) FUCOMI(P)	r	5			10	
FIADD FISUB(R)	m	3			9	
FIMUL	m	3	Mul		9	
FIDIV(R)	m	3	Div		73	
FICOM(P)	m	3			9	
FTST '		1		1	1	
FXAM		1		1	1	
FPREM		26		~110	·	
FPREM1		37		~130		
FRNDINT		19		48		
Math						
FSCALE		30		56		
FXTRACT		15		24		
FSQRT		1	Div	71		
FSIN FCOS		9		~260		
FSINCOS		112		~260		
F2XM1		25		~100		
FYL2X FYL2XP1		63		~220		

FPTAN	100	~300		
FPATAN	91	~300		
Other				
FNOP	1 1		1	
WAIT	2	5	5	
FNCLEX	4		26	
FNINIT	23	74		

Integer MMX and XMI	M instructions	Integer MMX and XMM instructions								
	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks				
Move instructions										
MOVD	r32/64,mm/x	1		4	2					
MOVD	m32/64,mm/x	1	Mem	5	1					
MOVD	mm/x,r32/64	1		3	1					
MOVD	mm/x,m32/64	1	Mem	4	1					
MOVQ	mm/x, mm/x	1	FP0/1	1	1/2					
MOVQ	mm/x,m64	1	Mem	4	1					
MOVQ	m64, mm/x	1	Mem	5	1					
MOVDQA	xmm, xmm	1	FP0/1	1	1/2					
MOVDQA	xmm, m128	1	Mem	4	1					
MOVDQA	m128, xmm	1	Mem	5	1					
MOVDQU	m128, xmm	3	Mem	6	6					
MOVDQU	xmm, m128	4	Mem	6	6					
LDDQU	xmm, m128	4	Mem	6	6					
MOVDQ2Q	mm, xmm	1		1	1					
MOVQ2DQ	xmm,mm	1		1	1					
MOVNTQ	m64,mm	1	Mem	~400	1					
MOVNTDQ	m128,xmm	1	Mem	~450	3					
PACKSSWB/DW										
PACKUSWB	mm/x, mm/x	1	FP0	1	1					
PUNPCKH/LBW/WD/DQ	mm/x, mm/x	1	FP0	1	1					
PUNPCKH/LQDQ	mm/x, mm/x	1	FP0	1	1					
PSHUFB	mm,mm	1	FP0	1	1					
PSHUFB	xmm,xmm	4		6	6					
PSHUFW	mm,mm,i	1	FP0	1	1					
PSHUFL/HW	xmm,xmm,i	1	FP0	1	1					
PSHUFD	xmm,xmm,i	1	FP0	1	1					
PALIGNR	xmm, xmm,i	1	FP0	1	1					
MASKMOVQ	mm,mm	1	Mem		2					
MASKMOVDQU	xmm,xmm	2	Mem		7					
PMOVMSKB	r32,mm/x	1		4	2					
PINSRW	mm/x,r32,i	1		3	1					
PEXTRW	r32,mm/x,i	2		5	5					
Arithmetic instructions										
PADD/SUB(U)(S)B/W/D	mm/x, mm/x	1	FP0/1	1	1/2					
PADDQ PSUBQ	mm/x, mm/x	2		5	5					
PHADD(S)W PHSUB(S)W	mm/x, mm/x	7		8	8					

PHADDD PHSUBD	mm/x, mm/x	3		6		
PCMPEQ/GTB/W/D	mm/x, mm/x	1	FP0/1	1	1/2	
PMULL/HW PMULHUW	mm,mm	1	FP0, Mul	4	1	
PMULL/HW PMULHUW	xmm,xmm	1	FP0, Mul	5	2	
PMULHRSW	mm,mm	1	FP0, Mul	4	1	
PMULHRSW	xmm,xmm	1	FP0, Mul	5	2	
PMULUDQ	mm,mm	1	FP0, Mul	4	1	
PMULUDQ	xmm,xmm	1	FP0, Mul	5	2	
PMADDWD	mm,mm	1	FP0, Mul	4	1	
PMADDWD	xmm,xmm	1	FP0, Mul	5	2	
PMADDUBSW	mm,mm	1	FP0, Mul	4	1	
PMADDUBSW	xmm,xmm	1	FP0, Mul	5	2	
PSADBW	mm,mm	1	FP0, Mul	4	1	
PSADBW	xmm,xmm	1	FP0, Mul	5	2	
PAVGB/W	mm/x,mm/x	1	FP0/1	1	1/2	
PMIN/MAXUB	mm/x,mm/x	1	FP0/1	1	1/2	
PMIN/MAXSW	mm/x,mm/x	1	FP0/1	1	1/2	
PABSB PABSW PABSD	mm/x,mm/x	1	FP0/1	1	1/2	
PSIGNB PSIGNW PSIGND						
	mm/x,mm/x	1	FP0/1	1	1/2	
Logic instructions						
PAND(N) POR PXOR	mm/x,mm/x	1	FP0/1	1	1/2	
PSLL/RL/RAW/D/Q	mm/x,mm/x	2	FP0	5	5	
PSLL/RL/RAW/D/Q	(x)xmm,i	1	FP0	1	1	
PSLL/RLDQ	xmm,i	1	FP0	1	1	
Other						
EMMS		9			9	

Floating point XMM instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVAPS/D	xmm,xmm	1	FP0/1	1	1/2	
MOVAPS/D	xmm,m128	1	Mem	4	1	
MOVAPS/D	m128,xmm	1	Mem	5	1	
MOVUPS/D	xmm,m128	4	Mem	6	6	
MOVUPS/D	m128,xmm	3	Mem	6	6	
MOVSS/D	xmm,xmm	1	FP0/1	1	1/2	
MOVSS/D	xmm,m32/64	1	Mem	4	1	
MOVSS/D	m32/64,xmm	1	Mem	5	1	
MOVHPS/D MOVLPS/D	xmm,m64	1	Mem	5	1	
MOVHPS/D	m64,xmm	1	Mem	4	1	
MOVLPS/D	m64,xmm	1	Mem	4	1	
MOVLHPS MOVHLPS	xmm,xmm	1	FP0	1	1	
MOVMSKPS/D	r32,xmm	1		4	2	
MOVNTPS/D	m128,xmm	1	Mem	~500	3	
SHUFPS	xmm,xmm,i	1	FP0	1	1	
SHUFPD	xmm,xmm,i	1	FP0	1	1	
MOVDDUP	xmm,xmm	1	FP0	1	1	
MOVSH/LDUP	xmm,xmm	1	FP0	1	1	

		-			
UNPCKH/LPS	xmm,xmm	1	FP0	1	1
UNPCKH/LPD	xmm,xmm	1	FP0	1	
Conversion	_	4		44	4.4
CVTPD2PS	xmm,xmm	4		11	11
CVTSD2SS	xmm,xmm	3		10	10
CVTPS2PD	xmm,xmm	4		7	6
CVTSS2SD	xmm,xmm	3		6	6
CVTDQ2PS	xmm,xmm	3		6	6
CVT(T) PS2DQ	xmm,xmm	3		6	6
CVTDQ2PD	xmm,xmm	3		7	6
CVT(T)PD2DQ	xmm,xmm	3		6	6
CVTPI2PS	xmm,mm	1		6	5
CVT(T)PS2PI	mm,xmm	1		4	1
CVTPI2PD	xmm,mm	3		7	6
CVT(T) PD2PI	mm,xmm	4		7	7
CVTSI2SS	xmm,r32	3		7	6
CVT(T)SS2SI	r32,xmm	3		10	8
CVTSI2SD	xmm,r32	3		8	6
CVT(T)SD2SI	r32,xmm	3		10	8
Arithmetic					
ADDSS SUBSS	xmm,xmm	1	FP1	5	1
ADDSD SUBSD	xmm,xmm	1	FP1	5	1
ADDPS SUBPS	xmm,xmm	1	FP1	5	1
ADDPD SUBPD	xmm,xmm	3	FP1	6	6
ADDSUBPS	xmm,xmm	1	FP1	5	1
ADDSUBPD	xmm,xmm	3	FP1	6	6
HADDPS HSUBPS	xmm,xmm	5	FP0+1	8	7
HADDPD HSUBPD	xmm,xmm	5	FP0+1	8	7
MULSS	xmm,xmm	1	FP0, Mul	4	1
MULSD	xmm,xmm	1	FP0, Mul	5	2
MULPS	xmm,xmm	1	FP0, Mul	5	2
MULPD	xmm,xmm	6	FP0, Mul	9	9
DIVSS	xmm,xmm	3	FP0, Div	31	31
DIVSD	xmm,xmm	3	FP0, Div	60	60
DIVPS	xmm,xmm	6	FP0, Div	64	64
DIVPD	xmm,xmm	6	FP0, Div	122	122
RCPSS	xmm,xmm	1		4	1
RCPPS	xmm,xmm	5		9	8
CMPccSS/D	xmm,xmm	1	FP0	5	1
CMPccPS/D	xmm,xmm	3	FP0	6	6
COMISS/D UCOMISS/D	xmm,xmm	4	FP0	9	9
MAXSS/D MINSS/D	xmm,xmm	1	FP0	5	1
MAXPS/D MINPS/D	xmm,xmm	3	FP0	6	6
Math					
SQRTSS	xmm,xmm	3	FP0, Div	31	31
SQRTPS	xmm,xmm	5	FP0, Div	63	63
SQRTSD	xmm,xmm	3	FP0, Div	60	60
SQRTPD	xmm,xmm	5	FP0, Div	121	121
RSQRTSS	xmm,xmm	1	FP0	4	1
RSQRTPS	xmm,xmm	5	FP0	9	8
ı	1	T.	I .	I .	T. Control of the Con

Logic					
ANDPS/D	xmm,xmm	1	FP0/1	1	1/2
ANDNPS/D	xmm,xmm	1	FP0/1	1	1/2
ORPS/D	xmm,xmm	1	FP0/1	1	1/2
XORPS/D	xmm,xmm	1	FP0/1	1	1/2
Other					
LDMXCSR	m32	4		5	6
STMXCSR	m32	4		14	15
FXSAVE	m4096	121		142	144
FXRSTOR	m4096	116		149	150

Intel Silvermont

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the

same data. Instructions with or without V name prefix behave the same un-

less otherwise noted.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm

register, mm/x = mmx or xmm register, m = memory, m32 = 32-bit memory

operand, etc.

μορs: The number of μops from the decoder or ROM. A μop that goes to multiple

units is counted as one.

Unit: Tells which execution unit is used. Instructions that use the same unit cannot

execute simultaneously.

IP0 and IP1 means integer port 0 or 1 and their associated pipelines

IP0/1 means that either integer unit can be used.

IP0+1 means that the μop is split in two, using both units.

Mem means memory execution cluster

FP0 means floating point port 0 (includes multiply, divide, convert and shuf-

tle).

FP1 means floating point port 1 (adder).

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give

a similar delay.

Reciprocal throughput: The average number of clock cycles per instruction for a series of indepen-

dent instructions of the same kind in the same thread. Delays in the decoders are included in the latency and throughput timings. Values of 4 or more are often caused by bottlenecks in the decoders and microcode ROM

rather than the execution units.

Integer instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOV	r,r	1	IP0/1	1	0.5	
MOV	r,i	1	IP0/1	1	0.5	
MOV	r,m	1	Mem	4	1	All addr. modes
MOV	m,r	1	Mem	3	1	All addr. modes
MOV	m,i	1	Mem		1	
MOVNTI	m,r	1	Mem		2	
MOVSX MOVZX MOVSXD	r16,r8	2	IP0		4	
MOVSX MOVZX MOVSXD	r16,m8	3	IP0		10	
MOVSX MOVZX MOVSXD	r32/64,r/m	1	IP0	1	1	
CMOVcc	r,r	1	IP0/1	2	1	
CMOVcc	r,m	1			1	
XCHG	r,r	3	IP0/1	8	8	
XCHG	r,m	3		24	24	Implicit lock
XLAT		4		8	8	
PUSH	r	1	IP0+1		1	
PUSH	i	1	IP0+1		1	

PUSH PUSHF(D/Q) PUSHA(D) POP POP POP POPPOP(D/Q) POPA(D) LAHF SAHF SALC LEA LEA LEA LEA LEA LEA LEA LEA LEA LEA	r (E/R)SP m r,[r+d] r,[r+r*s] r,[rip+d] r16,[m] r r16,m16 r32/64,m32/64 m,r m	3 18 10 2 6 21 17 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	IP0+1 IP0+1 IP0 IP0/1 IP1 IP0/1 IP0/1	1 2 6 1 1 2 4 1	5 29 10 1 3 6 47 14 1 1 2 0.5 4 1 2 1 1 1 1 1 8	Not in x64 mode Not in x64 mode Not in x64 mode
MFENCE SFENCE		2 1			14 7	
Arithmetic instructions		ı			1	
ADD SUB	r,r/i	1	IP0/1	1	0.5	
ADD SUB	r,m	1	IP0/1, Mem		1	
ADD SUB ADC SBB	m,r/i r,r/i	1 1	IP0/1, Mem IP0/1	6 2	1 2	
ADC SBB	r,m	1	150/1		2	
ADC SBB	m,r/i	1		6	2	
ADCX	r32,r32	1	IP0+1	2	2	
ADCX	r64,r64	1	IP0+1	6	6	
ADOX	r32,r32	1		2	2	
ADOX	r64,r64	1		6	6	
CMP	r,r/i	1	IP0/1	1	0.5	
CMP	m,r/i	1			1	
INC DEC	r	1	IP0/1	1	1	latency to flag=2
NEG NOT	r	1	IP0/1	1	0.5	
INC DEC NEG NOT	m	1		6	1	
AAA		13		12		Not in x64 mode
AAS		13		12		Not in x64 mode
DAA		20		16		Not in x64 mode
DAS		21		16		Not in x64 mode
AAD		4		5	40	Not in x64 mode
AAM	"n	11	IDO	24	16	Not in x64 mode
MUL IMUL MUL IMUL	r8 r16	3 4	IP0 IP0	5 5	5 5	
MUL IMUL	r32	3	IP0	5 5	5	
MUL IMUL	r64	3	IP0	7	7	

IMUL	r16,r16	2	IP0	4	4	
IMUL	r32,r32	1	IP0	3	1	
IMUL	r64,r64	1	IP0	5	2	
IMUL	r16,r16,i	2	IP0	4	4	
IMUL	r32,r32,i	1	IP0	3	1	
IMUL	r64,r64,i	1	IP0	5	2	
MUL IMUL	m8	3	IP0			
MUL IMUL	m16	5	IP0			
MUL IMUL	m32	4	IP0			
MUL IMUL	m64	4	IP0	14		
MULX	r,r,r	3-4	IP0	8	8	
MULX		4	IP0		8-10	
	r,r,m			0.4		
DIV	r/m8	9	IP0, FP0	24	19	
DIV	r/m16	12	IP0, FP0	25-29	19-23	
DIV	r/m32	12	IP0, FP0	25-39	19-31	
DIV	r/m 64	23	IP0, FP0	34-94	25-94	
IDIV	r/m8	26	IP0, FP0	24-35	25	
IDIV	r/m16	29	IP0, FP0	37-41	30-32	
IDIV	r/m32	29	IP0, FP0	29-46	29-38	
IDIV	r/m64	44	IP0, FP0	47-107	47-107	
	1/11104				47-107	
CBW		2	IP0	4		
CWDE		1	IP0	1		
CDQE		1	IP0	1		
CWD		2	IP0	4		
CDQ		1	IP0	1		
CQO		1	IP0	1		
POPCNT	r16,r16	2		4	4	
POPCNT	r32,r32	1		3	1	
				3		
POPCNT	r64,r64	1			1	
CRC32	r32,r8	2		4	4	
CRC32	r32,r16	1		6	6	
CRC32	r32,r32	1		3	1	
Logic instructions						
AND OR XOR	r,r/i	1	IP0/1	1	0.5	
AND OR XOR	r,m	1	IP0/1, Mem		1	
AND OR XOR	m,r/i	1	IP0/1, Mem	6	1	
TEST	r,r/i	1	IP0/1	1	0.5	
TEST	m,r/i	1	IP0/1, Mem		1	
SHR SHL SAR	r,i/cl	1	IP0	1	1	
SHR SHL SAR	m,i/cl	1	IP0	'	1	
	,			4	-	
ROR ROL	r,i/cl	1	IP0	1	1	
ROR ROL	m,i/cl	1	IP0		1	
RCR	r,1	7	IP0	9		
RCL	r,1	1	IP0	2	2	
RCR	r,i/cl	11	IP0	12		
RCR	m,i/cl	14	IP0	13		
RCL	r,i/cl	13	IP0	12		
RCL	m,i/cl	16	IP0	14		
SHLD	r16,r16,i	10	IP0	10		2 more if mem
				2		
SHLD	r32,r32,i	1	IP0			4 more if mem
SHLD	r64,r64,i	10	IP0	10		2 more if mem
SHLD	r16,r16,cl	9	IP0	10		2 more if mem

SHLD	r32,r32,cl	2	IP0	4		2 more if mem
SHLD	r64,r64,cl	9	IP0	10		2 more if mem
SHRD	r16,r16,i	8	IP0	10		3 more if mem
SHRD	r32,r32,i	2	IP0	4		4 more if mem
SHRD	r64,r64,i	8-10	IP0	10		3 more if mem
SHRD	r16,r16,cl	7	IP0	10		2 more if mem
SHRD	r32,r32,cl	2	IP0	4		2 more if mem
SHRD		2	IP0	4		2 more if mem
1	r64,r64,cl	1		1	4	2 more ii mem
BT	r,r/i	1	IP0+1	1	1	
BT	m,r	7		9		
BT	m,i	1		1 1		
BTR BTS BTC	r,r/i	1	IP0+1	1	1	
BTR BTS BTC	m,r	8			10	
BTR BTS BTC	m,i	1	IP0+1		1	
BSF BSR	r,r/m	10	IP0+1	10	10	
SETcc	r/m	1	IP0+1	2	1	
CLC STC		1	IP0/1		1	
CMC		1		1	1	
CLD		4	IP0+1	'	7	
STD		5	IP0+1		35	
310			IFUTI		33	
Control transfer instruction	ns short/near	_	IP1		0	
JMP		1	IPI		2 2	
	r	1				
JMP	m(near)	1	15.4		2	
Conditional jump	short/near	1	IP1		1-2	
J(E/R)CXZ	short	2			2-15	
LOOP	short	7			10-20	
LOOP(N)E	short	8				
CALL	near	1			2	
CALL	r	1			9	
CALL	m	3			14	
RET		1			3	
RET	i	1			3	
BOUND	r,m	10			10	Not in x64 mode
INTO	.,	4			7	Not in x64 mode
String instructions						
LODS		3		5		
REP LODS		~4n		~2n		
STOS		2		4		
0100				4		per byte, best
REP STOS		~0.12B		~0.1B		case
MOVS		5		6		
REP MOVS		~ 0.2B		~0.15B		per byte, best case
SCAS		3		5		33.53
REP SCAS		~5n		~3n		
CMPS		6		6		
REP CMPS		~6n		~3n		
INLE CIVIES		-011		311		
Synchronization instruction						
XADD	m,r	6		6		

LOCK XADD	m,r	4		10		
LOCK ADD	m,r	1		10		
CMPXCHG	m,r	8		10		
LOCK CMPXCHG	m,r	6		11		
CMPXCHG8B	m,r	13		14		
LOCK CMPXCHG8B	m,r	11		14		
CMPXCHG16B	m,r	19		24		
LOCK CMPXCHG16B	m,r	17		27		
Other						
NOP (90)		1	IP0/1		0.5	
Long NOP (0F 1F)		1	IP0/1		0.5	
PAUSE		6		24		
ENTER	a,0	15		14		
ENTER	a,b	19+6b		59+5b		
LEAVE		4			5	
CPUID		31-80		54-108		
RDTSC		13		29		
RDTSCP		15		25		
RDPMC		19		19		
RDRAND	r	15			~1472	

Floating point x87 instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions					tin ougnput	
FLD	r	1		1	0.5	
FLD	m32/m64	1		4	1	
FLD	m80	5		9	8	
FBLD	m80	59		68	68	
FST(P)	r	1		1	0.5	
FST(P)	m32/m64	1		3	2	
FSTP	m80	8		9	9	
FBSTP	m80	204		239	239	
FXCH	r	2	FP0+1	1	1	
FILD	m	1		6	2	
FIST(P)	m	6		9	9	
FISTTP	m	7		6	13	
FLDZ		1			1	
FLD1		1			7	
FLDPI FLDL2E etc.		2			7	
FCMOVcc	r	3		6	6	
FNSTSW	AX	2		~9	9	
FNSTSW	m16	4			11	
FLDCW	m16	2		~6	4	
FNSTCW	m16	4		~5	5	
FINCSTP FDECSTP		1		1	0.5	
FFREE(P)		1			0.5	
FNSAVE	m	166		240	240	
FRSTOR	m	82		174	174	
Arithmetic instructions						

