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 – 2016. Last updated 2016-12-01.

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 Intel.

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

Introduction

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. The code examples in these manuals can be used without restrictions. A GNU Free Documentation License shall automatically come into force when I die. See www.gnu.org/copyleft/fdl.html

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.

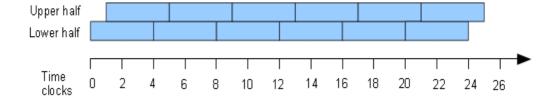
Operands

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

The 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.

Definition of terms

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.

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 μ op 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 μ op at a time. Two μ ops 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 different instruction sets are listed at the end of this manual. 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) are only available under operating systems that support this register set. Instructions that use YMM registers (AVX and later) are only available under operating systems that support this register set.

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 feature 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 needed as input for the next instruction.

Definition of terms

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.

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 μ op 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 μ op can execute simultaneously with another instruction/ μ op 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/ μ ops 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 80486	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. 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.

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.

Monitor The instructions MONITOR and MWAIT are available in some Intel and AMD multiprocessor CPUs with SSE3

SSF4.1 MPSADBW, PHMINPOSUW, PMULDQ, PMULLD, DPPS, DPPD, BLEND.., PMIN., PMAX., ROUND., INSERT., EXTRACT., PMOVSX., PMOVZX., PTEST, PCMPEQQ, PACKUSDW, MOVNTDQA

SSE4.2 CRC32, PCMPESTRI, PCMPESTRM, PCMPISTRI, PCMPISTRM, PCMPGTQ, POPCNT.

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

PCLMULQDQ. **CLMUL**

AVX

The 128-bit XMM registers are extended to 256-bit YMM registers with room for further extension in the future. The use of YMM registers requires operating system support. Floating point vector instructions are available in 256-bit 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.

AVX2

Integer vector instructions are available in 256-bit versions. Furthermore, the 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.

FMA3

(FMA): Fused multiply and add instructions: VFMADDxxxPD, VFMADDxxxPS, 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 AMD. Intel's FMA specification has later been changed to FMA3, which is now also supported by AMD.

MOVBE **MOVBE**

Instruction sets

POPCNT POPCNT PCLMUL PCLMULQDQ

XSAVE

XSAVEOPT

RDRAND RDRAND 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 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). 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	- Codo namo	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 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 Atom 330	Diamondville	6	1C	Step. 2
Intel Bay Trail	Silvermont	6	37	Step. 3
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)

AMD K7

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 pipe-

ine.