FADD(P) FSUB(R)(P)	r/m	1	FP1	3	1	
FMUL(P)	r/m	1	FP0	5	2	
FDIV(R)(P)	r/m	1	FP0	39	37	
FABS	1/111	1	110	1	1	
FCHS		1		1	1	
FCOM(P) FUCOM	r/m	1		5	1	
FCOMPP FUCOMPP	1/111	1		5	1	
FCOMI(P) FUCOMI(P)	r	1		5	1	
FIADD FISUB(R)	m '	3		0	5	
FIMUL	m	3			6	
FIDIV(R)	m	3			39	
FICOM(P)	m	3			5	
FTST		1		6	1	
FXAM		1		7	1	
FPREM		27		32-57	32-57	
FPREM1		27		32-57	32-57	
FRNDINT		18		26	26	
Math						
FSCALE		27			66	
FXTRACT		15		20	20	
FSQRT		1		13-40	13-40	
FSIN FCOS		18		40-170	40-170	
FSINCOS		110		40-170		
F2XM1		9		39-90		
FYL2X		34		80-140		
FYL2XP1		61		154		
FPTAN		101		45-200		
FPATAN		63		85-190		
Othor						
Other FNOP	-	1			0.5	
WAIT		2			4	
FNCLEX		4			24	
FNINIT		19			65	
FININII		19			บอ	

Integer MMX and XMM instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVD MOVQ	r32/64,mm/x	1		4	1	
MOVD MOVQ	m,mm/x	1	Mem	3	1	
MOVD MOVQ	mm/x,r32/64	1		3	1	
MOVD MOVQ	mm/x,m	1	Mem	4	1	
MOVQ	mm/x, mm/x	1	FP0/1	1	0.5	
MOVDQA	X, X	1	FP0/1	1	0.5	
MOVDQA MOVDQU	x, m128	1	Mem	4	1	
MOVDQA MOVDQU	m128, x	1	Mem	3	1	
LDDQU	x, m128	1	Mem	4	1	
MOVDQ2Q	mm, x	1		1	1	

MOVOODO	l	4	l I			1
MOVQ2DQ	x,mm	1	N.4	1	1	
MOVNTQ	m64,mm	1	Mem	~370	1	
MOVNTDQ	m128,x	1	Mem	~370	1	
MOVNTDQA	x, m128	1		4	1	
PACKSSWB/DW		4	ED0	4	4	
PACKUSWB	mm/x, mm/x	1	FP0	1	1	
PACKUSDW	x,x	1	FP0	1	1	
PUNPCKH/LBW/WD/DQ	mm/x, mm/x	1	FP0	1	1	
PUNPCKH/LQDQ	mm/x, mm/x	1	FP0	1	1	
PMOVSX/ZX BW BD BQ DW	V V	1		1	1	
DQ PMOVSX/ZX BW BD BQ DW	x,x	ı		ı	ļ	
DQ	x,m	1		1	1	
PSHUFB	mm,mm	1	FP0	1	1	
PSHUFB	x,x	4	FP0	5	5	
PSHUFW	mm,mm,i	1	FP0	1	1	
PSHUFL/HW	x,x,i	1	FP0	1	1	
PSHUFD	x,x,i	1	FP0	1	1	
PALIGNR		1	FP0	1	1	
PBLENDVB	X,X,İ	2	FP0	4	-	
	x,x,xmm0		FP0	4	4	
PBLENDVB	x,m,xmm0	3		4	5	
PBLENDW	x,x/m,i	1	FP0	1	1	
MASKMOVQ	mm,mm	1	Mem	~370	1	
MASKMOVDQU	x,x	3	Mem	~370	5	
PMOVMSKB	r32,mm/x	1		4	1	
PINSRW	mm/x,r32,i	1		3	1	
PINSRB/D/Q	x,r32,i	1		3	1	
PINSRB/D/Q	x,m8,i	1			1	
PEXTRW	r32,mm/x,i	2		5	4	
PEXTRB/W	r32,x,i	2		5	4	
PEXTRQ	r64,x,i	2		7	7	
PEXTRB/W	m8/16,x,i	5			6	
PEXTRD	m32,x,i	4			5	
PEXTRQ	m64,x,i	4			8	
Arithmetic instructions						
PADD/SUB(U)(S)B/W/D	mm/x, mm/x	1	FP0/1	1	0.5	
PADDQ PSUBQ	mm/x, mm/x	2		4	4	
PADDQ PSUBQ	mm/x, m	3			5	
PHADD(S)W PHSUB(S)W	mm, mm	5		6	6	
PHADD(S)W PHSUB(S)W	x, x/m	7-8		9	9	
PHADDD PHSUBD	mm/x, mm/x	3-4		5-6	5-6	
PCMPEQ/GTB/W/D	mm/x,mm/x	1	FP0/1	1	0.5	
PCMPEQQ	x, x	2		4	4	+1 if mem
PCMPGTQ	x, x	1	FP0	5	2	
PMULL/HW PMULHUW	mm,mm	1	FP0	4	1	
PMULL/HW PMULHUW	x, x	1	FP0	5	2	
PMULHRSW	mm,mm	1	FP0	4	1	
PMULHRSW	x, x	1	FP0	5	2	
PMULLD	x, x	7	FP0	11	_ 11	+1 if mem
PMULDQ	x, x	1	FP0	5	2	
PMULUDQ	mm,mm	1	FP0	4	1	
PMULUDQ	x, x	1	FP0	5	2	
PMADDWD	mm,mm	1	FP0	4	1	
1 (55115	'''''	'		•	'	l

PMADDWD	x, x	1	FP0	5	2	
PMADDUBSW	mm,mm	1	FP0	4	_ 1	
PMADDUBSW	x, x	1	FP0	5	2	
PSADBW	mm,mm	1	FP0	4	1	
PSADBW	x, x	1	FP0	5	2	
MPSADBW	x,x,i	3		7	6	
MPSADBW	x,m,i	4			6	
PAVGB/W	mm/x,mm/x	1	FP0/1	1	0.5	
PMIN/MAXUB	mm/x,mm/x	1	FP0/1	1	0.5	
PMIN/MAXSW	mm/x,mm/x	1	FP0/1	1	0.5	
PMIN/PMAX	·					
SB/SW/SD						
UB/UW/UD	x,x	1		1	1	
PHMINPOSUW	x,x	1	FP0	5	2	
PABSB PABSW PABSD	mm/x,mm/x	1	FP0/1	1	0.5-1	
PSIGNB PSIGNW PSIGND						
	mm/x,mm/x	1	FP0/1	1	0.5-1	
Logic instructions						
PAND(N) POR PXOR	mm/x,mm/x	1	FP0/1	1	0.5	
PTEST	x,x	1	110/1		1	
PSLL/RL/RAW/D/Q	mm/x,mm/x	2	FP0	2	2	
PSLL/RL/RAW/D/Q	mm/x,i	1	FP0	1	1	
PSLL/RLDQ	X,i	1	FP0	1 1	1	
i dee, i dee	Χ,:					
String instructions						
PCMPESTRI	x,x,i	9	FP0	21	21	+1 if mem
PCMPESTRM	x,x,i	8	FP0	17	17	+1 if mem
PCMPISTRI	x,x,i	6	FP0	17	17	+1 if mem
PCMPISTRM	x,x,i	5	FP0	13	13	+1 if mem
Encryption instructions						
PCLMULQDQ	x,x,i	8	FP0	10	10	+1 if mem
Other						
EMMS		9			10	
	l .	_	I.	1	-	

Floating point XMM instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVAPS/D	X, X	1	FP0/1	1	0.5	
MOVAPS/D	x,m128	1	Mem	4	1	
MOVAPS/D	m128,x	1	Mem	3	1	
MOVUPS/D	x,m128	1	Mem	4	1	
MOVUPS/D	m128,x	1	Mem	3	1	
MOVSS/D	x, x	1	FP0/1	1	0.5	
MOVSS/D	x,m32/64	1	Mem	4	1	
MOVSS/D	m32/64,x	1	Mem	3	1	
MOVHPS/D MOVLPS/D	x,m64	1	Mem	4	1	
MOVHPS/D MOVLPS/D	m64,x	1	Mem	3	1	
MOVLHPS MOVHLPS	X,X	1	FP0	1	1	
BLENDPS/PD	x,x/m,i	1		1	1	

BLENDVPS/PD	x,x,xmm0	2	FP0+1	4	4	
BLENDVPS/PD	x,m,xmm0	3	FP0+1	5	5	
INSERTPS	x,x,i	1		1	1	
INSERTPS	x,m32,i	3		5	5	
EXTRACTPS	r32,x,i	2			4	
EXTRACTPS	m32,x,i	4		4	5	
			ED0		3	
MOVMSKPS/D	r32,x	1	FP0	4	1	
MOVNTPS/D	m128,x	1	Mem	~370	1	
SHUFPS	x,x,i	1	FP0	1	1	
SHUFPD	x,x,i	1	FP0	1	1	
MOVDDUP	x, x	1	FP0	1	1	
MOVSH/LDUP	x, x	1	FP0	1	1	
UNPCKH/LPS	x, x	1	FP0	1	1	
UNPCKH/LPD	x, x	1	FP0	1	1	
Conversion			FD 2	_		
CVTPD2PS	X, X	1	FP0	5	2	
CVTSD2SS	X, X	1	FP0	4	2	
CVTPS2PD	x, x	1	FP0	5	2	
CVTSS2SD	x, x	1	FP0	4	2	
CVTDQ2PS	x, x	1	FP0	5	2	
CVT(T) PS2DQ	x, x	1	FP0	5	2	
CVTDQ2PD	x, x	1	FP0	5	2	
CVT(T)PD2DQ	x, x	1	FP0	5	2	
CVTPI2PS	x,mm	1	FP0	4	2	
CVT(T)PS2PI		1	FP0	4	2	
CVT(1)F32F1 CVTPI2PD	mm,x	1		5	2	
	x,mm		FP0			
CVT(T) PD2PI	mm,x	1	FP0	5	2	
CVTSI2SS	x,r32	1	FP0	5	2	
CVT(T)SS2SI	r32,x	1	FP0	5	1	
CVTSI2SD	xm,r32	1	FP0	5	2	
CVT(T)SD2SI	r32,x	3	FP0	5	1	
Arithmetic						
ADDSS SUBSS	· · ·	1	FP1	3	1	
ADDSO SOBSO	X, X	1	FP1	3	1	
ADDSD SUBSD	X, X					
1	X, X	1	FP1	3	1	
ADDPD SUBPD	x, x	1	FP1	4	2	
ADDSUBPS	x, x	1	FP1	3	1	
ADDSUBPD	x, x	1	FP1	4	2	
HADDPS HSUBPS	X, X	4		6	6	+1 if mem
HADDPD HSUBPD	x, x	4		6	5	+1 if mem
MULSS	x, x	1	FP0	4	1	
MULSD	x, x	1	FP0	5	2	
MULPS	x, x	1	FP0	5	2	
MULPD	x, x	1	FP0	7	4	
DIVSS	x, x	1	FP0	19	17	
DIVSD	x, x	1	FP0	34	32	
DIVPS		6	FP0	39	39	
	X, X		FP0			
DIVPD	X, X	6		69	69	
RCPSS	X, X	1	FP0	4	1	
RCPPS	x, x	5	FP0	9	8	
CMPccSS/D PS/D	x, x	1	FP1	3	1	

COMISS/D UCOMISS/D	x, x	1	FP1		1	
MAXSS/D MINSS/D	x, x	1	FP1	3	1	
MAXPS MINPS	x, x	1	FP1	3	1	
MAXPD MINPD	x, x	1	FP1	4	2	
ROUNDSS/D	x,x,i	1	FP0	4	2	
ROUNDPS/D	x,x,i	1	FP0	5	2	
DPPS	x,x,i	9	FP0	15	12	+1 if mem
DPPD	x,x,i	5	FP0	12	8	+1 if mem
Math						
SQRTSS	x, x	1	FP0	20	18	
SQRTPS	x, x	5	FP0	40	40	
SQRTSD	x, x	1	FP0	35	33	
SQRTPD	x, x	5	FP0	70	70	
RSQRTSS	x, x	1	FP0	4	1	
RSQRTPS	X, X	5	FP0	9	8	
Logic						
ANDPS/D	x, x	1	FP0/1	1	0.5	
ANDNPS/D	x, x	1	FP0/1	1	0.5	
ORPS/D	x, x	1	FP0/1	1	0.5	
XORPS/D	X, X	1	FP0/1	1	0.5	
Other						
LDMXCSR	m32	5		10	8	
STMXCSR	m32	4		12	11	
FXSAVE	m4096	115		132	132	32 bit mode
FXSAVE	m4096	123		143	143	64 bit mode
FXRSTOR	m4096	114		118	118	32 bit mode
FXRSTOR	m4096	123		122	122	64 bit mode

Intel Goldmont

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the

same data.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm

register, v = mmx or xmm vector register, m = memory, m32 = 32-bit memory

operand, etc.

μορs: The number of μops from the decoder or ROM. A μop that goes to multiple

units is counted as one.

Unit: Tells which execution unit is used. Instructions that use the same unit cannot

execute simultaneously.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give

a similar delay.

Reciprocal throughput: The average number of clock cycles per instruction for a series of indepen-

dent instructions of the same kind in the same thread. Delays in the decoders are included in the latency and throughput timings. Values of 4 or more are often caused by bottlenecks in the decoders and microcode ROM

rather than the execution units.

Integer instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOV	r,r	1		1	0.33	
MOV	r,i	1		1	0.33	
MOV	r,m	1		3	1	All addr. modes
MOV	m,r	1		2	1	All addr. modes
MOV	m,i	1			1	
MOVNTI	m,r	2		2	2	
MOVSX MOVZX	r16,r8	2			4	
MOVSX MOVZX	r16,m8	3			4	
MOVSX MOVZX MOVSX	r32/64,r	1		1	1	
MOVSX MOVSXD	r32/64,m	1		3	1	
MOVZX	r32/64,m	1		2	1	
CMOVcc	r,r	1		2	1	
CMOVcc	r,m	1			1	
XCHG	r,r	3		4	4	
XCHG	r,m	3		13	13	Implicit lock
XLAT		4		5	5	
PUSH	r	1			1	
PUSH	i	1			1	
PUSH	m	2			1	
PUSH	sp	2			1	
PUSHF(D/Q)	•	16			26	
PUSHA(D)		10			8	Not in x64 mode
POP	r	1			1	
POP	m	6			5	

POP	sp	1			2.5	
POPF(D/Q)	-7-	22			42	
POPA(D)		17			10	Not in x64 mode
						Not in X04 mode
LAHF		1		2	2	
SAHF		1		1	1	
SALC		2		4	4	Not in x64 mode
LEA	r16,[m]	2		5	4	
LEA	r,[r+d]	1		1 1	1	
LEA	r,[r+r*1]	1		1 1	1	
LEA				2	2	
	r32,[r+r*s]	1		I I		
LEA	r64,[r+r*s]	1		1	1	
LEA	r,[r+r*s+d]	1		2	2	
LEA	r,[rip+d]	1			0.5	
BSWAP	r	1		1	1	
MOVBE	r,m	1			1	
MOVBE	m,r	1			1	
PREFETCHNTA	m	1			1	
PREFETCHT0/1/2		-			•	
I	m	1			1	
PREFETCHNTW	m	1			1	
LFENCE		3			8	
MFENCE		3			22	
SFENCE		1			2	
CLFLUSH	m	1			165	
CLFLUSHOPT	m	4			165	
021 2001101 1	•••				100	
Arithmetic instructions						
ADD SUB	r r/i	1		1 1	0.33	
	r,r/i					
ADD SUB	r,m	1			1	
ADD SUB	m,r/i	1		5	1	
ADC SBB	r,r/i	1		2	2	
ADC SBB	r,m	1			2	
ADC SBB	m,r/i	1		6	2	
CMP	r,r/i	1		1 1	0.33	
CMP	m,r/i	1			1	
INC DEC	r	1		1 1	1	latency to flag=2
NEG NOT		1				laterity to mag-2
I	r			1 1	0.33	
INC DEC NEG NOT	m	1		5	1	
AAA		14		10		Not in x64 mode
AAS		14		10		Not in x64 mode
DAA		21		13		Not in x64 mode
DAS		22		14		Not in x64 mode
AAD		4		5		Not in x64 mode
AAM		5		14	11	Not in x64 mode
MUL IMUL	r8	2		4	4	110t III XO4 IIIOGO
				l I		
MUL IMUL	r16	4		5	5	
MUL IMUL	r32	2		4	2	
MUL IMUL	r64	2		6	2	
IMUL	r16,r16	2		4	4	
IMUL	r32,r32	1		3	1	
IMUL	r64,r64	1		5	2	
IMUL	r16,r16,i	2		4	4	
IMUL	r32,r32,i	1		3	1	
IMUL	r64,r64,i	1		5	2	
IIIVIOL	104,104,1	1	I	5	4	1

MUL IMUL MUL IMUL MUL IMUL MUL IMUL DIV DIV DIV IDIV IDIV IDIV IDIV CBW CWDE CDQE	m8 m16 m32 m64 r8 r16 r32 r64 r8 r16 r32 r64	3 5 2 2 3 6 6 6 6 6 6 6 2 1 1	13-14 14-19 14-27 14-43 13-14 14-19 14-27 14-43 4 1	1 2 11-12 13-18 13-26 13-42 11-12 13-18 13-26 13-42	
CWD CDQ		2 1	4 1		
CQO		1	1		
POPCNT	r16,r16	2	4	4	
POPCNT	r32,r32	1	3	1	
POPCNT	r64,r64	1	3	1	
CRC32	r32,r8	1	3	1	
CRC32	r32,r16	1	3	2	
CRC32	r32,r32	1	3	1	
Logic instructions					
AND OR XOR	r,r/i	1	1	0.33	
AND OR XOR	r,m	1		1	
AND OR XOR	m,r/i	1	5	1	
TEST	r,r/i	1	1	0.33	
TEST	m,r/i	1		1	
SHR SHL SAR	r,i/cl	1	1	1	
SHR SHL SAR ROR ROL	m,i/cl r,i/cl	1 1	1	1	
ROR ROL	m,i/cl	1	I	1	
RCL	r,1	1	2	2	
RCL	r,i	16	14	14	2 more if mem
RCL	r,cl	16	16	16	2 more if mem
RCR	r,1	10	11	11	3 more if mem
RCR	r,i	14	12	12	2 more if mem
RCR	r,cl	14	14	14	2 more if mem
SHLD	r16,r16,i	10	15	15	4 more if mem
SHLD	r32,r32,i	2 13	2	2	6 more if mem 2 more if mem
SHLD SHLD	r64,r64,i r16,r16,cl	10	12 17	12 17	4 more if mem
SHLD	r32,r32,cl	2	4	4	6 more if mem
SHLD	r64,r64,cl	12	14	14	2 more if mem
SHRD	r16,r16,i	10	11	11	3 more if mem
SHRD	r32,r32,i	2	2	2	6 more if mem
SHRD	r64,r64,i	13	12	12	2 more if mem
SHRD	r16,r16,cl	10	13	13	2 more if mem
SHRD	r32,r32,cl	2	4	4	6 more if mem
SHRD	r64,r64,cl	12	14	14	2 more if mem

BT BT BT BT BTR BTS BTC BTR BTS BTC BTR BTS BTC BSF BSR SETcc CLC STC CMC CLD STD	r,r/i m,r m,i r,r/i m,r m,i r,r/m	1 10 1 1 10 1 11 1 1 1 3 4	1 1 12 10 2	1 12 1 1 10 2 8 1 1 1 7 35	
Control transfer instruc	tions				
JMP JMP Conditional jump J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO	short/near r m(near) short/near short short short near r m	1 1 1 2 9 9 1 1 3 1 2 9		2 2 1-2 4-18 10-23 12-24 2 8 11 2-3 2 9	Not in x64 mode Not in x64 mode
String instructions LODS REP LODS STOS		3 ~5n 2	4 ~2n 4		per byte, best
REP STOS MOVS REP MOVS SCAS REP SCAS CMPS REP CMPS		~0.13B 5 ~0.2B 3 ~6n 6 ~8n	~0.07B 5 ~0.07B 4 ~2n 4 ~3n		case per byte, best case
Synchronization instruction XADD LOCK XADD LOCK ADD CMPXCHG LOCK CMPXCHG CMPXCHG8B LOCK CMPXCHG8B CMPXCHG16B LOCK CMPXCHG16B	m,r m,r m,r m,r m,r m,r m,r m,r m,r	6 6 1 8 8 14 14 19	5 13 13 8 14 9 16 23 30		

Other					
NOP (90)		1		0.33	
Long NOP (0F 1F)		1		1	
PAUSE		3	147		
ENTER	a,0	14	10		
ENTER	a,b	17+6b	~75+3b		
LEAVE		2		3	
CPUID		37-78	69-2800		
RDTSC		19		20	
RDTSCP		22		31	
RDPMC		13		9	
RDRAND	r	17		~3100	
RDSEED	r	17		~3100	

Floating point x87 instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions					tilloughput	
FLD	r	1		1	0.5	
FLD	m32/m64	1		3	1	
FLD	m80	1		3	1	
FBLD	m80	54		56	50	
FST(P)		1		1	0.5	
` ,	r m32/m64	1		3	2	
FST(P)		1		3	2	
FSTP	m80					
FBSTP	m80	195		190	190	
FXCH	r	2		1	1	
FILD	m	1		7	1	
FIST(P)	m	8		6	10	
FISTTP	m	8		6	13	
FLDZ		1			1	
FLD1		2			6	
FLDPI FLDL2E etc.		2			3	
FCMOVcc	r	4		5	5	
FNSTSW	AX	2			10	
FNSTSW	m16	3			11	
FLDCW	m16	4			15	
FNSTCW	m16	3			4	
FINCSTP FDECSTP		1		1	0.5	
FFREE(P)		1			0.5	
FNSAVE	m	151		173	173	
FRSTOR	m	85		155	155	
Arithmetic instructions						
FADD(P) FSUB(R)(P)	r	1		3	1	
FADD(P) FSUB(R)(P)	m	2			2	
FMUL(P)	r	1		5	2	
FMUL(P)	m	2			2	
FDIV(R)(P)	r	1		39	38	
FDIV(R)(P)	m	2			38	
FABS		1		1	1	

FCHS		1	1	1	
FCOM(P) FUCOM	r/m	1		1	
FCOMPP FUCOMPP		1		1	
FCOMI(P) FUCOMI(P)	r	1		1	
FIADD FISUB(R)	m	3		5	
FIMUL	m	3		5	
FIDIV(R)	m	4		38	
FICOM(P)	m	3		4	
FTST		1		1	
FXAM		1		1	
FPREM		29	37-42	37-42	
FPREM1		29	37-42	37-42	
FRNDINT		19	41	41	
Math					
FSCALE		30	32	32	
FXTRACT		16	22	22	
FSQRT		1	10-40	40-40	
FSIN FCOS		17-100	45-150	45-150	
FSINCOS		17-110	48-135	48-135	
F2XM1		9-27	40-90	40-90	
FYL2X		34-61	88-130	88-130	
FYL2XP1		61	140	140	
FPTAN		16-100	45-180	45-180	
FPATAN		33-65	85-190	85-190	
Other					
FNOP		1		0.5	
WAIT		2		6	
FNCLEX		4		24	
FNINIT		14		49	

Integer MMX and XMM instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVD MOVQ	r32/64,v	1		4	1 1	
MOVD MOVQ	m,v	1		3	1 1	
MOVD MOVQ	v,r32/64	1		4	1 1	
MOVD MOVQ	v,m	1		3	1	
MOVQ	V,V	1		1	0.5	
MOVDQA	x, x	1		1	0.5	
MOVDQA MOVDQU	x, m128	1		3	1 1	
MOVDQA MOVDQU	m128, x	1		3	1 1	
LDDQU	x, m128	1		3	1 1	
MOVDQ2Q	mm, x	1		1	0.5	
MOVQ2DQ	x,mm	1		1	0.5	
MOVNTQ	m64,mm	1		3	1	
MOVNTDQ	m128,x	1		3	1	
MOVNTDQA	x, m128	1		3	1	

			Coldinoni			
PACKSSWB/DW						
PACKUSWB	V,V	1		1	0.5	
PACKUSDW	x,x	1		1	0.5	
PUNPCKH/LBW/WD/DQ	V,V	1		1	0.5	
PUNPCKH/LQDQ	V,V	1		1	0.5	
PMOVSX/ZX BW BD BQ						
DW DQ	X,X	1		1	0.5	
PMOVSX/ZX BW BD BQ					_	
DW DQ	x,m	1		1	1	
PSHUFB	mm,mm	1		1	0.5	
PSHUFB	X,X	1		1	1	
PSHUFW	mm,mm,i	1		1	0.5	
PSHUFL/HW	x,x,i	1		1	0.5	
PSHUFD	x,x,i	1		1	0.5	
PALIGNR	x,x,i	1		1	0.5	
PBLENDVB	x,x,xmm0	2		4	4	
PBLENDVB	x,m,xmm0	3			4	
PBLENDW	x,x/m,i	1		1	0.5	
MASKMOVQ	mm,mm	1		~350	1	
MASKMOVDQU	x,x	3		~360	1	
PMOVMSKB	r32,v	1		4	1	
PINSRW	v,r32,i	1		4	1	
PINSRB/D/Q	x,r32,i	1		4	1	
PINSRB/D/Q	x,m8,i	1			1	
PEXTRB/W/D/Q	r,v,i	1		4	1	
PEXTRB/W/D/Q	m,v,i	4			4	
	, • ,.				•	
Arithmetic instructions						
PADD/SUB(U)(S)B/W/D	V,V	1		1	0.5	
PADDQ PSUBQ	V,V	1		2	1	
PHADD(S)W PHSUB(S)W	mm, mm	5		7	7	
PHADD(S)W PHSUB(S)W	x, x/m	7		6	6	+1 if mem
PHADDD PHSUBD	V,V	3		4	4	+1 if mem
PCMPEQ/GTB/W/D	V,V	1		1	0.5	
PCMPEQQ	ν, ν Χ, Χ	1		2	1	
PCMPGTQ	x, x x, x	1		2	1	
PMULL/HW PMULHUW	v, v	1		4	1	
PMULHRSW		1		4	1	
PMULLD	V,V	1		5	2	
PMULDQ	X, X	1		4	1	
	X, X	-			•	
PMULUDQ	V,V	1		4	1	
PMADDWD	V,V	1		4	1	
PMADDUBSW	V,V	1		4	1	
PSADBW	V,V	1		4	1	4 .6
		3		5	4	+1 if mem
MPSADBW	x,x,i					
PAVGB/W	V,V	1		1	0.5	
PAVGB/W PMIN/MAXUB				1 1	0.5 0.5	
PAVGB/W PMIN/MAXUB PMIN/PMAX	V,V	1		-		
PAVGB/W PMIN/MAXUB PMIN/PMAX SB/SW/SD	V,V V,V	1 1		1	0.5	
PAVGB/W PMIN/MAXUB PMIN/PMAX SB/SW/SD UB/UW/UD	v,v v,v	1 1		1	0.5 0.5	
PAVGB/W PMIN/MAXUB PMIN/PMAX SB/SW/SD UB/UW/UD PHMINPOSUW	v,v v,v x,x x,x	1 1 1 1		1 1 5	0.5 0.5 2	
PAVGB/W PMIN/MAXUB PMIN/PMAX SB/SW/SD UB/UW/UD PHMINPOSUW PABSB PABSW PABSD	v,v v,v	1 1		1	0.5 0.5	
PAVGB/W PMIN/MAXUB PMIN/PMAX SB/SW/SD UB/UW/UD PHMINPOSUW	v,v v,v x,x x,x	1 1 1 1		1 1 5	0.5 0.5 2	

Logic instructions					
PAND(N) POR PXOR	V,V	1	1	0.5	
PTEST	x,x	1	1	1	
PSLL/RL/RAW/D/Q	V,V	1	2	1	
PSLL/RL/RAW/D/Q	v,i	1	1	0.5	
PSLL/RLDQ	x,i	1	1	0.5	
String instructions					
PCMPESTRI	x,x,i	10	13	13	+1 if mem
PCMPESTRM	x,x,i	9	14	14	+1 if mem
PCMPISTRI	x,x,i	6	8	8	+1 if mem
PCMPISTRM	x,x,i	5	12	12	+1 if mem
Encryption instructions					
PCLMULQDQ	x,x,i	3	6	4	
AESDEC	x,x	1	6	2	+1 if mem
AESDECLAST	x,x	1	6	2	+1 if mem
AESENC	x,x	1	6	2	+1 if mem
AESENCLAST	x,x	1	6	2	+1 if mem
AESIMC	x,x	1	5	2	
AESKEYGENASSIST	x,x,i	1	5	2	
SHA1RNDS4	x,x,i	1	5	2	
SHA1NEXTE	x,x	1	3	1	
SHA1MSG1	x,x	1	3	1	
SHA1MSG2	x,x	1	3	1	
SHA256RNDS2	x,x	3	8	4	
SHA256MSG1	x,x	1	3	1	
SHA256MSG2	x,x	1	3	1	
Other					
EMMS		13		23	

Floating point XMM instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVAPS/D	x, x	1		0-1	0.5	
MOVAPS/D	x,m128	1		3	1	
MOVAPS/D	m128,x	1		3	1	
MOVUPS/D	x,m128	1		3	1	
MOVUPS/D	m128,x	1		3	1	
MOVSS/D	X, X	1		1	0.5	
MOVSS/D	x,m32/64	1		3	1	
MOVSS/D	m32/64,x	1		3	1	
MOVHPS/D MOVLPS/D	x,m64	1		4	1	
MOVHPS/D MOVLPS/D	m64,x	1		3	1	
MOVLHPS MOVHLPS	x,x	1		1	0.5	
BLENDPS/PD	x,x/m,i	1		1	0.5	
BLENDVPS/PD	x,x,xmm0	2		4	4	+1 if mem
INSERTPS	x,x,i	1		1	0.5	
INSERTPS	x,m32,i	3		4	4	

EXTRACTPS	r32,x,i	1	4	1	
EXTRACTPS	m32,x,i	1	4	2	
MOVMSKPS/D	r32,x	1	4	1	
MOVNTPS/D	m128,x	1	3	1	
SHUFPS	x,x,i	1	1	0.5	
SHUFPD	x,x,i	1	1	0.5	
MOVDDUP	x, x	1	1	0.5	
MOVSH/LDUP	x, x	1	1	0.5	
UNPCKH/LPS/PD	x, x	1	1	0.5	
	,		-		
Conversion					
CVTPD2PS	x, x	1	4	1	
CVTPS2PD	x, x	1	4	1	
CVTSD2SS	x, x	1	4	2	
CVTSS2SD	x, x	1	4	2	
CVTDQ2PS	x, x	1	4	1	
CVT(T) PS2DQ	X, X X, X	1	4	1	
CVT(1) P 32DQ CVTDQ2PD			4	1	
CVTDQ2PD CVT(T)PD2DQ	X, X	1	4	1	
CVT(1)PD2DQ CVTPI2PS	X, X		4	2	
	x,mm	1			
CVT(T)PS2PI	mm,x	1	4	1	
CVTPI2PD	x,mm	1	4	1	
CVT(T) PD2PI	mm,x	1	4	1	
CVTSI2SS	x,r32	1	6	2	
CVT(T)SS2SI	r32,x	1	5	1	
CVTSI2SD	xm,r32	1	6	2	
CVT(T)SD2SI	r32,x	1	5	1	
A with we atio					
Arithmetic		4	2	_	
ADDSS/SD/PS/PD	x, x	1	3	1	
SUBSS/SD/PS/PD	x, x	1	3	1	
ADDSUBPS/PD	x, x	1	3	1	. 4 '6
HADDPS HSUBPS	x, x	4	6	6	+1 if mem
HADDPD HSUBPD	x, x	4	5	5	+1 if mem
MULSS/SD/PS/PD	X, X	1	4	1	
DIVSS	x, x	1	19	18	
DIVSD	x, x	1	34	33	
DIVPS	x, x	1	36	35	
DIVPD	x, x	1	66	65	
RCPSS	x, x	1	4	1	
RCPPS	x, x	5	9	6	+1 if mem
CMPccSS/SD/PS/PD	x, x	1	3	1	
(U)COMISS/SD	X, X	1	4	1	
MAXSS/SD/PS/PD	x, x	1	3	1	
MINSS/SD/PS/PD	x, x	1	3	1	
ROUNDSS/SD/PS/PD	x,x,i	1	4	1	
DPPS	x,x,i	9	14	10	+1 if mem
DPPD	x,x,i	10	8	8	+1 if mem
Math					
Math		4	20	10	
SQRTSS	x, x	1	20	19	
SQRTPS	X, X	1	38	37	
SQRTSD	x, x	1	35	34	

SQRTPD RSQRTSS RSQRTPS	x, x x, x x, x	1 1 5		68 4 9	67 1 6	
Logic						
ANDPS/D	x, x	1		1	0.5	
ANDNPS/D	X, X	1	FP0/1	1	0.5	
ORPS/D	x, x	1		1	0.5	
XORPS/D	x, x	1		1	0.5	
Other						
LDMXCSR	m32	6		18	18	
STMXCSR	m32	3		12	12	
FXSAVE	m4096	202		420	420	32 bit mode
FASAVE	1114090	202		130	130	32 bit mode
FXSAVE	m4096	123		143	143	64 bit mode
FXSAVE	m4096	123		143	143	64 bit mode
FXSAVE FXRSTOR	m4096 m4096	123 160		143 218	143 218	64 bit mode 32 bit mode

Intel Goldmont Plus

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the

same data.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm

register, v = mmx or xmm vector register, m = memory, m32 = 32-bit memory

operand, etc.

μορs: The number of μops from the decoder or ROM. A μop that goes to multiple

units is counted as one.

Unit: Tells which execution unit is used. Instructions that use the same unit cannot

execute simultaneously.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give

a similar delay.

Reciprocal throughput: The average number of clock cycles per instruction for a series of indepen-

dent instructions of the same kind in the same thread. Delays in the decoders are included in the latency and throughput timings. Values of 4 or more are often caused by bottlenecks in the decoders and microcode ROM

rather than the execution units.

Integer instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOV	r,r	1		1	0.33	
MOV	r,i	1		1	0.33	
MOV	r,m	1		3	1	All addr. modes
MOV	m,r	1		2	1	All addr. modes
MOV	m,i	1			1	
MOVNTI	m,r	2		2	2	
MOVSX MOVZX	r16,r8	2			4	
MOVSX MOVZX	r16,m8	3			4	
MOVSX MOVZX MOVSX	r32/64,r	1		1	1	
MOVSX MOVSXD	r32/64,m	1		3	1	
MOVZX	r32/64,m	1		2	1	
CMOVcc	r,r	1		2	1	
CMOVcc	r,m	1			1	
XCHG	r,r	3		4	4	
XCHG	r,m	3		13	13	Implicit lock
XLAT		4		5	5	
PUSH	r	1			1	
PUSH	i	1			1	
PUSH	m	2			1	
PUSH	sp	2			1	
PUSHF(D/Q)		11			16	
PUSHA(D)		10			8	Not in x64 mode
POP	r	1			1	
POP	m	7			6	

POP	sp	1			2	
POPF(D/Q)		23			32	
POPA(D)		17			10	Not in x64 mode
LAHF		1		1	1	
SAHF		1		2	1	
SALC		2		4	4	Not in x64 mode
LEA	r16,[m]	2		5	4	
LEA	r,[r+d]	1		1	1	
LEA	r,[r+r*1]	1		1	1	
LEA	r32,[r+r*s]	1		2	2	
LEA	r64,[r+r*s]	1		1	1	
LEA	r,[r+r*s+d]	1		2	2	
LEA	r,[rip+d]	1			0.5	
BSWAP	r	1		1	1	
MOVBE	r,m	1			1	
MOVBE	m,r	1			1	
PREFETCHNTA	m	1			1	
PREFETCHT0/1/2	m	1			1	
PREFETCHNTW	m	1			1	
LFENCE		2			6	
MFENCE		2			18	
SFENCE		1			2	
CLFLUSH	m	1			165	
CLFLUSHOPT	m	4			165	
Arithmetic instructions		_				
ADD SUB	r,r/i	1		1	0.33	
ADD SUB	r,m	1			1	
ADD SUB	m,r/i	1		5	1	
ADC SBB	r,r/i	1		2	2	
ADC SBB	r,m	1		_	2	
ADC SBB	m,r/i	1		6	2	
CMP	r,r/i	1		1	0.33	
CMP	m,r/i	1		_	1	
INC DEC	r	1		1	1	latency to flag=2
NEG NOT	r	1		1 -	0.33	
INC DEC NEG NOT	m	1		5	1	
AAA		14		10		Not in x64 mode
AAS		14		10		Not in x64 mode
DAA		21		13		Not in x64 mode
DAS		22		14		Not in x64 mode
AAD		4		5		Not in x64 mode
AAM	•	5		14	11	Not in x64 mode
MUL IMUL	r8	2		4	4	
MUL IMUL	r16	4		5	5	
MUL IMUL	r32	2		4	2	
MUL IMUL	r64	2		6	2	
IMUL	r16,r16	2		4	4	
IMUL	r32,r32	1		3	1	
IMUL	r64,r64	1		5	2	
IMUL	r16,r16,i	2		4	4	
IMUL	r32,r32,i	1		3	1	
IMUL	r64,r64,i	1	I	5	2	

MUL IMUL MUL IMUL MUL IMUL MUL IMUL DIV DIV DIV IDIV IDIV IDIV IDIV CBW CWDE CDQE	m8 m16 m32 m64 r8 r16 r32 r64 r8 r16 r32 r64	3 5 2 2 3 6 6 6 6 6 6 6 2 1 1	13-14 14-19 14-27 14-43 13-14 14-19 14-27 14-43 4 1	1 2 11-12 13-18 13-26 13-42 11-12 13-18 13-26 13-42	
CWD CDQ		2 1	4 1		
CQO		1	1		
POPCNT	r16,r16	2	4	4	
POPCNT	r32,r32	1	3	1	
POPCNT	r64,r64	1	3	1	
CRC32	r32,r8	1	3	1	
CRC32	r32,r16	1	3	2	
CRC32	r32,r32	1	3	1	
Logic instructions					
AND OR XOR	r,r/i	1	1	0.33	
AND OR XOR	r,m	1		1	
AND OR XOR	m,r/i	1	5	1	
TEST	r,r/i	1	1	0.33	
TEST	m,r/i	1		1	
SHR SHL SAR	r,i/cl	1	1	1	
SHR SHL SAR ROR ROL	m,i/cl	1	4	1	
ROR ROL	r,i/cl m,i/cl	1 1	1	1 1	
RCL	r,1	1	2	2	
RCL	r,i	16	14	14	2 more if mem
RCL	r,cl	16	16	16	2 more if mem
RCR	r,1	10	11	11	3 more if mem
RCR	r,i	14	12	12	2 more if mem
RCR	r,cl	14	14	14	2 more if mem
SHLD	r16,r16,i	10	15	15	4 more if mem
SHLD	r32,r32,i	2	2	2	6 more if mem
SHLD	r64,r64,i	12	11	11	3 more if mem
SHLD	r16,r16,cl	10	16	16	4 more if mem
SHLD SHLD	r32,r32,cl r64,r64,cl	2 12	4 13	4 13	6 more if mem 2 more if mem
SHRD	r16,r16,i	10	10	10	3 more if mem
SHRD	r32,r32,i	2	2	2	6 more if mem
SHRD	r64,r64,i	12	11	11	3 more if mem
SHRD	r16,r16,cl	10	12	12	2 more if mem
SHRD	r32,r32,cl	2	4	4	6 more if mem
SHRD	r64,r64,cl	12	13	13	2 more if mem

BT BT BT BT BTR BTS BTC BTR BTS BTC BTR BTS BTC BSF BSR SETcc CLC STC CMC CLD STD	r,r/i m,r m,i r,r/i m,r m,i r,r/m	1 8 1 9 1 11 1 1 3 4	1 1 7 9 2	1 9 1 1 7 2 8 1 1 1 7 35	
Control transfer instruc					
JMP JMP Conditional jump J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET	short/near r m(near) short/near short short short near r	1 1 1 2 9 9 1 1 3		1-6 1 1-6 4-18 10-23 12-24 2 8 11 2-3	when predicted when predicted
RET BOUND INTO	i r,m	2 9 4		2 9 7	Not in x64 mode Not in x64 mode
String instructions LODS REP LODS STOS REP STOS MOVS		3 ~5n 2 ~0.13B 5	4 ~2n 4 ~0.07B 5		per byte, best case
REP MOVS SCAS REP SCAS CMPS REP CMPS		~ 0.2B 3 ~6n 6 ~8n	~0.07B 4 ~2n 4 ~3n		per byte, best case
Synchronization instruc	tions				
XADD LOCK XADD LOCK ADD CMPXCHG LOCK CMPXCHG CMPXCHG8B LOCK CMPXCHG8B CMPXCHG16B LOCK CMPXCHG16B	m,r m,r m,r m,r m,r m,r m,r m,r	6 6 1 8 8 14 14 19	5 13 13 6 14 9 15 24 31		

Other					
NOP (90)		1		0.33	
Long NOP (0F 1F)		1		0.33	
PAUSE		3	150		
ENTER	a,0	14	10		
ENTER	a,b	17+6b	~75+3b		
LEAVE		2		3	
CPUID		63-104	80-200	80-200	
RDTSC		21		21	
RDTSCP		24		33	
RDPMC		13		9	
RDRAND	r	16		~2300	
RDSEED	r	16		~2300	

Floating point x87 instructions

	Operands	μops	Unit	Latency		Remarks
Move instructions					throughput	
FLD		1		1	0.5	
FLD	r m32/m64	1				
1				3	1	
FLD	m80	1		3	1 1	
FBLD	m80	54		56	50	
FST(P)	r	1		1	0.5	
FST(P)	m32/m64	1		3	2	
FSTP	m80	1		3	2	
FBSTP	m80	195		190	190	
FXCH	r	2		1	1	
FILD	m	1		7	1	
FIST(P)	m	8		6	10	
FISTTP	m	8		6	13	
FLDZ		1			1	
FLD1		2			6	
FLDPI FLDL2E etc.		2			3	
FCMOVcc	r	4		5	5	
FNSTSW	AX	2			10	
FNSTSW	m16	3			11	
FLDCW	m16	4			15	
FNSTCW	m16	3			4	
FINCSTP FDECSTP		1		1	0.5	
FFREE(P)		1			0.5	
FNSAVE	m	146		183	183	
FRSTOR	m	78		162	162	
Arithmetic instructions						
FADD(P) FSUB(R)(P)	r	1		3	1	
FADD(P) FSUB(R)(P)	m	2			2	
FMUL(P)	r	1		5	2	
FMUL(P)	m	2			2	
FDIV(R)(P)	r	1		15	11	
FDIV(R)(P)	m	2			11	
FABS		1		1	1	