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 execu-

tion 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		

MOVZX	l		_		l _	I	1
MOVZX, MOVSX r,m	MOV	sr,r/m	6	9-13	8		
CMOVcc			-				
CMOVcc			1				
XCHG		r,r	1	1			
XCHG		r,m			1/2	-	
XCHG	XCHG	r,r	3	2	1	ALU	
XLAT							
PUSH		r,m			16		on hw
PUSH				5		· ·	
PUSH		r	1		1		
PUSH (D) sr 2 1 ALU, AGU AGU ALU, AGU		i			1		
PUSHF(D)		m	2		1	ALU, AGU	
PUSHA(D)	PUSH	sr	2		1	ALU, AGU	
POP r 2 1 ALU, AGU POP DS/ES/FS/GS 6 10 ALU, AGU POP DS/ES/FS/GS 6 10 ALU, AGU POPF(D) SS 9 18 ALU, AGU POPA(D) 9 4 ALU, AGU POPA(D) 9 4 ALU, AGU LEA r16.[m] 2 3 1 AGU Any addr. size LEA r32.[m] 1 2 13 AGU Any addr. size LAHF 4 3 2 ALU Any addr. size LAHF 4 3 2 ALU Any addr. size LAHF 2 2 2 2 ALU Any addr. size LAHF 1 1 1 1 ALU ALU ANY addr. size LAHF 2 2 2 2 ALU ALU ALU ALU ALU ALU ALU ALU ALU <td>PUSHF(D)</td> <td></td> <td>1</td> <td></td> <td>1</td> <td>ALU, AGU</td> <td></td>	PUSHF(D)		1		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) 9 4 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 LEA r32,[m] 1 2 1/3 AGU Any addr. size LEA r32,[m] 1 2 1/3 AGU Any addr. size LEA r32,[m] 1 1 1 ALU AU ANy addr. size SAHF 2 2 2 ALU AU ANy addr. size SALC 1 1 1 1 ALU ALU SALC 1 1 1 1/3 ALU ADD, SUB r,rm 1	PUSHA(D)		9		4	ALU, AGU	
POP DS/ES/FS/GS 6 10 ALU, AGU ALU, AGU POPF(D) SS 9 18 ALU, AGU ALU, AGU POPA(D) 9 4 ALU, AGU AU, AGU LEA r16,[m] 2 3 1 AGU Any addr. size LEA r32,[m] 1 2 1/3 AGU Any addr. size LAHF 4 3 2 ALU SALC 1 1 1 1 ALU SALC 1 1 1 1 ALU LDS, LES, r,m 10 9 BSWAP ALU ADD, SUB r,r/i 1 1 1/3 ALU ADD, SUB m,r 1 7 2,5 ALU, AGU ADC, SBB r,r/i 1 1 1/3 ALU ADC, SBB m,r/i 1 7 2,5 ALU, AGU ADC, SBB m,r/i 1 1	POP	r	2		1	ALU, AGU	
POP (POPF(D)) SS 9 18 ALU, AGU ANy addr. size LEA (r16,[m]) 2 3 1 AGU ALU, AGU ANy addr. size	POP	m	3		1	ALU, AGU	
POPF(D)	POP	DS/ES/FS/GS	6		10	ALU, AGU	
POPA(D)	POP	SS	9		18	ALU, AGU	
LEA r16,[m] 2 3 1 AGU Any addr. size LEA r32,[m] 1 2 1/3 AGU Any addr. size LAHF 4 3 2 ALU SAHF 2 2 2 ALU SALC 1 1 1 ALU LDS, LES, r,m 10 9 BSWAP ADD, SUB, LES, r,m 10 9 BSWAP ADD, SUB r,r/i 1 1 1/3 ALU ADD, SUB r,r/i 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,r/i 1 7 2,5 ALU, AGU ADC, SBB m,r/i 1 7 2,5 ALU, AGU ADC, SBB m,r/i 1 1 1/3 ALU CMP r,	POPF(D)		2		1	ALU, AGU	
LEA	POPA(D)		9		4	ALU, AGU	
LEA	LEA	r16,[m]	2	3	1	AGU	Any addr. size
LAHF	LEA	1	1		1/3	AGU	
SAHF 2 2 2 ALU SALC 1 1 1 1 ALU LDS, LES, r,m 10 9 BSWAP ALU Arithmetic instructions 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 ADC, SBB m,r/i 1 7 2,5 ALU, AGU CMP r,r/i 1 1 1/3 ALU CMP r,r/i 1 1 1/3 ALU INC, DEC, NEG r 1 1 1 1/3 ALU DAA 16	LAHF		4	3	2	ALU	
SALC	SAHF		2	2		ALU	
DS, LES, RSWAP R	SALC		1				
Arithmetic instructions		r.m	10		9		