FCHS		1	1	1	
FCOM(P) FUCOM	r/m	1	'	1	
FCOMPP FUCOMPP	1/111	1		1	
FCOMI(P) FUCOMI(P)	r	1		1	
FIADD FISUB(R)	m	3		5	
FIMUL	m	3		5	
FIDIV(R)	m	4		11	
FICOM(P)	m	3		4	
FTST		1		1	
FXAM		1		1	
FPREM		29	35	35	
FPREM1		29	35	35	
FRNDINT		19	41	41	
Math					
FSCALE		30	32	32	
FXTRACT		16	22	22	
FSQRT		1	12-20	6-14	
FSIN FCOS		17-100	45-150	45-150	
FSINCOS		16-110	48-135	48-135	
F2XM1		9-27	40-90	40-90	
FYL2X		34-61	88-130	88-130	
FYL2XP1		61	140	140	
FPTAN		16-100	45-180	45-180	
FPATAN		33-65	85-190	85-190	
Other					
FNOP		1		0.5	
WAIT		2		6	
FNCLEX		4		21	
FNINIT		12		50	

Integer MMX and XMM instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVD MOVQ	r32/64,v	1		4	1 1	
MOVD MOVQ	m,v	1		3	1 1	
MOVD MOVQ	v,r32/64	1		5	1 1	
MOVD MOVQ	v,m	1		3	1 1	
MOVQ	V,V	1		1	0.5	
MOVDQA	x, x	1		0-1	0.5	
MOVDQA MOVDQU	x, m128	1		3	1 1	
MOVDQA MOVDQU	m128, x	1		3	1 1	
LDDQU	x, m128	1		3	1 1	
MOVDQ2Q	mm, x	1		1	0.5	
MOVQ2DQ	x,mm	1		1	0.5	
MOVNTQ	m64,mm	1		3	1	
MOVNTDQ	m128,x	1		3	1	
MOVNTDQA	x, m128	1		3	1	

PACKSSWB/DW					0.5	
PACKUSWB	V,V	1		1	0.5	
PACKUSDW	X,X	1		1	0.5	
PUNPCKH/LBW/WD/DQ	V,V	1		1	0.5	
PUNPCKH/LQDQ	V,V	1		1	0.5	
PMOVSX/ZX BW BD BQ	v v	4		4	0.5	
DW DQ PMOVSX/ZX BW BD BQ	x,x	1		1	0.5	
DW DQ	x,m	1		1	1	
PSHUFB	mm,mm	1		1	0.5	
PSHUFB	x,x	1		1	1	
PSHUFW	mm,mm,i	1		1	0.5	
PSHUFL/HW	x,x,i	1		1	0.5	
PSHUFD	x,x,i	1		1	0.5	
PALIGNR	x,x,i	1		1	0.5	
PBLENDVB	x,x,xmm0	2		4	4	
PBLENDVB	x,m,xmm0	3			4	
PBLENDW	x,x/m,i	1		1	0.5	
MASKMOVQ	mm,mm	1		~300	1	
MASKMOVDQU	x,x	3		~330	4	
PMOVMSKB	r32,v	1		4	1	
PINSRW	v,r32,i	1		4	1	
PINSRB/D/Q	x,r32,i	1		4	1	
PINSRB/D/Q	x,m8,i	1			1	
PEXTRB/W/D/Q	r,v,i	1		4	1	
PEXTRB/W/D/Q	m,v,i	4			4	
Arithmetic instructions						
PADD/SUB(U)(S)B/W/D	V,V	1		1	0.5	
PADDQ PSUBQ	V,V	1		2	1	
PHADD(S)W PHSUB(S)W	mm, mm	5		7	7	
PHADD(S)W PHSUB(S)W	x, x/m	7		6	6	+1 if mem
PHADDD PHSUBD	V,V	3		4	4	+1 if mem
PCMPEQ/GTB/W/D	V,V	1		1	0.5	
PCMPEQQ	X , X	1		2	1	
PCMPGTQ	X, X	1		2	1	
PMULL/HW PMULHUW	V,V	1		4	1	
PMULHRSW	V,V	1		4	1	
PMULLD	X, X	1		5	2	
PMULDQ	X, X	1		4	1	
PMULUDQ	V,V	1		4	1	
PMADDWD	V,V	1		4	1	
PMADDUBSW	V,V	1		4	1	
PSADBW	V,V	1		4	1	
MPSADBW	x,x,i	3		5	4	+1 if mem
PAVGB/W	V,V	1		1 1	0.5	
PMIN/MAXUB	V,V	1		1	0.5	
PMIN/PMAX SB/SW/SD						
UB/UW/UD	x,x	1		1	0.5	
PHMINPOSUW	x,x x,x	1		5	2	
PABSB PABSW PABSD	V,V	1		1	0.5	
PSIGNB PSIGNW	•,•			'	0.0	
PSIGND	V,V	1		1	0.5	
		1	'	. '		

Logic instructions					
PAND(N) POR PXOR	V,V	1	1	0.5	
PTEST	x,x	1	1	1	
PSLL/RL/RAW/D/Q	V,V	1	2	1	
PSLL/RL/RAW/D/Q	v,i	1	1	0.5	
PSLL/RLDQ	x,i	1	1	0.5	
String instructions					
PCMPESTRI	x,x,i	11	26	10	+1 if mem
PCMPESTRM	x,x,i	10	17	12	+1 if mem
PCMPISTRI	x,x,i	6	17	8	+1 if mem
PCMPISTRM	x,x,i	5	12	7	+1 if mem
Encryption instructions					
PCLMULQDQ	x,x,i	3	6	4	
AESDEC	x,x	1	4	1	
AESDECLAST	x,x	1	4	1	
AESENC	x,x	1	4	1	
AESENCLAST	x,x	1	4	1	
AESIMC	x,x	1	4	1	
AESKEYGENASSIST	x,x,i	1	4	1	
SHA1RNDS4	x,x,i	1	5	2	
SHA1NEXTE	x,x	1	3	1	
SHA1MSG1	x,x	1	3	1	
SHA1MSG2	x,x	1	3	1	
SHA256RNDS2	x,x	3	8	4	
SHA256MSG1	x,x	1	3	1	
SHA256MSG2	X,X	1	3	1	
Other					
EMMS		19		38	

Floating point XMM instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVAPS/D	x, x	1		0-1	0.5	
MOVAPS/D	x,m128	1		3	1	
MOVAPS/D	m128,x	1		3	1	
MOVUPS/D	x,m128	1		3	1	
MOVUPS/D	m128,x	1		3	1	
MOVSS/D	X, X	1		1	0.5	
MOVSS/D	x,m32/64	1		3	1	
MOVSS/D	m32/64,x	1		3	1	
MOVHPS/D MOVLPS/D	x,m64	1		4	1	
MOVHPS/D MOVLPS/D	m64,x	1		3	1	
MOVLHPS MOVHLPS	x,x	1		1	0.5	
BLENDPS/PD	x,x/m,i	1		1	0.5	
BLENDVPS/PD	x,x,xmm0	2		4	4	+1 if mem
NSERTPS	x,x,i	1		1	0.5	
INSERTPS	x,m32,i	3		4	4	

EVED A OTDO		1 4	I	1 4	1	1
EXTRACTPS	r32,x,i	1		4	1	
EXTRACTPS	m32,x,i	1		4	2	
MOVMSKPS/D	r32,x	1		4	1	
MOVNTPS/D	m128,x	1		3	1	
SHUFPS	x,x,i	1		1	0.5	
SHUFPD	x,x,i	1		1	0.5	
MOVDDUP	x, x	1		1	0.5	
MOVSH/LDUP	x, x	1		1	0.5	
UNPCKH/LPS/PD	x, x	1		1	0.5	
Conversion						
CVTPD2PS	x, x	1		4	1	
CVTPS2PD	x, x	1		4	1	
CVTSD2SS	x, x	1		4	2	
CVTSS2SD	x, x	1		4	2	
CVTDQ2PS	x, x	1		4	1	
CVT(T) PS2DQ	x, x	1		4	1	
CVTDQ2PD		1		4	1	
CVTDQ2FD CVT(T)PD2DQ	X, X	1		4	1 1	
CVT(1)FD2DQ CVTPI2PS	X, X	-			2	
	x,mm	1		4	1	
CVT(T)PS2PI	mm,x	1		4	1	
CVTPI2PD	x,mm	1		4	1	
CVT(T) PD2PI	mm,x	1		4	1	
CVTSI2SS	x,r32	1		6	2	
CVT(T)SS2SI	r32,x	1		5	1	
CVTSI2SD	xm,r32	1		6	2	
CVT(T)SD2SI	r32,x	1		5	1	
Arithmetic						
ADDSS/SD/PS/PD	X, X	1		3	1	
SUBSS/SD/PS/PD	X, X	1		3	1	
ADDSUBPS/PD	x, x	1		3	1	
HADDPS HSUBPS	X, X	4		6	6	+1 if mem
HADDPD HSUBPD	x, x	4		5	5	+1 if mem
MULSS/SD/PS/PD	X, X	1		4	1	
DIVSS	x, x	1		11	7	
DIVSD	x, x	1		14	10	
DIVPS	x, x	1		16	12	
DIVPD	x, x	1		22	18	
RCPSS	x, x	1		4	1	
RCPPS	x, x	1		5	2	
CMPccSS/SD/PS/PD	x, x	1		3	1	
(U)COMISS/SD		1		4		
MAXSS/SD/PS/PD	X, X			3	1	
	X, X	1		3		
MINSS/SD/PS/PD	X, X	1			1	
ROUNDSS/SD/PS/PD	x,x,i	1		4	1	
DPPS	x,x,i	9		17	10	+1 if mem
DPPD	x,x,i	6		5	5	+1 if mem
Math						
SQRTSS	X, X	1		12	8	
SQRTPS				18	14	
	X, X	1				
SQRTSD	X, X	1		18	14	

SQRTPD	x, x	1	30	26	
RSQRTSS	x, x	1	4	1	
RSQRTPS	x, x	1	5	2	
Logic					
ANDPS/D	x, x	1	1	0.5	
ANDNPS/D	x, x	1	1	0.5	
ORPS/D	x, x	1	1	0.5	
XORPS/D	X, X	1	1	0.5	
Other					
LDMXCSR	m32	6	18	18	
STMXCSR	m32	3	12	12	
FXSAVE	m4096	130		106	32 bit mode
FXSAVE	m4096	156		116	64 bit mode
FXRSTOR	m4096	110		136	32 bit mode
FXRSTOR	m4096	136		146	64 bit mode
XSAVE	m	193		119	32 bit mode
XSAVE	m	452		315	64 bit mode
XRSTOR	m	156		220	32 bit mode
XRSTOR	m	173		237	64 bit mode
XSAVEC	m	447		350	32 bit mode
XSAVEC	m	449		370	64 bit mode
XSAVEOPT	m	193		118	32 bit mode
XSAVEOPT	m	219		128	64 bit mode

Intel Knights Landing

List of instruction timings and µop breakdown

Explanation of column headings:

Instruction: Name of instruction. Multiple names mean that these instructions have the

same data. Instructions with or without V name prefix behave the same unless

otherwise noted.

Operands: i = immediate data, r = register, mm = 64 bit mmx register, x = 128 bit xmm

register, mm/x = mmx or xmm register, y = 256 bit ymm register, z = 512 bit zmm register, v = any vector register (mmx, xmm, ymm, zmm), k = mask register. same = same register for both operands. m = memory operand, m32 = 32-

bit memory operand, etc.

μορs: The number of μops from the decoder or ROM. A μop that goes to multiple

units is counted as one.

Unit: Tells which execution unit is used. Instructions that use the same unit cannot

execute simultaneously.

IP0 and IP1 means integer port 0 or 1 and their associated pipelines

IP0/1 means that either integer unit can be used.

IP0+1 means that the μop is split in two, using both units.

Mem means memory execution cluster

FP0 means floating point port 0 (includes multiply, divide, convert and shuffle).

FP1 means floating point port 1.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give a similar de-

lay.

Some instructions have a range of latencies. For example VPSHUFD has a latency of 3-6. The short latency is measured in a chain of similar instructions. The long latency is measured when the input comes from an instruction of a different type and the output goes to an instruction of a different type, for example a move instruction. The long latency will apply in most cases. Division and some square root instructions have latencies that depend on the values of

the operands.

Reciprocal throughput: The average number of clock cycles per instruction for a series of independent

instructions of the same kind in the same thread. Delays in the decoders are included in the latency and throughput timings. Values of 4 or more are often caused by bottlenecks in the decoders and microcode ROM rather than the

execution units.

Integer instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOV	r,r	1	IP0/1	1	0.5	
MOV	r,i	1	IP0/1	1	0.5	
MOV	r,m	1	Mem	4	1	All addr. modes
MOV	m,r	1	Mem	3	1	All addr. modes
MOV	m,i	1	Mem		1	
MOVNTI	m,r	1	Mem		2	
MOVSX MOVZX MOVSXD	r16,r8	2	IP0	7	7	
MOVSX MOVZX MOVSXD	r16,m8	3	IP0	7	8	

MOVSX MOVZX MOVSXD r32/64.r			_	_			
CMOVcc	MOVSX MOVZX MOVSXD	r32/64,r	1	IP0	1	1	
CMOVcc	MOVSX MOVZX MOVSXD	r32/64,m	1	IP0	4	1	
CMOVcc	CMOVcc	r,r	1	IP0/1	2	1	
XCHG	CMOVcc		1			1	
XCHG			3	IP0/1	8		
XLAT				" "			Implicit lock
PUSH		1,					Implion look
PUSH		<u>_</u>			O	1	
PUSH			=	IDO.4			
PUSHA(D)			=				
PUSHA(D) POP CE/R)SP 2		m		I I			
POP	, ,			IP0+1			
POP	, ,						Not in x64 mode
POP	I	· ·					
POPF(D/Q)	POP	(E/R)SP					
POPA(D)	POP	m	6			9	
LAHF SAHF	POPF(D/Q)		21			48	
LAHF SAHF	POPA(D)		17			15	Not in x64 mode
SAHF SALC 2	1 1		1	IP0	1		
SALC			1	I I			
LEA	I		=				Not in x64 mode
LEA		r [r+d]		IPO/1			Not in Xo i modo
LEA	I			I I			
LEA	I		-	I I			
LEA			-	I I	2		
BSWAP			=	IP0/1	_	0.5	
MOVBE r16,m16 1 <th< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td></th<>							
MOVBE r/m32/64 1 1 MOVBE m,r 1 1 PREFETCHTO/1/2 m 1 0.5 PREFETCHTO/1/2 m 1 0.5 PREFETCHNTW m 1 0.5 PREFETCHNTW m 1 0.5 LFENCE 2 8 MFENCE 2 17 SFENCE 1 10 Arithmetic instructions 1 1 ADD SUB r,r/i 1 1PO/1 1 ADD SUB r,r,m 1 1PO/1, Mem 1 ADD SUB r,r/i 1 1PO/1, Mem 7 1 ADC SBB r,r/i 1 1PO/1, Mem 7 1 ADC SBB r,r/i 1 1PO/1, Mem 7 1 ADC SBB r,r/i 1 1PO/1 2 2 ADC SBB m,r/i 1 7 2 ADCX ADOX r64,r64 <td< td=""><td></td><td>·</td><td>=</td><td> IP0 </td><td>1</td><td></td><td></td></td<>		·	=	IP0	1		
MOVBE m,r 1 0.5 PREFETCHNTA m 1 0.5 PREFETCHT0/1/2 m 1 0.5 PREFETCHNTW m 1 0.5 PREFETCHNTW m 1 0.5 LFENCE 2 8 8 MFENCE 2 17 10 SFENCE 1 10 10 Arithmetic instructions 7 1 10 ADD SUB r,r/i 1 1PO/1 1 0.5 ADD SUB r,m 1 1PO/1 1 0.5 1 ADD SUB m,r/i 1 1PO/1 <			1				
PREFETCHNTA m 1 0.5 PREFETCHT0/1/2 m 1 0.5 PREFETCHNTW m 1 0.5 LFENCE 2 8 MFENCE 2 17 SFENCE 1 10 Arithmetic instructions ADD SUB r,r/i 1 IP0/1 1 0.5 ADD SUB r,m 1 IP0/1, Mem 1 1 ADD SUB m,r/i 1 IP0/1, Mem 7 1 ADC SBB r,r/i 1 IP0+1 2 2 ADC SBB r,m 1 IP0+1 2 2 ADC SBB m,r/i 1 T 2 2 ADC SBB m,r/i 1 IP0+1 2 2 ADC ADOX r64,r64 1 IP0+1 2 2 ADC X ADOX r64,r64 1 IP0+1 2 6 due to decoder CMP		r/m32/64	1				
PREFETCHT0/1/2 m 1 0.5 PREFETCHNTW m 1 0.5 LFENCE 2 8 MFENCE 2 17 SFENCE 1 10 Arithmetic instructions ADD SUB r,r/i 1 IP0/1 1 0.5 ADD SUB r,m 1 IP0/1, Mem 1 1 ADD SUB m,r/i 1 IP0/1, Mem 7 1 ADC SBB r,r/i 1 IP0/1, Mem 7 1 ADC SBB r,m 1 IP0+1 2 2 ADC SBB r,m 1 7 2 2 ADC SBB m,r/i 1 7 2 2 ADC SBB m,r/i 1 7 2 2 ADC ADOX r32,r32 1 IP0+1 2 2 ADC X ADOX r64,r64 1 IP0+1 2 6 due to decoder	MOVBE	m,r	1			1	
PREFETCHNTW m 1 0.5 LFENCE 2 17 MFENCE 2 17 SFENCE 1 10 Arithmetic instructions ADD SUB ADD SUB F,r/i ADD SUB F,rm ADD SUB M,r/i ADD SUB M,r/i ADD SUB M,r/i ADD SUB M,r/i ADC SBB F,rm ADC SBB F,rm ADC SBB F,rm M,r/i ADC SBB ADC SBB ADC SB	PREFETCHNTA	m	1			0.5	
LFENCE 2 8 MFENCE 1 17 SFENCE 1 10 Arithmetic instructions r,r/i 1 IPO/1 1 0.5 ADD SUB r,m 1 IPO/1, Mem 1 1 ADD SUB m,r/i 1 IPO/1, Mem 7 1 ADC SUB m,r/i 1 IPO/1, Mem 7 1 ADC SBB r,r/i 1 IPO+1 2 2 ADC SBB m,r/i 1 7 2 ADCX ADOX r32,r32 1 IPO+1 2 2 ADCX ADOX r64,r64 1 IPO+1 2 2 ADCX ADOX r64,r64 1 IPO+1 2 6 due to decoder CMP r,r/i 1 IPO/1 1 0.5 1 CMP m,r/i 1 IPO/1 1-2 1 INC DEC r 1 IPO/1 1-2 1 NEG NOT m 1 7 1 1 <td>PREFETCHT0/1/2</td> <td>m</td> <td>1</td> <td></td> <td></td> <td>0.5</td> <td></td>	PREFETCHT0/1/2	m	1			0.5	
MFENCE SFENCE 2 17 10 Arithmetic instructions r,r/i 1 IP0/1 1 0.5 ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADC SBB F,rm 1 m,r/i 1 IP0/1, Mem 7 1 1 IP0/1, Mem 7 1 1 IP0/1, Mem 7<	PREFETCHNTW	m	1			0.5	
MFENCE SFENCE 2 17 10 Arithmetic instructions r,r/i 1 IP0/1 1 0.5 ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADC SBB F,rm m,r/i 1 IP0/1, Mem 7 1 1 IP0/1, Mem 7 1 1 ADC SBB 7 1 IP0/1, Mem 7	LFENCE		2			8	
SFENCE	MFENCE		2			17	
Arithmetic instructions							
ADD SUB r,r/i 1 IP0/1 1 0.5 ADD SUB r,m 1 IP0/1, Mem 1 ADD SUB m,r/i 1 IP0/1, Mem 7 1 ADC SBB r,r/i 1 IP0+1 2 2 ADC SBB m,r/i 1 7 2 ADCX ADOX r32,r32 1 IP0+1 2 2 ADCX ADOX r64,r64 1 IP0+1 2 6 due to decoder CMP r,r/i 1 IP0/1 1 0.5 due to decoder CMP m,r/i 1 IP0/1 1 0.5 due to decoder CMP r,r/i 1 IP0/1 1 0.5 due to decoder CMP r,r/i 1 IP0/1 1-2 1 IP0/1 1-2 1 INC DEC r 1 IP0/1 1 0.5 IP0/1 1 0.5 INC DEC NEG NOT m							
ADD SUB r,r/i 1 IP0/1 1 0.5 ADD SUB r,m 1 IP0/1, Mem 1 ADD SUB m,r/i 1 IP0/1, Mem 7 1 ADC SBB r,r/i 1 IP0+1 2 2 ADC SBB m,r/i 1 7 2 ADCX ADOX r32,r32 1 IP0+1 2 2 ADCX ADOX r64,r64 1 IP0+1 2 6 due to decoder CMP r,r/i 1 IP0/1 1 0.5 due to decoder CMP m,r/i 1 IP0/1 1 0.5 due to decoder CMP r,r/i 1 IP0/1 1 0.5 due to decoder CMP r,r/i 1 IP0/1 1-2 1 IP0/1 1-2 1 INC DEC r 1 IP0/1 1 0.5 IP0/1 1 0.5 INC DEC NEG NOT m	Arithmetic instructions						
ADD SUB r,m 1 IP0/1, Mem 1 1 ADD SUB 1 IP0/1, Mem 7 1 IP0/1, Mem 7 1 IP0/1, Mem 7 1 IP0/1, Mem 7 1 IP0/1, Mem 7 1 IP0/1 IP0/1 1 IP0/1 1 IP0/1 <td></td> <td>r.r/i</td> <td>1</td> <td>IP0/1</td> <td>1</td> <td>0.5</td> <td></td>		r.r/i	1	IP0/1	1	0.5	
ADD SUB m,r/i 1 IP0/1, Mem 7 1 ADC SBB r,r/i 1 IP0+1 2 2 ADC SBB r,m 1 7 2 ADCX SBB m,r/i 1 7 2 ADCX ADOX r32,r32 1 IP0+1 2 2 ADCX ADOX r64,r64 1 IP0+1 2 6 due to decoder CMP r,r/i 1 IP0/1 1 0.5 1 CMP m,r/i 1 IP0/1 1-2 1 1 INC DEC r 1 IP0/1 1-2 1 1 0.5 INC DEC NEG NOT m 1 7 1 1 Not in x64 mode DAA 20 17 Not in x64 mode Not in x64 mode				I I	•		
ADC SBB ADC SBB ADC SBB ADC SBB ADC SBB ADC SBB ADC SBB ADC SBB ADCX ADOX AD			-	· ·	7		
ADC SBB r,m 1 2 ADC SBB m,r/i 1 7 2 ADCX ADOX r32,r32 1 IP0+1 2 2 ADCX ADOX r64,r64 1 IP0+1 2 6 due to decoder CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1-2 1 INC DEC r 1 IP0/1 1-2 1 NEG NOT r 1 IP0/1 1 0.5 INC DEC NEG NOT m 1 7 1 AAA AAS 13 13 Not in x64 mode DAA 20 17 Not in x64 mode			-				
ADC SBB					2		
ADCX ADOX r32,r32 1 IP0+1 2 2 due to decoder CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1-2 1 INC DEC r 1 IP0/1 1 0.5 INC DEC NEG NOT m 1 7 1 AAA AAS DAA 13 Not in x64 mode DAA			=		7		
ADCX ADOX r64,r64 1 IP0+1 2 6 due to decoder CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 1 1 INC DEC r 1 IP0/1 1-2 1 NEG NOT r 1 IP0/1 1 0.5 INC DEC NEG NOT m 1 7 1 AAA AAS 13 13 Not in x64 mode DAA 20 17 Not in x64 mode				IDO: 4			
CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 1 1 INC DEC r 1 IP0/1 1-2 1 NEG NOT r 1 IP0/1 1 0.5 INC DEC NEG NOT m 1 7 1 AAA AAS 13 13 Not in x64 mode DAA 20 17 Not in x64 mode	I			I I			1
CMP m,r/i 1 1 1 INC DEC r 1 IP0/1 1-2 1 NEG NOT r 1 IP0/1 1 0.5 INC DEC NEG NOT m 1 7 1 AAA AAS 13 13 Not in x64 mode DAA 20 17 Not in x64 mode		•		I I			due to decoder
INC DEC				IP0/1	1		
NEG NOT r 1 IP0/1 1 0.5 INC DEC NEG NOT m 1 7 1 AAA AAS 13 13 Not in x64 mode DAA 20 17 Not in x64 mode	I		=				
INC DEC NEG NOT m 1 7 1 AAA AAS 13 13 Not in x64 mode DAA 20 17 Not in x64 mode		r	-	I I			
AAA AAS 13 Not in x64 mode DAA 20 17 Not in x64 mode		r	1	IP0/1		0.5	
DAA 20 17 Not in x64 mode		m				1	
	AAA AAS		13		13		Not in x64 mode
DAS 21 17 Not in x64 mode	DAA		20		17		Not in x64 mode
	DAS		21		17		Not in x64 mode

		3	5			
AAD		4		8		Not in x64 mode
AAM		10		30	14	Not in x64 mode
MUL IMUL	r8	3	IP0	8		
MUL IMUL	r16	4	IP0	8		
MUL IMUL	r32	3	IP0	8		
MUL IMUL	r64	3	IP0	8		
IMUL	r16,r16	2	IP0	7	7	
IMUL	r32,r32	1	IP0	3	1	
IMUL	r64,r64	1	IP0	5	2	
IMUL	r16,r16,i	2	IP0	7	7	
IMUL	r32,r32,i	1	IP0	3	1	
IMUL	r64,r64,i	1	IP0	5	2	
MUL IMUL	m8	3	IP0		_	
MUL IMUL	m16	3	IP0			
MUL IMUL	m32	4	IP0			
MUL IMUL	m64	3	IP0			
DIV	r/m8	9	IP0, FP0	30	12	
DIV	r/m16	12	IP0, FP0	30-35	13-15	
DIV	r/m32	12	IP0, FP0	29-42	13-13	
DIV	r/m 64	23	IP0, FP0	39-95	22-95	
IDIV	r/m8	23 26	IP0, FP0	39-95	22-95	
IDIV	r/m16	20 29	IP0, FP0	38-42	20	
IDIV		29 29		37-49	22-26	
IDIV	r/m32		IPO, FPO			
	r/m64	44	IP0, FP0	53-108	36-107	
CBW		2	IP0	7		
CWDE		1	IP0	1		
CDQE		1	IP0	1 1		
CWD		2	IP0	7		
CDQ		1	IP0	1		
CQO	10.10	1	IP0	1 1	-	
POPCNT	r16,r16	2		7	7	
POPCNT	r32,r32	1		3	1	
POPCNT	r64,r64	1		3	1	
CRC32	r32,r8	2		7	2	
CRC32	r32,r16	1		6	6	
CRC32	r32,r32	1		3	1	
CRC32	r64,r64	1		6	1	
Logic instructions		4	100/4	_	0.5	
AND OR XOR	r,r/i	1	IP0/1	1	0.5	
AND OR XOR	r,m	1	IP0/1, Mem		1	
AND OR XOR	m,r/i	1	IP0/1, Mem	6	1	
TEST	r,r/i	1	IP0/1	1	0.5	
TEST	m,r/i	1	IP0/1, Mem	_	1	
SHR SHL SAR	r,i/cl	1	IP0	1	1	
SHR SHL SAR	m,i/cl	1	IP0		1	
ROR ROL	r,i/cl	1	IP0	1	1	
ROR ROL	m,i/cl	1	IP0		1	
RCR	r,1	7	IP0	10	10	
RCL	r,1	1	IP0	2	2	
RCR	r,i/cl	11	IP0	13	13	
RCR	m,i/cl	14	IP0	13		
RCL	r,i/cl	13	IP0	13	13	

		. J			, .
RCL	m,i/cl	16	IP0	16	16
SHLD	r16,r16,i	10	IP0	11	11
SHLD	r16,m16,i	13	IP0	13	13
SHLD	r32,r32,i	1	IP0	2	2
SHLD	r32,m32,i	6	IP0	9	9
SHLD	r64,r64,i	10	IP0	11	11
SHLD	r64,m64,i	13	IP0	13	13
SHLD	r16,r16,cl	9	IP0	11	11
SHLD	r16,m16,cl	12	IP0	13	13
SHLD	r32,r32,cl	2	IP0	7	7
SHLD	r32,m32,cl	6	IP0	9	9
SHLD	r64,r64,cl	9	IP0	11	11
SHLD	r64,m64,cl	12	IP0	13	13
SHRD	r16,r16,i	8	IP0	11	11
SHRD	r16,m16,i	11	IP0	12	12
SHRD	r32,r32,i	2	IP0	7	7
SHRD	r32,m32,i	6	IP0	9	9
SHRD	r64,r64,i	10	IP0	11	11
SHRD	r64,m64,i	13	IP0	15	15
SHRD	r16,r16,cl	7	IP0	11	11
SHRD	r16,m16,cl	10	IP0	12	12
SHRD	r32,r32,cl	2	IP0	7	7
SHRD	r32,r32,cl	6	IP0	9	9
SHRD	r64,r64,cl	9	IP0	11	11
SHRD	r64,m64,cl	12	IP0	14	14
SHLX SHRX SARX	r,r,r	1	IP0	2	1
RORX	r,r,i	1	IP0	1	1
BT		1	IP0+1		1
BT	r,r/i m,r	7	IFUT I	10	10
BT	m,i	1		10	10
BTR BTS BTC	r,r/i	1	IP0+1		1
BTR BTS BTC		8	IFOTI	11	11
BTR BTS BTC	m,r m,i	1	IP0+1	1	1
BSF BSR	r,r/m	10	IP0/1	11	11
SETcc	,	1	IP0+1	2	1
CLC STC	r/m	1	IP0		1
CMC		1	IP0	1	1
CLD		4	IP0/1	I	8
STD		5	IP0/1		36
LZCNT	r r/m	1	150/1	2	1
TZCNT	r,r/m	1		3	1
ANDN	r,r/m	1		1	0.5
ANDN	r,r,r			1 1	
BLSI BLSMSK BLSR	r,r,m	1		1	1 1
1	r,r/m	-		7	7
BEXTR BEXTR	r,r,r	2 3		'	8
BZHI	r,r,m	1		2	1
	r,r,r	1		3	1
PDEP	r,r,r			3	1
PEXT	r,r,r	1		၂ ၁	1
Control transfer instruction	 ne				
JMP	short/near	1	IP1		2
JMP	r	1	IP1		2
5.711	l '			I	

JMP Conditional jump J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL RET RET BOUND INTO	m(near) short/near short short short near r m	1 1 2 7 8 1 1 3 1 1 10 4	IP1 IP1		2 1-2 7-18 14-23 14-23 2 2 14 2 2 11	Not in x64 mode Not in x64 mode
Of when an increase we are						
String instructions		2			O	
LODS		3			8 ~2n	
REP LODS STOS		~4n 2			~2n 7	
REP STOS MOVS		~0.07B 5			~0.054B 9	per byte, best case
REP MOVS		5 ∼ 0.1B			9 ~0.08B	
SCAS		3			~0.06B 8	per byte, best case
REP SCAS		~5n			~3n	
CMPS		6			9	
REP CMPS		~6n			~3n	
Synchronization instruction						
XADD	m,r	6		9		
LOCK XADD	m,r	4		24		
LOCK ADD	m,r	1		13		
CMPXCHG	m,r	8		11		
LOCK CMPXCHG	m,r	6 13		26 15		
CMPXCHG8B LOCK CMPXCHG8B	m,r	11		29		
CMPXCHG16B	m,r	19		31		
LOCK CMPXCHG16B	m,r m,r	17		48		
	,.					
Other						
NOP (90)		1	IP0/1		0.5	
Long NOP (0F 1F)		1	IP0/1		0.5	
PAUSE		6		25		
ENTER	а,0	15		14		
ENTER	a,b	19+6b		66+4b		
LEAVE		4		8	5	
XGETBV		7			14	
CPUID		40-83		63-270		
RDTSC		15		30		
RDTSCP		17		36		
RDPMC		19		20	000	
RDRAND	r	17			200	
RDSEED	r	17			200	

Floating point x87 instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions					tinougnput	
FLD	r	1		2	1	
FLD	m32/m64	1		8	1	
FLD	m80	5		9	12	
FBLD	m80	52			66	
FST(P)	r	1		2	1	
FST	m32/m64	5		14	13	
FSTP	m32/m64	6		14	13	
FSTP	m80	8		11	14	
FBSTP	m80	189			264	
FXCH	r	3	FP0+1	9	9	
FILD		1	FFUTI	5	2	
l .	m			18	12	
FIST(P)	m	6		10		
FISTTP	m	6			14	
FLDZ		1			1	
FLD1		2			10	
FLDPI FLDL2E etc.		2			10	
FCMOVcc	r	3		9		
FNSTSW	AX	3			12	
FNSTSW	m16	4			13	
FLDCW	m16	3			15	
FNSTCW	m16	5			15	
FINCSTP FDECSTP		1		1	1	
FFREE(P)		1			1	
Arithmetic instructions						
FADD(P) FSUB(R)(P)	r	1	FP0	6	1.5	
FADD(P) FSUB(R)(P)	m	1	FP0		12	
FMUL(P)	r	1	FP0	7	2	
FMUL(P)	m	1	FP0	7	12	
FDIV(R)(P)	r	1	FP0	41	37	
FDIV(R)(P)	m	1	FP0	41	44	
FABS		1		2		
FCHS		1		2		
FCOM(P) FUCOM	r	1			1	
FCOM(P) FUCOM	m	1			2	
FCOMPP FUCOMPP		1			1	
FCOMI(P) FUCOMI(P)	r	3			9	
FIADD FISUB(R)	m	3			17	
FIMUL	m	3			17	
FIDIV(R)	m	3			41	
FICOM(P)	m	3			8	
FTST		1			1	
FXAM		1			1	
FPREM		27		26-47	25-32	
FPREM1		27		33-72	25-32	
FRNDINT		18		36	36	
Math						
	+	30		31		
FSCALE		,)()				

FSQRT		1	15-42	11-38	
FSIN FCOS	16-	100	40-250	40-250	
FSINCOS	17-	110	50-250	50-250	
F2XM1	9-	24	100-400		
FYL2X	34	-61	126-190	98-190	
FYL2XP1	6	31	190	190	
FPTAN	17-	100	50-280	50-280	
FPATAN	33	-63	125-265	125-265	
Other					
FNOP		1		1	
WAIT		2		7	
FNCLEX		5		26	
FNINIT	,	15		63	

Integer MMX and XMM instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVD MOVQ	r32/64,mm/x	1		4	1	
MOVD MOVQ	mm/x,r32/64	1		5	1	
MOVD MOVQ	m32/64,mm/x	1	Mem	5	1	
MOVD MOVQ	mm/x,m32/64	1	Mem	5	0.5	
MOVQ	mm/x, mm/x	1	FP0/1	2	0.5	
(V)MOVDQA/U	V,V	1	FP0/1	2	0.5	
(V)MOVDQA/U	v,m	1	Mem	5	0.5	
VMOVDQA/U	v{k},m	1	Mem	7	0.5	
(V)MOVDQA/U	m,v	1	Mem	5	1	
VMOVDQA/U	m{k},v	1	Mem	9	1	
LDDQU	x, m128	1	Mem	5	0.5	
MOVDQ2Q	mm, x	1	FP0/1	2	0.5	
MOVQ2DQ	x,mm	1	FP0/1	2	0.5	
MOVNTQ	m64,mm	1	Mem	~650	1	
MOVNTDQ	m128,x	1	Mem	~550	1	
(V)MOVNTDQA	v, m	1	Mem	5	0.5	
MASKMOVQ	mm,mm	6	Mem	~550	12	
MASKMOVDQU	X,X	6	Mem	~550	12	
VPMASKMOVD/Q	v,v,m	5	Mem	7	9	
VPMASKMOVD/Q	m,v,v	4	Mem	6	8	
VPACKSSWB/DW						
VPACKUSWB/DW	mm/x, mm/x	1	FP0	2-6	1	
VPACKSSWB/DW						
VPACKUSWB/DW	y/z,y/z,y/z	5		11-14	9	
VPACKSSWB/DW	, ,	•				
VPACKUSWB/DW	y/z,y/z,m	6			9	
PUNPCKH/LBW/WD/DQ	mm/x, mm/x	1	FP0	2-6	1	
/PUNPCKH/LBW/WD	y/z,y/z,y/z	5		11-14	9	
VPUNPCKH/LBW/WD	y/z,y/z,m	6			9	
PUNPCKH/LQDQ	mm/x, mm/x	1	FP0	2-6	1	
VPUNPCKH/L(Q)DQ	y/z,y/z,y/z	1	FP0	4-7	2	
(V)PMOVSX BW BD BQ DW DQ	V,V	2		8	7-8	
(V)PMOVZX BW BD BQ DW DQ	V,V	1		3	2	

		rangino L	-arraning		
VPMOV QB QW QD DB DW	V,V	1	FP0	3	1
VPMOV(U)S QB QW QD DB DW	V,V	2		8	7
PSHUFB	mm,mm	1	FP0	2-6	1
PSHUFB	X,X	5	FP0	11-13	10
PSHUFB	x,m	6	FP0		10
VPSHUFB	у,у, у	12	FP0	23-25	12
VPSHUFB	y,y,m	13	FP0	20 20	13
PSHUFW	mm,mm,i	1	FP0	2-6	1
PSHUFL/HW	x,x,i	1	FP0	2-6	1
VPSHUFL/HW	y,y,i	4	FP0	11-14	8
VPSHUFL/HW	y,y,ı y,m,i	5	FP0	11-14	9
(V)PSHUFD		1	FP0	3-6	1
PALIGNR	V,V,İ	1	FP0	2-6	1
	mm,mm,i	1		2-6	
PALIGNR	x,x,i		FP0		2
VPALIGNR	y,y,y,i	5	FP0	11-14	9
VPALIGNR	y,y,m,i	6	FP0	0.0	9
VALIGND/Q	z,z,z,i	1	FP0	3-6	1
VPCOMPRESSD/Q	z{k},z	1		3-6	3
VPEXPANDD/Q	z{k},z	1		3-6	3
PBLENDVB	x,x,xmm0	5	FP0	9-10	9
PBLENDVB	x,m,xmm0	6	FP0		9
VPBLENDVB	V,V,V,V	4		8-10	8
PBLENDW	x,x/m,i	1		2	2
VPBLENDW	y,y,y/m,i	1		2	0.5
VPBLENDD	v,v,v/m,i	1	FP0/1	2	0.5
VPBLENDMD/Q	z{k},z,z	1	FP0/1	2	0.5
VPERMD	V,V,V	1	FP0	3-6	1
VPERMQ	v,v,i	1	FP0	3-6	1
VPERM2I128	v,v,v,i	1	FP0	4-7	2
VPERMI2D VPERMT2D	V,V,V	1	FP0	4-7	2
VPERMI2Q VPERMT2Q	V,V,V	1	FP0	4-7	2
VSHUFI32X4	z,z,z,i	1	FP0	4-7	2
VSHUFI64X2	z,z,z,i	1	FP0	4-7	2
PMOVMSKB	r32,mm	4		14	8
PMOVMSKB	r32,x	5		19	8
PMOVMSKB	r32,y	12		26	12
PEXTRB/W/D	r32,x,i	2		8	7
PEXTRQ	r64,x,i	2		8	10
VEXTRACTI128	x,y,i	1	FP0	3-6	1
VEXTRACTI128	m128,y,i	4	-	7	8
PINSRB/W	x,r32,i	1	FP0	5	1
PINSRD	x,r32,i	1	FP0	4	1.5
PINSRQ	x,r64,i	1	FP0	4	6
VINSERTI128	y,y,x,i	1	FP0	3-6	1
VINSERTI32X4	z,z,x,i	1	FP0	3-6	1
VINSERTI64X4	z,z,y,i	1	FP0	3-6	1
VPBROADCASTB/W	۷,X	2		8	7
VPBROADCASTD/Q	V,X V,X	1	FP0	3	1
VBROADCASTI128	y,n y,m128	1	'''	5	0.5
VBROADCASTI32X4	z,m128	1		5	0.5
VBROADCASTI64X4	z,111126 z,m256	1		5	0.5
V DI (OAD OAS I 104A4	۷,۱۱۱۷	'			0.5
Gather and scatter					
	•		•	•	,

		3	3		
VPGATHERDD	x,[r+s*x],x	6			12
VPGATHERDD	y,[r+s*y],y	6			12
VPGATHERDD	z,[r+s*z],z	1			11
VPGATHERQD	x,[r+s*x],x	6			12
VPGATHERQD	x,[r+s*y],x	6			12
VPGATHERQD	y,[r+s*z],y	1			7
VPGATHERDQ	x,[r+s*x],x	6			12
VPGATHERDQ	y,[r+s*x],y	6			12
VPGATHERDQ	z,[r+s*y],z	1			7
VPGATHERQQ	x,[r+s*x],x	6			12
VPGATHERQQ	y,[r+s*y],y	6			12
VPGATHERQQ	z,[r+s*z],z	1			7
VPSCATTERDD	z,[r+s*z],z	4			17
VPSCATTERQD	y,[r+s*z],y	4			11
VPSCATTERDQ	z,[r+s*y],z	4			11
VPSCATTERQQ	z,[r+s*z],z	4			11
VI 30/XI IZIXQQ	2,[1 10 2],2				
Arithmetic instructions					
PADD/SUB(U,S)B/W/D/Q	mm/x, mm/x	1	FP0/1	2	0.5
VPADD/SUB(U,S)B/W/D/Q	v,v,v	1	FP0/1	2	0.5
PHADD(S)W PHSUB(S)W	mm, mm	5		18	9
PHADD(S)W PHSUB(S)W	x, x	6	FP0	28	28
PHADD(S)W PHSUB(S)W	x, m	7			28
PHADD(S)W PHSUB(S)W	y, y	7	FP0	23	9
PHADDD PHSUBD	mm, mm	4		14	8
PHADDD PHSUBD	x, x	3		11	9
VPHADDD VPHSUBD	y,y,y	3		11	8
PCMPEQ/GTB/W/D	mm/x,mm/x	1	FP0/1	2	0.5
VPCMPEQ/GTB/W/D	y,y,y	1	FP0/1	2	0.5
PCMPEQ/GTQ	x, x	1		2	2
VPCMPEQQ	y,y,y	1	FP0/1	2	0.5
VPCMP(U)D/Q	k,z,z,i	1	FP0/1	2	0.5
VPTESTMD/Q	k,z,z	1	FP0/1	2	0.5
PMULL/HW PMULHUW	mm,mm	1	FP0	6	1
PMULL/HW PMULHUW	x, x	1	FP0	7	2
VPMULL/HW VPMULHUW	y,y,y	5	FP0	16	9
PMULHRSW	mm,mm	1	FP0		1
PMULHRSW	x, x	1	FP0	7	2
PMULLD	x, x	1	FP0	7	2
VPMULLD	V,V,V	1	FP0	7	1
PMULDQ	x, x	1	FP0	6	2
VPMULDQ	V,V,V	1	FP0/1	6	0.5
PMULUDQ	mm/x,mm/x	1	FP0/1	6	0.5
VPMULUDQ	V,V,V	1	FP0/1	6	0.5
PMADDWD	mm,mm	1	FP0	6	1
PMADDWD	x, x	1	FP0	7	2
PMADDUBSW	mm,mm	1	FP0	6	1
PMADDUBSW	X, X	1	FP0	7	2
PMADDUBSW	у,у,у	5	FP0	16	9
PSADBW	mm,mm	1	FP0	6	1
PSADBW	x, x	1	FP0	7	2
PSADBW	y,y,y	5	FP0	16	9
MPSADBW	x,x,i	3		9	9

VMPSADBW PAVGB/W PAVGB/W PMIN/MAXUB/SW VPMIN/MAXUB/SW/D/Q PHMINPOSUW PABSB/W/D PABSB/W/D VPABSB/W/D/Q	y,y,y,i mm/x,mm/x y,y,y mm/x,mm/x y,y,y x,x mm,mm x,x y,y	9 1 1 1 1 1 1 1	FP0/1 FP0/1 FP0/1 FP0/1 FP0/1 FP0/1 FP0/1	19 2 2 2 2 3 2 2 2	13 0.5 0.5 2 0.5 2 0.5 2 0.5	
PSIGNB/W/D	mm,mm	1	FP0/1	2 2	0.5 2	
PSIGNB/W/D VPSIGNB/W/D	x,x y,y	1 1	FP0/1 FP0/1	2	0.5	
Logic instructions PAND(N)/OR/XOR VPAND(N)/OR/XOR VPAND(N)/OR/XORD/Q VPTERNLOGD/Q PTEST VPTEST(N)MD/Q VPTEST	mm/x,mm/x y,y,y z,z,z z,z,z,i x,x k,z,z y,y	1 1 1 1 4 1	FP0/1 FP0/1 FP0/1 FP0+1 FP0/1	2 2 2 2 9 2 9	0.5 0.5 0.5 1 9 0.5	
PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q VPSLL/RL/RAW/D/Q VPSLL/RA/RLVD/Q VPROL/RD/Q	mm,mm x,x mm/x,i y,y,i z,z,z z,z,i	1 2 1 4 1	FP0 FP0 FP0 FP0 FP0	2 13 2 11 2 2	1 13 1 8 1	
VPROL/RVD/Q VPLZCNTD/Q VPCONFLICTD/Q	Z,Z,Z Z,Z Z,Z	1 1 1	FP0 FP0 FP0	2 2 3	1 1 1	
String instructions		•	ED0	0.4	04	.4 %
PCMPESTRI PCMPESTRM	x,x,i x,x,i	9 8	FP0 FP0	21 17	21 17	+1 if mem +1 if mem
PCMPISTRI PCMPISTRM	x,x,i x,x,i	6 5	FP0 FP0	17 13	17 13	+1 if mem +1 if mem
Encryption instructions						
PCLMULQDQ AESDEC, AESDECLAST, AESENC, AESENCLAST, AESIMC,	x,x,i	1	FP0	3-6	2	+1 if mem
AESKEYGENASSIST	x,x	1		3-6	2	
Other EMMS		10			13	

Floating point XMM instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
(V)MOVAPS/D	V,V	1	FP0/1	2	0.5	
(V)MOVAPS/D	v,m	1	Mem	5	0.5	

		rangnio	Larianing		
(V)MOVAPS/D	m,v	1	Mem	5	1 1
(V)MOVUPS/D	v,m	1	Mem	5	0.5
(V)MOVUPS/D	m,v	1	Mem	5	1
VMOVAPS/D VMOVUPS/D	z{k},m	1	Mem	7	0.5
VMOVAPS/D VMOVUPS/D	m{k},z	1	Mem	9	1
MOVSS/D	x, x	1	FP0/1	2	0.5
MOVSS/D	x,m	1	Mem	5	0.5
MOVSS/D	m,x	1	Mem	5	1
MOVHPS/D	x,m64	1	Mem	6	1.5
MOVHPS/D	m64,x	4	Mem	9	1
MOVLPS/D	x,m64	1	Mem	6	1.5
MOVLPS/D	m64,x	1	Mem	6	1
(V)MOVNTPS/D	m,v	1	Mem	~500	
MOVMSKPS/D	r32,x	2	FP0	6	7
MOVLHPS MOVHLPS	x,x	1	FP0	3-6	1
MOVDDUP	x,x	1	FP0	3-6	1
MOVDDUP	x,m	1		14	
VMOVDDUP	V,V	1		3-6	1
(V)MOVSH/LDUP	v,v	1	FP0	3-6	1
VBROADCASTSS/D	v,x	1		3-6	1
VBROADCASTSS/D	v,m	1		5	0.5
VBROADCASTF128	y,m128	1		5	0.5
VBROADCASTF32X4	v,m128	1		5	0.5
VBROADCASTF64X4	z,m256	1		5	0.5
UNPCKH/LPS/D	X,X	1	FP0	2	2
VUNPCKH/LPS/D	V,V,V	1	FP0	4-7	2
INSERTPS	x,x,i	2	'''	8	7
INSERTPS	x,m32,i	4		17	8
INSERTF128	y,x	1		3-6	1
INSERTF128	y,m128	1		7	1
VINSERTF32X4	Z,Z,X	1		3-6	1
VINSERTF32X4	z,z,m128	1		7	1
VINSERTF64X4	Z,Z,Y	1		3-6	1 1
VINSERTF64X4	z,z,m256	1		7	1 1
EXTRACTPS	r32,x,i	2		8	7
EXTRACTPS	m32,x,i	4		7	8
VEXTRACTF128	x,y,i	1		3-6	1
VEXTRACTF128	m128,y	4		7	8
VEXTRACTF32X4	X,Z	1		3-6	1
VEXTRACTF32X4	m128,z	4		7	8
VEXTRACTF64X4	y,z	1		3-6	1
VEXTRACTF64X4	m256,z	4		7	8
BLENDPS/PD	x,x/m,i	1	FP0/1	2	2
VBLENDPS/PD	V,V,V,i	1	FP0/1	2	0.5
(V)BLENDVPS/PD	V,V,V	2	110/1	7	7
BLENDVPS/PD	x,m,xmm0	3		,	8
VBLENDMPS/D	z{k},z,z	1	FP0/1	2	0.5
SHUFPS/D	Ζ{ κ },Ζ,Ζ Χ,Χ,İ	1	FP0	4	2
VSHUFPS/D		1	FP0	4-7	2
VSHUFF32X4	V,V,V,İ	1	FP0	4- <i>7</i> 4-7	2
VSHUFF64X2	Z,Z,Z,İ Z,Z,Z,İ	1	FP0	4-7 4-7	2
VPERMILPS/PD	v,v/m,i	1	FP0	3-6	1
VPERMILPS/PD		1	FP0	3-6	1
VECTIVILES/FD	v,v,v/m	ļ ,	reu	J 3-0	I

1	I	1	ı		I.
VPERM2F128	y,y,y/m,i	1	FP0	4-7	2
VPERMPS/PD	v,v,v/m	1	FP0	3-6	1
VPERMI2PS/PD	z,z,z/m	1	FP0	4-7	2
VCOMPRESSPS/D	z{k},z	1		3-6	3
VEXPANDPS/D	z{k},z	1		3-6	3
VEXI / II O/B	2(11),2				
Gather and scatter					
VPGATHERDPS	x,[r+s*x],x	6			12
VPGATHERDPS		6			12
	y,[r+s*y],y				
VPGATHERDPS	z,[r+s*z],z	1			11
VPGATHERQPS	x,[r+s*x],x	6			12
VPGATHERQPS	x,[r+s*y],x	6			12
VPGATHERQPS	y,[r+s*z],y	1			7
VPGATHERDPD	x,[r+s*x],x	6			12
VPGATHERDPD	y,[r+s*x],y	6			12
VPGATHERDPD	z,[r+s*y],z	1			7
VPGATHERQPD	x,[r+s*x],x	6			12
VPGATHERQPD	y,[r+s*y],y	6			12
VPGATHERQPD	z,[r+s*z],z	1			7
VGATHERPF0DPS	z,[r+s*z],z	1			~200
VGATHERPF0QPS	y,[r+s*z],y	1			~100
VGATHERI FOOT O	z,[r+s*y],z	1			~100
VGATHERPF0QPD		1			~100
	z,[r+s*z],z	-			
VGATHERPF1DPS	z,[r+s*z],z	1			~200
VGATHERPF1QPS	y,[r+s*z],y	1			~100
VGATHERPF1DPD	z,[r+s*y],z	1			~100
VGATHERPF1QPD	z,[r+s*z],z	1			~100
VPSCATTERDPS	z,[r+s*z],z	4			17
VPSCATTERQPS	y,[r+s*z],y	4			11
VPSCATTERDPD	z,[r+s*y],z	4			11
VPSCATTERQPD	z,[r+s*z],z	4			11
VSCATTERPF0DPS	z,[r+s*z],z	1			~200
VSCATTERPF0QPS	y,[r+s*z],y	1			~100
VSCATTERPF0DPD	z,[r+s*y],z	1			~100
VSCATTERPF0QPD	z,[r+s*z],z	1			~100
VSCATTERPF1DPS	z,[r+s*z],z	1			~200
VSCATTERPF1QPS	y,[r+s*z],y	1			~100
VSCATTERPF1DPD	z,[r+s*y],z	1			~100
VSCATTERPF1QPD	z,[r+s*z],z	1			~100
VSCATTERFFIGED	2,[1+5 2],2	ı			~100
Conversion					
(V)CVTSD2SS	x,x	1	FP0	2	1
(V)CVTSS2SD		1	FP0	2	1
(V)CVTPD2PS	X,X		1	7	7
` '	V,V	2	FP0		
(V)CVTPS2PD	V,V	2	FP0	7	7
VCVTPS2PH	V,V	2		7	7
VCVTPS2PH	m,v	5		_	9
VCVTPH2PS	V,V	2		7	7
VCVTPH2PS	v,m	3			8
(V)CVT(T)SS2(U)SI	r32/64,x	2	FP0	6	7
(V)CVT(U)SI2SS	x,r32/64	1	FP0	5	1
(V)CVT(T)SD2(U)SI	r32/64,x	2	FP0	6	7
(V)CVT(U)SI2SD	x,r32/64	1	FP0	5	1

0) (T(T) D00 D1	l				
CVT(T)PS2PI	mm,x	1	FP0	3	1 -
CVTPI2PS	x,mm	2	FP0	7	7
CVT(T) PD2PI	mm,x	2	FP0	7	7
CVTPI2PD	x,mm	2	FP0	7	7
(V)CVT(T) PS2DQ	V,V	1	FP0	2	1
(V)CVTDQ2PS	V,V	1	FP0	2	1
(V)CVT(T)PD2DQ	V,V	2	FP0	7	7
(V)CVTDQ2PD	V,V	2	FP0	7	7
VCVT(T)PS2UDQ	Z,Z	1	FP0	2	1
VCVTUDQ2PS	Z,Z	1	FP0	2	1
VCVT(T)PD2UDQ	Z,Z	2	FP0	7	7
VCVTUDQ2PD	z,z	2	FP0	7	7
Arithmetic					
ADDSS SUBSS	x,x	1	FP0/1	6	0.5
ADDSD SUBSD	x,x	1	FP0/1	6	0.5
ADDPS SUBPS	x,x	1	FP0/1	6	0.5
VADDPS VSUBPS		1	FP0/1	6	0.5
ADDPD SUBPD	V,V,V	1	FP0/1	6	
	X,X	-	1		0.5
VADDPD VSUBPD	V,V,V	1	FP0/1	6	0.5
ADDSUBPS/D	X,X	1	FP0/1	6	0.5
VADDSUBPS/D	V,V,V	1	FP0/1	6	0.5
HADDPS/D HSUBPS/D	X,X	3		15	8
VHADDPS/D VHSUBPS/D	уу,у,	3		15	8
MULSS/D	X,X	1	FP0/1	6	0.5
MULPS/D	X,X	1	FP0/1	6	0.5
VMULPS/D	V,V,V	1	FP0/1	6	0.5
DIVSS	X,X	3	FP0	27	17
DIVSD	x,x	3	FP0	42	42
DIVPS	x,x	18	FP0	32	20
VDIVPS	V,V,V	18	FP0	32	32
DIVPD	x,x	18	FP0	32	20
VDIVPD	V,V,V	18	FP0	32	32
RCPSS	X,X	1	FP0	7	2
(V)RCPPS	V,V	1	FP0	8	3
VRCP14SS	x,x,x	1	FP0	7	2
VRCP14PS	V,V	1	FP0	8	3
VRCP28SS	x,x,x	1	FP0	7	2
VRCP28PS	v,v	1	FP0	8	3
VRCP28SD	x,x,x	1	FP0	7	2
VRCP28PD	v,v	1	FP0	7	2
CMPccSS/D PS/D	x,x	1	FP0/1	2	0.5
VCMPccPS/D	k,z,z	1	FP0/1	2	0.5
COMISS/D UCOMISS/D	x,x	2	1 . 0, .	7	7
COMISS/D UCOMISS/D	x,m	3		'	8
MAXSS/D MINSS/D	x,x	1	FP0/1	2	0.5
MAXPS/D MINPS/D		1	FP0/1	2	0.5
VMAXPS/D VMINPS/D	X,X	1	FP0/1 FP0/1	2	0.5
	V,V,V	-	FF0/1		
ROUNDSS/D	x,x,i	1		6	2
(V)ROUNDPS/D	v,v,i	1		6	0.5
VRNDSCALESS/D	x,x,x,i	1		6	0.5
VRNDSCALEPS/D	v,v,i	1	ED0/4	6	0.5
VSCALEFSS/D	X,X,X	1	FP0/1	6	0.5

	i	1			1	
VSCALEFPS/D	Z,Z,Z	1	FP0/1	6	0.5	
DPPS	x,x,i	14		36	14	
VDPPS	y,y,y,i	14		36	13	
DPPD	x,x,i	12		24	13	
VFMADD (all FMA instr.)	V,V,V	1		6	0.5	
VI WADD (all I WA IIISti.)	, v, v, v				0.0	
Math						
SQRTSS	x,x	3	FP0	28	18	
SQRTPS		18	FP0	38	16	
	X,X					
VSQRTPS	V,V	18	FP0	38	16	
SQRTSD	X,X	30	FP0	43	35	
SQRTPD	X,X	18	FP0	37	16	
VSQRTPD	V,V	18	FP0	37	16	
RSQRTSS	x,x	1	FP0	7	2	
RSQRTPS	x,x	1	FP0	8	3	
VRSQRTPS	V,V	1	FP0	7	3	
VRSQRT14SS	v,v,v	1	FP0	7	2	
VRSQRT14PS	v,v	1	FP0	7	3	
VRSQRT28SS	V,V,V	1	FP0	7	2	
VRSQRT28PS		1	FP0	7	3	
	V,V				2	
VRSQRT28SD	V,V,V	1	FP0	7		
VRSQRT28PD	V,V	1	FP0	6	2	
VEXP2PS	V,V	2		10	7	
VEXP2PD	V,V	2		9	7	
VFIXUPIMMSS/D/PS/D	v,v,v,i	1	FP0	2	1	
VGETEXPSS/D	V,V,V	1	FP0/1	6	0.5	
VGETEXPPS/D	V,V	1	FP0/1	6	0.5	
VGETMANTSS/D	V,V,V	1	FP0/1	6	0.5	
VGETMANTPS/D	v,v	1	FP0/1	6	0.5	
VFIXUPIMMSS/SD/PS/PD	v,v,v	1	FP0	2	1	
VI IXOI IIVIIVIOGIODII GII D	,,,,,		'''	_		
Logic						
ANDPS/D ANDNPS/D	x,x	1	FP0/1	2	0.5	
ORPS/D XORPS/D	x,x	1	FP0/1	2	0.5	
VANDPS/D VANDNPS/D		1	FP0/1	2	0.5	
	V,V,V			1		
VORPS/D VXORPS/D	V,V,V	1	FP0/1	2	0.5	
Otto a se						
Other						
VZEROUPPER		11			30	32 bit mode
VZEROUPPER		19			36	64 bit mode
VZEROALL		11			30	32 bit mode
VZEROALL		19			36	64 bit mode
LDMXCSR	m32	6			21	
STMXCSR	m32	5			15	
FXSAVE	m	90			113	32 bit mode
FXSAVE	m	98			119	64 bit mode
FXRSTOR	m	98			122	32 bit mode
FXRSTOR	m	114			130	64 bit mode
FNSAVE				205	205	OT DIL HIDGE
	m m	135		1		
FRSTOR	m	78		191	191	00.1.11
XSAVE	m	251			396	32 bit mode
XSAVE	m	291			430	64 bit mode
XRSTOR	m	116			231	32 bit mode

XRSTOR	m	157	273	64 bit mode
XSAVEOPT	m	251	396	32 bit mode
XSAVEOPT	m	291	428	64 bit mode

Mask register instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
KMOVW	k,k	1	FP0/1	2	0.5	
KMOVW	k,m	1		7	0.5	
KMOVW	m,k	1		7	1	
KMOVW	k,r	1	FP0	5	1	
KMOVW	r,k	1		4	1	
KUNPCKBW	k,k,k	1	FP0/1	2	0.5	
VPBROADCASTMB2Q	v,k	1	FP0	6	1	
VPBROADCASTMW2D	v,k	1	FP0	6	1	
Arithmetic						
KSHIFTLW	k,k,i	1	FP0/1	2	0.5	
KSHIFTRW	k,k,i	1	FP0/1	2	0.5	
Logic						
KANDW KANDNW	k,k,k	1	FP0/1	2	0.5	
KORW KXORW KXNORW	k,k,k	1	FP0/1	2	0.5	
KNOTW	k,k	1	FP0/1	2	0.5	
KORTESTW	k,k	1	FP1	5	1	

VIA Nano 2000 series

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, xmm = 128 bit xmm

register, mm/x = mmx or xmm register, sr = segment register, m = memory,

m32 = 32-bit memory operand, etc.

μορs: The number of micro-operations from the decoder or ROM. Note that the VIA

Nano 2000 processor has no reliable performance monitor counter for μ ops. Therefore the number of μ ops cannot be determined except in simple cases.

Port: Tells which execution port or unit is used. Instructions that use the same port

cannot execute simultaneously.

Integer add, Boolean, shift, etc.Integer add, Boolean, move, jump.

I12: Can use either I1 or I2, whichever is vacant first.MA: Multiply, divide and square root on all operand types.MB: Various Integer and floating point SIMD operations.

MBfadd: Floating point addition subunit under MB.

SA: Memory store address.

ST: Memory store. LD: Memory load.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give a similar de-

lay.

Note: There is an additional latency for moving data from one unit or subunit to another. A table of these latencies is given in manual 3: "The microarchitecture of Intel, AMD and VIA CPUs". These additional latencies are not included in the listings below where the source and destination operands are of the same type.

Reciprocal throughput: The average number of clock cycles per instruction for a series of independent instructions of the same kind in the same thread.

Integer instructions

	Operands	µops	Port	Latency	Reciprocal thruoghput	Remarks
Move instructions						
MOV	r,r	1	12	1	1	
MOV	r,i	1	12	1	1	
						Latency 4 on
MOV	r,m	1	LD	2	1	pointer register
MOV	m,r	1	SA, ST	2	1.5	-
MOV	m,i	1	SA, ST		1.5	
MOV	r,sr				1	
MOV	m,sr				2	
MOV	sr,r			20	20	
MOV	sr,m			20	20	
MOVNTI	m,r		SA, ST	2	1.5	

1	ı	I	1		ı	1
MOVSX MOVSXD						
MOVZX	r,r	1	12	1	1	
MOVSX MOVSXD	r,m	2	LD, I2	3	1	
MOVZX	r,m	1	LD	2	1	
CMOVcc	r,r	2	I1, I2	2	1	
CMOVcc	r,m		LD, I1	5	2	
XCHG	r,r	3	12	3	3	
XCHG			12	20	20	Implicit lock
	r,m			6	20	IIIIpiicit lock
XLAT	m m		04.07	О	4.0	
PUSH	r		SA, ST		1-2	
PUSH	i		SA, ST		1-2	
PUSH	m		Ld, SA, ST		2	
PUSH	sr				17	
PUSHF(D/Q)				8	8	
PUSHA(D)					15	Not in x64 mode
POP	r		LD		1.25	110t III XO I IIIOGO
POP			LD			
	(E/R)SP				4	
POP	m				5	
POP	sr				20	
POPF(D/Q)				9	9	
POPA(D)					12	Not in x64 mode
LAHF		1	I1	1	1	
SAHF		1	l1	1	1	
SALC				9	6	Not in x64 mode
	r m	4	C A	1		
LEA	r,m	1	SA	ı	1	3 clock latency on
DEMAD	_	4	10	4	4	input register
BSWAP	r	1	12	1	1	
LDS LES LFS LGS LSS	m m			30	30	
PREFETCHNTA	m		LD		1-2	
PREFETCHT0/1/2	m		LD		1-2	
LFENCE					14	
MFENCE					14	
SFENCE					14	
0. 2.102						
Arithmetic instructions						
ADD SUB	r,r/i	1	l12	1	1/2	
ADD SUB		2	LD 112	· ·	1	
	r,m			_		
ADD SUB	m,r/i	3	LD I12 SA ST	5	2	
ADC SBB	r,r/i	1	I1	1	1	
ADC SBB	r,m	2	LD I1		1	
ADC SBB	m,r/i	3	LD I1 SA ST	5	2	
CMP	r,r/i	1	l12	1	1/2	
CMP	m,r/i	2	LD I12		1	
INC DEC NEG NOT	r	1	l12	1	1/2	
INC DEC NEG NOT	m .	3	LD I12 SA ST	5	172	
	'''	3	LD 112 SA S1	3	27	Not in v64 mode
AAA					37	Not in x64 mode
AAS					37	Not in x64 mode
DAA					22	Not in x64 mode
DAS					24	Not in x64 mode
AAD					23	Not in x64 mode
AAM					30	Not in x64 mode
						Extra latency to
MUL IMUL	r8		MA	7-9		other ports
I .	-	I	1	=	I	1

MUL IMUL	r16		MA	7-9		do.
MUL IMUL	r32		MA	7-9		do.
MUL IMUL	r64		MA	8-10		do.
IMUL	r16,r16		MA	4-6	1	do.
IMUL	r32,r32		MA	4-6	1	do.
IMUL	r64,r64		MA	5-7	2	do.
IMUL	r16,r16,i		MA	4-6	1	do.
IMUL	r32,r32,i		MA	4-6	1	do.
IMUL	r64,r64,i		MA	5-7	2	do.
DIV	r8		MA	26	26	do.
DIV	r16		MA	27-35	27-35	do.
DIV	r32		MA	25-41	25-41	do.
DIV	r64		MA	148-183	148-183	do.
IDIV	r8		MA	26	26	do.
IDIV	r16		MA	27-35	27-35	do.
IDIV	r32		MA	23-39	23-39	do.
IDIV	r64		MA	187-222	187-222	do.
CBW CWDE CDQE	104	1	III	107-222	107-222	do.
CWD CDQ CQO		1	I1	1	1	
Logic instructions						
AND OR XOR	r,r/i	1	l12	1	1/2	
AND OR XOR	r,m	2	LD 112		1	
AND OR XOR	m,r/i	3	LD I12 SA ST	5	2	
TEST	r,r/i	1	112	1	1/2	
TEST		2	LD 112		1/2	
SHR SHL SAR	m,r/i	1	11	4		
	r,i/cl			1	1	
ROR ROL	r,i/cl	1	l1	1	1	
RCR RCL	r,1	1	l1	1	1	
RCR RCL	r,i/cl		l1	28+3n	28+3n	
SHLD SHRD	r16,r16,i		l1	11	11	
SHLD SHRD	r32,r32,i		l1	7	7	
SHLD	r64,r64,i		l1	33	33	
SHRD	r64,r64,i		l1	43	43	
SHLD SHRD	r16,r16,cl		l1	11	11 _	
SHLD SHRD	r32,r32,cl		I1	7	7	
SHLD	r64,r64,cl		I1	33	33	
SHRD	r64,r64,cl		I1	43	43	
BT	r,r/i	1	I1	1	1	
BT	m,r		I1		8	
BT	m,i	2	I1		1	
BTR BTS BTC	r,r/i	2	I1	2	2	
BTR BTS BTC	m,r		I1	10	10	
BTR BTS BTC	m,i		I1	8	8	
BSF BSR	r,r		I1	3	2	
SETcc	r		I1	2	1	
SETcc	m				1	
CLC STC CMC			I1	3	3	
CLD STD				3	3	
Control transfer instruc	tions					
IMD	obort/n	4	10	2	3	8 if >2 jumps in 16
JMP	short/near	1	12	3	3	bytes block

l			I	l 	I	
JMP	far			58		Not in x64 mode
IMD			12	2	2	8 if >2 jumps in 16
JMP	r (m = = m)		IZ	3	3 3	bytes block
JMP	m(near)			3	3	do.
JMP	m(far)			55		
Conditional jump	short/near			1-3-8	1-3-8	1 if not jumping.
						3 if jumping.
						8 if >2 jumps in 16 bytes block
J(E/R)CXZ	short			1-3-8	1-3-8	do.
LOOP	short			1-3-8	1-3-8	do.
LOOP(N)E	short			25	25	uo.
LOOF (N)L	SHOIL			23	25	0 if > 0 iumno in 16
CALL	near			3	3	8 if >2 jumps in 16 bytes block
CALL	far			72	72	Not in x64 mode
OALL	lai			12	12	8 if >2 jumps in 16
CALL	r			3	3	bytes block
CALL	m(near)			4	3	do.
CALL	m(far)			72	72	do.
CALL	ili(iai)			12	12	9 if >2 iumpe in 16
RETN				3	3	8 if >2 jumps in 16 bytes block
RETN	i			3	3	do.
RETF	·			39	39	do.
RETF	i			39	39	
BOUND	· ·			39	13	Not in x64 mode
INTO	r,m				7	Not in x64 mode
INTO					/	Not in xo4 mode
Ctuin a in atmosti and						
String instructions	1				_	
LODSB/W/D/Q					1	
REP LODSB/W/D/Q					3n+22	
STOSB/W/D/Q					1-2	
REP STOSB/W/D/Q					Small: 2n+2, Big: 6 bytes	
					per clock	
MOVSB/W/D/Q					2 per clock	
REP MOVSB/W/D/Q					Small: 2n+45,	
REF MOVSB/W/D/Q					Big: 6 bytes	
					per clock	
					por order	
SCASB/W/D/Q					1	
REP SCASB					2.2n	
REP SCASW/D/Q					Small: 2n+50	
112. 33/13/17/27					Big: 5 bytes	
					per clock	
CMPSB/W/D/Q					6	
REP CMPSB/W/D/Q					2.4n+24	
Other						
NOP (90)	1	1	All		1	Blocks all ports
Long NOP (0F 1F)		1	l12		1/2	
PAUSE					25	
ENTER	a,0				23	
ENTER	a,b				52+5b	
LEAVE				4	4	
CPUID				53-173		
15. 5.5	I		I	1 33 1.3	I	1 1

RDTSC			39	
RDPMC		40	40	

Floating point x87 instructions

Floating point x87 ii			Dont and	Lotonor	Dooinroos	Domorko
	Operands	µops	Port and Unit	Latency	Reciprocal thruoghput	Remarks
Move instructions						
FLD	r	1	MB	1	1	
FLD	m32/m64	2	LD MB	4	1	
FLD	m80	2	LD MB	4	1	
FBLD	m80			54	54	
FST(P)	r	1	MB	1	1	
FST(P)	m32/m64	3	MB SA ST	5	1-2	
FSTP	m80	3	MB SA ST	5	1-2	
FBSTP	m80			125	125	
FXCH	r	1	12	0	1	
FILD	m16	'	12	7		
FILD	m32			5		
FILD	m64			5		
FIST(T)(P)	m16			6		
	m32					
FIST(T)(P)				5 5		
FIST(T)(P)	m64		MD	5	_	
FLDZ FLD1		1	MB		1	
FLDPI FLDL2E etc.					10	
FCMOVcc	r			2	2	
FNSTSW	AX				5	
FNSTSW	m16				3	
FLDCW	m16			13	13	
FNSTCW	m16				2	
FINCSTP FDECSTP		1	12	0	1	
FFREE(P)		1	MB		1	
FNSAVE	m			321	321	
FRSTOR	m			195	195	
Arithmetic instructions		-				
FADD(P) FSUB(R)(P)	r/m	1	MB	2	1	Lower precision:
FMUL(P)	r/m	1	MA	4	1 2	Lat: 4, Thr: 2
, ,		'				
FDIV(R)(P)	r/m		MA	15-42	15-42	
FABS		1	MB	1	1	
FCHS	,	1	MB	1	1	
FCOM(P) FUCOM	r/m	1	MB		1	
FCOMPP FUCOMPP		1	MB		1	
FCOMI(P) FUCOMI(P)	r	1	MB		1	
FIADD FISUB(R)	m		MB		2	
FIMUL	m				4	
FIDIV(R)	m				42	
FICOM(P)	m	1			2	
FTST		1	MB		1	
FXAM					41	
FPREM				151-171		
FPREM1				106-155		

FRNDINT			29		
Math					
FSCALE			39		
FXTRACT			36-57		
FSQRT			73		
FSIN FCOS			51-159		
FSINCOS			270-360		
F2XM1			50-200		
FYL2X			~60		
FYL2XP1			~170		
FPTAN			300-370		
FPATAN			~170		
Other					
FNOP	1	MB		1	
WAIT	1	l12	0	1/2	
FNCLEX				57	
FNINIT				85	

Integer MMX and XMM instructions

	Operands	µops	Port and Unit	Latency	Reciprocal thruoghput	Remarks
Move instructions			01110			
MOVD	r32/64,mm/x	1		3	1 1	
MOVD	m32/64,mm/x	1	SA ST	2-3	1-2	
MOVD	mm/x,r32/64			4	1 1	
MOVD	mm/x,m32/64	1	LD	2-3	1 1	
MOVQ	mm/x, mm/x	1	MB	1	1 1	
MOVQ	mm/x,m64	1	LD	2-3	1 1	
MOVQ	m64, mm/x	1	SA ST	2-3	1-2	
MOVDQA	xmm, xmm	1	MB	1	1 1	
MOVDQA	xmm, m128	1	LD	2-3	1 1	
MOVDQA	m128, xmm	1	SA ST	2-3	1-2	
MOVDQU	m128, xmm	1	SA ST	2-3	1-2	
MOVDQU	xmm, m128	1	LD	2-3	1 1	
LDDQU	xmm, m128	1	LD	2-3	1 1	
MOVDQ2Q	mm, xmm	1	MB	1	1 1	
MOVQ2DQ	xmm,mm	1	MB	1	1 1	
MOVNTQ	m64,mm	3		~300	2	
MOVNTDQ	m128,xmm	3		~300	2	
PACKSSWB/DW PACKUSWB PUNPCKH/LBW/WD/	V,V	1	MB	1	1	
DQ	V,V	1	MB	1	1 1	
PUNPCKH/LQDQ	V,V	1	MB	1	1 1	
PSHUFB	v,v	1	MB	1	1 1	
PSHUFW	mm,mm,i	1	MB	1	1 1	
PSHUFL/HW	x,x,i	1	MB	1	1 1	
PSHUFD	x,x,i	1	MB	1	1 1	
PALIGNR	x,x,i	1	MB	1	1 1	
MASKMOVQ	mm,mm				1-3	
MASKMOVDQU	xmm,xmm				1-3	

PMOVMSKB	r32,mm/x			3	1	
PEXTRW	r32 ,mm/x,i			3	1	
PINSRW	mm/x,r32,i			9	9	
	,				_	
Arithmetic instructions	I					
PADD/SUB(U)(S)B/W/D	V,V	1	MB	1	1	
PADDQ PSUBQ	V,V	1	MB	1	1	
PHADD(S)W						
PHSUB(S)W	V,V	3	MB	3	3	
PHADDD PHSUBD	V,V	3	MB	3	3	
PCMPEQ/GTB/W/D	V,V	1	MB	1	1	
PMULL/HW PMULHUW	V,V	1	MA	3	1	
PMULHRSW	V,V	1	MA	3	1	
PMULUDQ	V,V	1	MA	3	1	
PMADDWD	V,V			4	2	
PMADDUBSW	V,V			10	8	
PSADBW	V,V		MB	2	1	
PAVGB/W	V,V	1	MB	1	1	
PMIN/MAXUB	V,V	1	MB	1	1	
PMIN/MAXSW	V,V	1	MB	1	1	
PABSB PABSW PABSD	,					
	V,V	1	MB	1	1	
PSIGNB PSIGNW						
PSIGND	V,V	1	MB	1	1	
Logic instructions						
PAND(N) POR PXOR	V,V	1	MB	1	1	
PSLL/RL/RAW/D/Q	V,V	1	MB	1	1	
PSLL/RL/RAW/D/Q	v,i	1	MB	1	1	
PSLL/RLDQ	x,i	1	MB	1	1	
Other						
EMMS		1	MB		1	

Floating point XMM instructions

	Operands	µops	Port and Unit	Latency	Reciprocal thruoghput	Remarks
Move instructions						
MOVAPS/D	xmm,xmm	1	MB	1	1	
MOVAPS/D	xmm,m128	1	LD	2-3	1	
MOVAPS/D	m128,xmm	1	SA ST	2-3	1-2	
MOVUPS/D	xmm,m128	1	LD	2-3	1	
MOVUPS/D	m128,xmm	1	SA ST	2-3	1-2	
MOVSS/D	xmm,xmm	1	MB	1	1	
MOVSS/D	x,m32/64	1	LD	2-3	1	
MOVSS/D	m32/64,x	1	SA ST	2-3	1-2	
MOVHPS/D	xmm,m64			6	1	
MOVLPS/D	xmm,m64			6	1	
MOVHPS/D	m64,xmm			6	1-2	
MOVLPS/D	m64,xmm			2	1-2	
MOVLHPS MOVHLPS	xmm,xmm	1	MB	1	1	
MOVMSKPS/D	r32,xmm			3	1	
MOVNTPS/D	m128,xmm			~300	2.5	

SHUFPS	xmm,xmm,i	1	MB	1	1 1
SHUFPD	xmm,xmm,i	1	MB	1	1 1
MOVDDUP	xmm,xmm	1	MB	1	1 1
MOVSH/LDUP	xmm,xmm	1	MB	1	1
UNPCKH/LPS	xmm,xmm	1	MB	1	1
UNPCKH/LPD	xmm,xmm	1	MB	1	1 1
	7	•	2		
Conversion					
CVTPD2PS	xmm,xmm			3-4	
CVTSD2SS	xmm,xmm			15	
CVTPS2PD	xmm,xmm			3-4	
CVTSS2SD	xmm,xmm			15	
CVTDQ2PS	xmm,xmm			3	
CVT(T) PS2DQ	xmm,xmm			2	
CVTDQ2PD	xmm,xmm			4	
CVTDQ2FD CVT(T)PD2DQ	xmm,xmm			3	
CVT(1)FD2DQ CVTPI2PS	· ·			4	
CVT(T)PS2PI	xmm,mm			3	
CVT(1)F32F1 CVTPI2PD	mm,xmm			4	
	xmm,mm			3	
CVT(T) PD2PI	mm,xmm				
CVTSI2SS	xmm,r32			5	
CVT(T)SS2SI	r32,xmm			4	
CVTSI2SD	xmm,r32			5	
CVT(T)SD2SI	r32,xmm			4	
A .:!41 41					
Arithmetic	vmm vmm	1	MDfodd	2.2	1 1
ADDSS SUBSS	xmm,xmm	1	MBfadd MBfadd	2-3 2-3	1 1
ADDSD SUBSD ADDPS SUBPS	xmm,xmm	1		2-3	1 1
	xmm,xmm	1	MBfadd		1 1
ADDPD SUBPD	xmm,xmm	-	MBfadd	2-3	· .
ADDSUBPS	xmm,xmm	1 1	MBfadd	2-3	1 1
ADDSUBPD	xmm,xmm	ı	MBfadd	2-3	1
HADDPS HSUBPS	xmm,xmm		MBfadd	5	3
HADDPD HSUBPD	xmm,xmm	4	MBfadd	5	3
MULSS	xmm,xmm	1	MA	3	1
MULSD	xmm,xmm	1	MA	4	2
MULPS	xmm,xmm		MA	3	1
MULPD	xmm,xmm		MA	4	2
DIVSS	xmm,xmm		MA	15-22	15-22
DIVSD	xmm,xmm		MA	15-36	15-36
DIVPS	xmm,xmm		MA	42-82	42-82
DIVPD	xmm,xmm		MA	24-70	24-70
RCPSS	xmm,xmm			5	5
RCPPS	xmm,xmm			14	11
CMPccSS/D	xmm,xmm	1	MBfadd	2	1
CMPccPS/D	xmm,xmm	1	MBfadd	2	1
COMISS/D UCOMISS/D					
MAN/00/D MINIOC/D	xmm,xmm	_	NADE ::	3	1 1
MAXSS/D MINSS/D	xmm,xmm	1	MBfadd	2	1 1
MAXPS/D MINPS/D	xmm,xmm	1	MBfadd	2	1
Moth					
Math SQRTSS	vmm vmm		MA	33	33
OWITIOO	xmm,xmm		IVIA	33	33

SQRTPS	xmm,xmm		MA	126	126	
SQRTSD	xmm,xmm		MA	62	62	
SQRTPD	xmm,xmm		MA	122	122	
RSQRTSS	xmm,xmm			5	5	
RSQRTPS	xmm,xmm			14	11	
Logic						
ANDPS/D	xmm,xmm	1	MB	1	1	
ANDNPS/D	xmm,xmm	1	MB	1	1	
ORPS/D	xmm,xmm	1	MB	1	1	
XORPS/D	xmm,xmm	1	MB	1	1	
Other						
LDMXCSR	m32			45	29	
STMXCSR	m32			13	13	
FXSAVE	m4096			208	208	
FXRSTOR	m4096			232	232	

VIA-specific instructions

Instruction	Conditions	Clock cycles, approximately
XSTORE	Data available	160-400 clock giving 8 bytes
XSTORE	No data available	50-80 clock giving 0 bytes
REP XSTORE	Quality factor = 0	4800 clock per 8 bytes
REP XSTORE	Quality factor > 0	19200 clock per 8 bytes
REP XCRYPTECB	128 bits key	44 clock per 16 bytes
REP XCRYPTECB	192 bits key	46 clock per 16 bytes
REP XCRYPTECB	256 bits key	48 clock per 16 bytes
REP XCRYPTCBC	128 bits key	54 clock per 16 bytes
REP XCRYPTCBC	192 bits key	59 clock per 16 bytes
REP XCRYPTCBC	256 bits key	63 clock per 16 bytes
REP XCRYPTCTR	128 bits key	43 clock per 16 bytes
REP XCRYPTCTR	192 bits key	46 clock per 16 bytes
REP XCRYPTCTR	256 bits key	48 clock per 16 bytes
REP XCRYPTCFB	128 bits key	54 clock per 16 bytes
REP XCRYPTCFB	192 bits key	59 clock per 16 bytes
REP XCRYPTCFB	256 bits key	63 clock per 16 bytes
REP XCRYPTOFB	128 bits key	54 clock per 16 bytes
REP XCRYPTOFB	192 bits key	59 clock per 16 bytes
REP XCRYPTOFB	256 bits key	63 clock per 16 bytes
REP XSHA1		3 clock per byte
REP XSHA256		4 clock per byte

VIA Nano 3000 series

List of instruction timings and µop breakdown

Explanation of column headings:

Operands: i = immediate data, r = register, mm = 64 bit mmx register, xmm = 128 bit xmm

register, mm/x = mmx or xmm register, sr = segment register, m = memory,

m32 = 32-bit memory operand, etc.

μορs: The number of micro-operations from the decoder or ROM. Note that the VIA

Nano 3000 processor has no reliable performance monitor counter for μ ops. Therefore the number of μ ops cannot be determined except in simple cases.

Port: Tells which execution port or unit is used. Instructions that use the same port

cannot execute simultaneously.

Integer add, Boolean, shift, etc.Integer add, Boolean, move, jump.

I12: Can use either I1 or I2, whichever is vacant first.MA: Multiply, divide and square root on all operand types.MB: Various Integer and floating point SIMD operations.

MBfadd: Floating point addition subunit under MB.

SA: Memory store address.

ST: Memory store.
LD: Memory load.

Latency: This is the delay that the instruction generates in a dependency chain. The

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give a similar de-

lav.

Note: There is an additional latency for moving data from one unit or subunit to another. A table of these latencies is given in manual 3: "The microarchitecture of Intel, AMD and VIA CPUs". These additional latencies are not included in the listings below where the source and destination operands are of the same

type.

Reciprocal throughput: The average number of clock cycles per instruction for a series of independent instructions of the same kind in the same thread.

Integer instructions

	Operands	µops	Port	Latency	Reciprocal thruogh- put	Remarks
Move instructions						
MOV	r,r	1	12	1	1	
MOV	r,i	1	l12	1	1/2	
						Latency 4 on pointer
MOV	r,m	1	LD	2	1	register
MOV	m,r	1	SA, ST	2	1.5	
MOV	m,i	1	SA, ST		1.5	
MOV	r,sr		l12		1/2	
MOV	m,sr				1.5	
MOV	sr,r			20	20	
MOV	sr,m			20	20	
MOVNTI	m,r		SA, ST	2	1.5	

MOVSX MOVZX	rr	1	l12	1	1/2	
MOVSX	r,r r64,r32	1	112	1	1/2	
	· ·		1 D 142			
MOVSX MOVSXD	r,m	2	LD, I12	3	1	
MOVZX	r,m	1	LD	2	1	
CMOVcc	r,r	1	I12	1	1/2	
CMOVcc	r,m		LD, I12	5	1	
XCHG	r,r	3	l12	3	1.5	
XCHG	r,m			18	18	Implicit lock
XLAT	m	3	LD, I1	6	2	
PUSH	r	1	SA, ST		1-2	
PUSH	i	1	SA, ST		1-2	
PUSH	m .		LD, SA, ST		2	
PUSH	sr		LB, 6/1, 61		6	
	31	3		2	2	
PUSHF(D/Q)						Nighting of August als
PUSHA(D)		9			15	Not in x64 mode
POP	r	2	LD		1.25	
POP	(E/R)SP				4	
POP	m	3			2	
POP	sr				11	
POPF(D/Q)		3			1	
POPA(D)		16			12	Not in x64 mode
LAHF		1	l1	1	1	
SAHF		1	l1	1	1	
SALC		2		10	6	Not in x64 mode
57 KES		_		10	J	Extra latency to other
LEA	r,m	1	SA	1	1	ports
BSWAP	r ',,'''		12	1	1	ports
		12	12	_		
LDS LES LFS LGS LSS	m			28	28	
PREFETCHNTA	m	1	LD		1	
PREFETCHT0/1/2	m	1	LD		1	
LFENCE MFENCE					4-	
SFENCE					15	
A -141						
Arithmetic instructions	<i>I</i> :		140		4.00	
ADD SUB	r,r/i	1	I12	1	1/2	
ADD SUB	r,m	2	LD I12		1	
ADD SUB	m,r/i	3	LD I12 SA ST	5	2	
ADC SBB	r,r/i	1	l1	1	1	
ADC SBB	r,m	2	LD I1		1	
ADC SBB	m,r/i	3	LD I1 SA ST	5	2	
CMP	r,r/i	1	l12	1	1/2	
CMP	m,r/i	2	LD I12		1	
INC DEC NEG NOT	r	1	l12	1	1/2	
INC DEC NEG NOT	m .	3	LD I12 SA ST	5		
AAA		12	LD 112 O/(O)		37	Not in x64 mode
I .		12				Not in x64 mode
AAS					22	
DAA		14			22	Not in x64 mode
DAS		14			24	Not in x64 mode
AAD		7			24	Not in x64 mode
AAM		13			31	Not in x64 mode
MUL IMUL	r8	1	12	2		
MUL IMUL	r16	3	12	3		
MUL IMUL	r32	3	12	3		
1			· -	-	l	ı I

MUL IMUL r64 3 MA 8 8 Extra latency to ports IMUL r16,r16 1 I2 2 1 Extra latency to ports IMUL r32,r32 1 I2 2 1 Extra latency to ports IMUL r16,r16,i 1 I2 2 1 Extra latency to ports IMUL r32,r32,i 1 I2 2 1 Extra latency to ports IMUL r64,r64,i 1 MA 5 2 Extra latency to ports IMUL r64,r64,i 1 MA 5 2 Extra latency to ports DIV r8 MA 22-24 22-24 Extra latency to ports DIV r32 MA 24-28 24-28 24-28 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	other
IMUL r16,r16 1 I2 2 1 IMUL r32,r32 1 I2 2 1 IMUL r64,r64 1 MA 5 2 ports IMUL r16,r16,i 1 I2 2 1 Extra latency to ports IMUL r32,r32,i 1 I2 2 1 Extra latency to ports IMUL r64,r64,i 1 MA 5 2 ports IMUL r8 MA 22-24 22-24 Extra latency to ports IMA 24-28 24-28 24-28 24-28 24-28 DIV r32 MA 22-30 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	
IMUL r32,r32 1 I2 2 1 Extra latency to ports IMUL r64,r64 1 MA 5 2 ports IMUL r16,r16,i 1 I2 2 1 Extra latency to ports IMUL r32,r32,i 1 I2 2 1 Extra latency to ports IMUL r64,r64,i 1 MA 5 2 ports DIV r8 MA 22-24 22-24 ports DIV r16 MA 24-28 24-28 DIV r32 MA 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	
IMUL	
IMUL r64,r64 1 MA 5 2 ports IMUL r16,r16,i 1 I2 2 1 IMUL r32,r32,i 1 I2 2 1 IMUL r64,r64,i 1 MA 5 2 Extra latency to ports DIV r8 MA 22-24 22-24 Dovents DIV r16 MA 24-28 24-28 DIV r32 MA 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	
IMUL r16,r16,i 1 I2 2 1 IMUL r32,r32,i 1 I2 2 1 IMUL r64,r64,i 1 MA 5 2 DIV r8 MA 22-24 22-24 DIV r16 MA 24-28 24-28 DIV r32 MA 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	other
IMUL r32,r32,i 1 I2 2 1 Extra latency to ports IMUL r64,r64,i 1 MA 5 2 Extra latency to ports DIV r8 MA 22-24 22-24 22-24 DIV r16 MA 24-28 24-28 DIV r32 MA 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	other
MUL r64,r64,i 1 MA 5 2 Extra latency to ports	other
IMUL r64,r64,i 1 MA 5 2 ports DIV r8 MA 22-24 22-24 DIV r16 MA 24-28 24-28 DIV r32 MA 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	other
DIV r8 MA 22-24 22-24 DIV r16 MA 24-28 24-28 DIV r32 MA 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	
DIV r16 MA 24-28 24-28 DIV r32 MA 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	
DIV r32 MA 22-30 22-30 DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	
DIV r64 MA 145-162 145-162 IDIV r8 MA 21-24 21-24	
IDIV r8 MA 21-24 21-24	
IDIV r16 MA 24-28 24-28	
IDIV r32 MA 18-26 18-26	
IDIV r64 MA 182-200 182-200	
CBW CWDE CDQE 1 12 1 1	
CWD CDQ CQO 1 1 12 1 1	
OWD ODG OGO	
Logic instructions	
AND OR XOR r,r/i 1 112 1 1/2	
AND OR XOR m,r/i 3 LD 112 SA ST 5 2	
TEST r,r/i 1 112 1 1/2	
TEST m,r/i 2 LD I12 1	
SHR SHL SAR r,i/cl 1 I12 1 1/2	
ROR ROL r,i/cl 1 1 1 1	
RCR RCL r,1 1 1 1 1	
RCR RCL r,i/cl 5+2n I1 28+3n 28+3n	
SHLD SHRD r16,r16,i/cl 2 11 2 2	
SHLD SHRD r32,r32,i/cl 2 11 2 2	
SHLD r64,r64,i/cl 16 I1 32 32	
SHRD r64,r64,i/cl 23 I1 42 42	
BT r,r/i 1 I1 1 1	
BT m,r 6 I1 8	
BT m,i 2 I1 1	
BTR BTS BTC	
BTR BTS BTC m,r 8 I1 10 10	
BTR BTS BTC m,i 5 11 8 8	
BSF BSR	
SETcc	
SETCC m 2	
CLD STD 3 I1 3 3	
Control transfer instructions	
8 if >2 jumps in	16
JMP short/near 1 I2 3 bytes block	
JMP far 14 50 Not in x64 mg	de
8 if >2 jumps in	16
JMP r 2 l2 3 bytes block	
JMP m(near) 2 3 3 do.	

JMP	m(far)	17			42	
						1 if not jumping. 3 if jumping. 8 if >2 jumps in 16
Conditional jump	short/near	1	12	1-3-8	1-3-8	bytes block
J(E/R)CXZ LOOP	short	2		1-3-8	1-3-8	
LOOP LOOP(N)E	short short	2 5		1-3-8 24	1-3-8 24	
LOOI (IV)L	SHOIL	3		24	27	8 if >2 jumps in 16
CALL	near	2		3	3	bytes block
CALL	far	17			58	Not in x64 mode
CALL	r	2		3	3	8 if >2 jumps in 16 bytes block
CALL	m(near)	3		4	3	do.
CALL	m(far)	19			54	
RETN		3		2	2	8 if >2 jumps in 16
RETN	i	4		3 3	3 3	bytes block do.
RETF		20		Ŭ	49	uo.
RETF	i	20			49	
BOUND	r,m	9			13	Not in x64 mode
INTO		3			7	Not in x64 mode
String instructions						
LODSB/W/D/Q		2			1	
REP LODSB/W/D/Q		3n			3n+27	
STOSB/W/D/Q		1			1-2	
					Small: n+40, Big:	
					6-7 bytes/	
REP STOSB/W/D/Q		0			clk	
MOVSB/W/D/Q		3			2 Small:	
					2n+20,	
DED MOVIODAMIDIO					Big: 6-7	
REP MOVSB/W/D/Q SCASB/W/D/Q		3			bytes/clk 1	
REP SCASB		3			2.4n	
					Small:	
					2n+31,	
REP SCASW/D/Q					Big: 5 bytes/clk	
CMPSB/W/D/Q		5			6	
REP CMPSB/W/D/Q					2.2n+30	
Othor						
Other NOP (90)		0-1	l12	0	1/2	Sometimes fused
long NOP (0F 1F)		0-1	l12	0	1/2	
PAUSE		2			6	
ENTER	a,0	10			21	
ENTER LEAVE	a,b	3		2	52+5b 2	
CPUID		3		∠ 55-146		
RDTSC					37	
RDPMC					40	

Floating point x87 instructions

	Operands	μops	Port	Latency	Reciprocal thruogh-	Remarks
					put	
Move instructions						
LD	r	1	MB	1	1 1	
ELD	m32/m64	2	LD MB	4	1 1	
ELD	m80	2	LD MB	4	1 1	
BLD	m80	36		54	54	
ST(P)	r	1	MB	1	1 1	
ST(P)	m32/m64	3	MB SA ST	5	1-2	
STP	m80	3	MB SA ST	5	1-2	
BSTP	m80	80		125	125	
XCH	r	1	12	0	1 1	
FILD	m16	3		7		
FILD	m32	2		5		
FILD	m64	2		5		
FIST(T)(P)	m16	3		6		
FIST(T)(P)	m32	3		5		
FIST(T)(P)	m64	3		5		
FLDZ FLD1	11104	1	MB		1 1	
FLDPI FLDL2E etc.		3	IVID		10	
FCMOVcc	r	1	MB	2	2	
NSTSW	AX		IVID		1	
NSTSW	m16	3			2	
LDCW	m16	5			8	
NSTCW	m16	3			2	
FINCSTP FDECSTP	11116	1	12	0	1	
			MB	U	1 1	
FREE(P) FNSAVE		122	IVID	319	319	
RSTOR	m	115			196	
-KSTUK	m	115		196	196	
Arithmetic instructions	I					
FADD(P) FSUB(R)(P)	r/m	1	MB	2	1	
FMUL(P)	r/m	1	MA	4	2	
FDIV(R)(P)	r/m		MA	14-23	14-23	
FABS		1	MB	1	1	
CHS		1	MB	1	1	
FCOM(P) FUCOM	r/m	1	MB		1	
COMPP FUCOMPP		1	MB		1 1	
FCOMI(P) FUCOMI(P)	r	1	MB	2	1 1	
FIADD FISUB(R)	m	3	MB		2	
FIMUL	m	3			4	
FIDIV(R)	m	3			16	
FICOM(P)	m	3			2	
TST		1	MB	2	1 1	
XAM		15		38	38	
PREM				~130		
PREM1				~130		
FRNDINT		11		27		

Math					
FSCALE	22		37		
FXTRACT	13		57		
FSQRT FSIN FCOS FSINCOS F2XM1 FYL2X FYL2XP1 FPTAN FPATAN			73 ~150 270-360 50-200 ~50 ~50 300-370 ~180		Less at lower precision
Other					
FNOP	1	MB		1	
WAIT	1	l12	0	1/2	
FNCLEX				59	
FNINIT				84	

Integer MMX and XMM instructions

	Operands	µops	Port	Latency	Reciprocal thruogh- put	Remarks
Move instructions						
MOVD	r,mm/x	1	MB	3	1	
MOVD	m,mm/x	1	SA ST	2	1-2	
MOVD	mm/x,r	1	12	4	1	
MOVD	mm/x,m	1	LD	2	1	
MOVQ	V,V	1	MB	1	1	
MOVQ	mm/x,m64	1	LD	2	1	
MOVQ	m64, mm/x	1	SA ST	2	1-2	
MOVDQA	x,x	1	MB	1	1	
MOVDQA	x, m128	1	LD	2	1	
MOVDQA	m128, x	1	SA ST	2	1-2	
MOVDQU	m128, x	1	SA ST	2	1-2	
MOVDQU	x, m128	1	LD	2	1	
LDDQU	x, m128	1	LD	2	1	
MOVDQ2Q	mm, x	1	MB	1	1 1	
MOVQ2DQ	x,mm	1	MB	1	1 1	
MOVNTQ	m64,mm	2		~360	2	
MOVNTDQ	m128,x	2		~360	2	
MOVNTDQA	x,m128	1		2	1 1	
PACKSSWB/DW						
PACKUSWB	V,V	1	MB	1	1 1	
PACKUSDW	x,x	1	MB	1	1 1	
PUNPCKH/LBW/WD/DQ	V,V	1	MB	1	1	
PUNPCKH/LQDQ	V,V	1	MB	1	1 1	
PSHUFB	V,V	1	MB	1	1 1	
PSHUFW	mm,mm,i	1	MB	1	1	
PSHUFL/HW	x,x,i	1	MB	1	1	
PSHUFD	x,x,i	1	MB	1	1	
PBLENDVB	x,x,xmm0	1	MB	2	2	

PBLENDW	x,x,i	1	MB	1	1	1
PALIGNR	x,x,i x,x,i	1	MB	1	1	
MASKMOVQ		'	IVID	ı	1-2	
MASKMOVDQU	mm,mm				1-2	
PMOVMSKB	X,X			2		
	r32,mm/x	4	MD	3	1	
PEXTRW	r32 ,mm/x,i	1	MB	3	1	
PEXTRB/D/Q	r32/64,x,i	1	MB	3	1	
PINSRW	mm/x,r32,i	2	MB	5	1	
PINSRB/D/Q	x,r32/64,i	2	MB	5	1	
PMOVSX/ZXBW/BD/			MD	4	4	
BQ/WD/WQ/DQ	X,X	1	MB	1	1	
Arithmetic instructions						
PADD/SUB(U)(S)B/W/D	V,V	1	MB	1	1	
PADDQ PSUBQ	V,V	1	MB	1	1	
PHADD(S)W	V, V		IVID	Į.	'	
PHSUB(S)W	V,V	3	MB	3	3	
PHADDD PHSUBD	V,V	3	MB	3	3	
PCMPEQ/GTB/W/D	V,V V,V	1	MB	1	1	
PCMPEQQ	x,x	1	MB	1	1	
PMULL/HW PMULHUW	V,V	1	MA	3	1	
PMULHRSW		1	MA	3	1	
PMULLD	V,V	1	MA	3	1	
	X,X	1	MA		1	
PMULUDQ	V,V	-		3	· •	
PMULDQ	X,X	1	MA	3	1	
PMADDWD	V,V	1	MA	4	2	
PMADDUBSW	V,V	7	145	10	8	
PSADBW	V,V	1	MB	2	1	
MPSADBW	x,x,i	1	MB	2	1	
PAVGB/W	V,V	1	MB	1	1	
PMIN/MAXSW	V,V	1	MB	1	1	
PMIN/MAXUB	V,V	1	MB	1	1	
PMIN/MAXSB/D	x,x	1	MB	1	1	
PMIN/MAXUW/D	x,x	1	MB	1	1	
PHMINPOSUW	x,x	1	MB	2	1	
PABSB PABSW PABSD						
	V,V	1	MB	1	1	
PSIGNB PSIGNW				_	_	
PSIGND	V,V	1	MB	1	1	
Logic instructions						
PAND(N) POR PXOR	V V	1	MB	1	1	
` '	V,V			3		
PTEST	V,V	1	MB		1	
PSLL/RL/RAW/D/Q	V,V	1	MB MB	1	1	
PSLL/RL/RAW/D/Q	(x)xmm,i	1	MB	1	1	
PSLL/RLDQ	x,i	1	MB	1	1	
Other						
EMMS		1	MB		1	
LIVIIVIO		ı	טועו	<u> </u>	1	

Floating point XMM instructions

	Operands	µops	Port	Latency	Reciprocal thruogh- put	Remarks
Move instructions						
MOVAPS/D	x,x	1	MB	1	1	
MOVAPS/D	x,m128	1	LD	2	1	
MOVAPS/D	m128,x	1	SA ST	2	1	
MOVUPS/D	x,m128	1	LD	2	1 1	
MOVUPS/D	m128,x	2	SA ST	2	1 1	
MOVSS/D	x,x	1	MB	1	1 1	
MOVSS/D	x,m32/64	1	LD	2-3	1 1	
MOVSS/D	m32/64,x	2	SA ST	2-3	1-2	
MOVHPS/D	x,m64	2		6	1	
MOVLPS/D	x,m64	2		6	1 1	
MOVHPS/D	m64,x	3		6	1-2	
MOVLPS/D	m64,x	1		2	1-2	
MOVLHPS MOVHLPS	X,X	1		1	1	
MOVMSKPS/D	r32,x	•		3	1 1	
MOVNTPS/D	m128,x	2		~360	1-2	
SHUFPS	x,x,i	1	MB	1	1 1	
SHUFPD	x,x,i	1	MB	1	1 1	
MOVDDUP	X,X,I X,X	1	MB	1	1 1	
MOVSH/LDUP	X,X X,X	1	MB	1	1 1	
UNPCKH/LPS	X,X X,X		MB	1	1 1	
UNPCKH/LPD	X,X X,X		MB	1	1 1	
ONF ON I/LF D	^,^	'	IVID		' I	
Conversion						
CVTPD2PS	x,x	2		5	2	
CVTSD2SS	x,x	1		2		
CVTPS2PD	x,x	2		5	1	
CVTSS2SD	x,x	1		2		
CVTDQ2PS	x,x	1	MB	3	1 1	
CVT(T) PS2DQ	x,x	1		2	1	
CVTDQ2PD	x,x	2		5	1	
CVT(T)PD2DQ	x.x	_		4	2	
CVTPI2PS	x,mm	2		5	2	
CVT(T)PS2PI	mm,x	1		4	1	
CVTPI2PD	x,mm	2		4	1	
CVT(T) PD2PI	mm,x	2		4	2	
CVTSI2SS	x,r32	2		5	_	
CVT(T)SS2SI	r32,x	1		4	1 1	
CVTSI2SD	x,r32	2		5		
CVT(T)SD2SI	r32,x	1		4	1 1	
, ,						
Arithmetic						
ADDSS SUBSS	x,x	1	MBfadd	2	1	
ADDSD SUBSD	x,x	1	MBfadd	2	1	
ADDPS SUBPS	x,x	1	MBfadd	2	1	
ADDPD SUBPD	x,x	1	MBfadd	2	1	
ADDSUBPS	x,x	1	MBfadd	2	1	
ADDSUBPD	x,x	1	MBfadd	2	1	
HADDPS HSUBPS	x,x	3	MBfadd	5	3	
HADDPD HSUBPD	x,x	3	MBfadd	5	3	

T.	ı		ı	1	1	1
MULSS	x,x	1	MA	3	1	
MULSD	X,X	1	MA	4	2	
MULPS	X,X	1	MA	3	1	
MULPD	X,X	1	MA	4	2	
DIVSS	x,x	1	MA	13	13	
DIVSD	x,x	1	MA	13-20	13-20	
DIVPS	x,x	1	MA	24	24	
DIVPD	x,x	1	MA	21-38	21-38	
RCPSS	x,x	1	MA	5	5	
RCPPS	x,x	3	MA	14	11	
CMPccSS/D	x,x	1	MBfadd	2	1	
CMPccPS/D	x,x	1	MBfadd	2	1	
COMISS/D UCOMISS/D						
	x,x	1	MBfadd	3	1	
MAXSS/D MINSS/D	x,x	1	MBfadd	2	1	
MAXPS/D MINPS/D	x,x	1	MBfadd	2	1	
Math						
SQRTSS	x,x	1	MA	33	33	
SQRTPS	x,x	1	MA	64	64	
SQRTSD	x,x	1	MA	62	62	
SQRTPD	x,x	1	MA	122	122	
RSQRTSS	x,x	1		5	5	
RSQRTPS	x,x	3		14	11	
	,					
Logic						
ANDPS/D	x,x	1	MB	1	1	
ANDNPS/D	X,X	1	MB	1	1	
ORPS/D	x,x	1	MB	1	1	
XORPS/D	x,x	1	MB	1	1	
	7.,7.			·		
Other						
LDMXCSR	m32				31	
STMXCSR	m32				13	
FXSAVE	m4096				97	
FXRSTOR	m4096				201	

VIA-specific instructions

Instruction	Conditions	Clock cycles, approximately
XSTORE	Data available	160-400 clock giving 8 bytes
XSTORE	No data available	50-80 clock giving 0 bytes
REP XSTORE	Quality factor = 0	1300 clock per 8 bytes
REP XSTORE	Quality factor > 0	5455 clock per 8 bytes
REP XCRYPTECB	128 bits key	15 clock per 16 bytes
REP XCRYPTECB	192 bits key	17 clock per 16 bytes
REP XCRYPTECB	256 bits key	18 clock per 16 bytes
REP XCRYPTCBC	128 bits key	29 clock per 16 bytes
REP XCRYPTCBC	192 bits key	33 clock per 16 bytes
REP XCRYPTCBC	256 bits key	37 clock per 16 bytes
REP XCRYPTCTR	128 bits key	23 clock per 16 bytes
REP XCRYPTCTR	192 bits key	26 clock per 16 bytes
REP XCRYPTCTR	256 bits key	27 clock per 16 bytes
REP XCRYPTCFB	128 bits key	29 clock per 16 bytes

REP XCRYPTCFB	192 bits key	33 clock per 16 bytes
REP XCRYPTCFB	256 bits key	37 clock per 16 bytes
REP XCRYPTOFB	128 bits key	29 clock per 16 bytes
REP XCRYPTOFB	192 bits key	33 clock per 16 bytes
REP XCRYPTOFB	256 bits key	37 clock per 16 bytes
REP XSHA1		5 clock per byte
REP XSHA256		5 clock per byte