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 INC, DEC, NEG r 1 1 1/3 ALU INC, DEC, NEG m 1 7 3 ALU, AGU AAA, AAS 9 5 5 ALU DAS 16 7 7 ALU AAM <td></td> <td></td> <td></td> <td>1</td> <td></td> <td>ALU</td> <td></td>				1		ALU	
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 INC, DEC, NEG r 1 1 1/3 ALU INC, DEC, NEG m 1 7 3 ALU, AGU AAA, AAS 9 5 5 ALU DAS 16 7 7 ALU AAM <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>							
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 INC, DEC, NEG r 1 1 1/3 ALU INC, DEC, NEG m 1 7 3 ALU, AGU AAA, AAS 9 5 5 ALU DAA 12 6 6 ALU AAD 4 5 ALU AAM 31 13 ALU MUL, IMUL r16/m16 3 3 2 ALUO_1 MUL, IMUL r32/m32 3 4 3	Arithmetic instructions						
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/i 1 1/2 ALU, AGU INC, DEC, NEG r 1 1 1/3 ALU, AGU INC, DEC, NEG r 1 1 1/3 ALU, AGU AAA, AAS 9 5 5 ALU DAA 12 6 6 ALU DAS 16 7 7 ALU AAM 31 13 ALU MUL, IMUL r8/m8 3 3 2 ALU0_1 MUL, IMUL r16/m16 3 3 2 ALU0_1 MUL, IMUL r32/m32 3 4 <td>ADD, SUB</td> <td>r,r/i</td> <td>1</td> <td>1</td> <td>1/3</td> <td>ALU</td> <td></td>	ADD, SUB	r,r/i	1	1	1/3	ALU	
ADC, SBB	ADD, SUB	r,m	1	1	1/2	ALU, AGU	
ADC, SBB	ADD, SUB	m,r	1	7	2,5	ALU, AGU	
ADC, SBB CMP CMP r,r/i 1 1 7 2,5 ALU, AGU CMP r,m 1 1/2 INC, DEC, NEG R 1 1 1 1/3 INC, DEC, NEG R 1 1 1 1/3 INC, DEC, NEG R 1 1 1 1/3 INC, DEC, NEG R 1 7 3 ALU AAA, AAS P 5 5 ALU DAA DAA DAA DAA AD AAD AAD AAM MUL, IMUL R8/m8 RUL, IMUL R16/m16 R17 R16/m16 R18 RUL, AGU ALU ALU ALU ALU ALU ALU ALU ALU ALU AL	ADC, SBB	r,r/i	1	1	1/3	ALU	
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 3 ALU, AGU AAA, AAS 9 5 5 ALU DAA 12 6 6 ALU DAS 16 7 7 ALU AAD 4 5 ALU AAM 31 13 ALU MUL, IMUL r8/m8 3 3 2 ALUO_1 MUL, IMUL r16/m16 3 3 2 ALUO_1 dx=4 MUL, IMUL r32/m32 3 4 3 ALUO_1 dx=4	ADC, SBB	r,m	1	1	1/2	ALU, AGU	
CMP r,m 1 1/2 ALU, AGU INC, DEC, NEG r 1 1 1/3 ALU INC, DEC, NEG m 1 7 3 ALU, AGU AAA, AAS 9 5 5 ALU DAS 12 6 6 ALU AAD 4 5 ALUO AAM 31 13 ALU MUL, IMUL r8/m8 3 3 2 ALUO MUL, IMUL r16/m16 3 3 2 ALUO_1 MUL, IMUL r32/m32 3 4 3 ALUO_1	ADC, SBB	m,r/i	1	7	2,5	ALU, AGU	
INC, DEC, NEG	CMP	r,r/i	1	1	1/3	ALU	
INC, DEC, NEG AAA, AAS DAA DAS ADA AAB DAS AAB BAC BAC BAC BAC BAC BAC B	CMP	r,m	1		1/2	ALU, AGU	
AAA, AAS 9 5 5 ALU DAA 12 6 6 ALU DAS 16 7 7 ALU AAD 4 5 ALU AAM 31 13 ALU MUL, IMUL r8/m8 3 3 2 ALUO MUL, IMUL r16/m16 3 3 2 ALUO_1 dx=4 MUL, IMUL r32/m32 3 4 3 ALUO_1	INC, DEC, NEG	r	1	1	1/3	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 MUL, IMUL r16/m16 3 3 2 ALU0_1 dx=4 MUL, IMUL r32/m32 3 4 3 ALU0_1	INC, DEC, NEG	m	1	7	3	ALU, AGU	
DAS 16 7 7 ALU AAD 4 5 ALU0 AAM 31 13 ALU MUL, IMUL r8/m8 3 3 2 ALU0 MUL, IMUL r16/m16 3 3 2 ALU0_1 dx=4 MUL, IMUL r32/m32 3 4 3 ALU0_1	AAA, AAS		9	5	5	ALU	
AAD 4 5 ALU0 AAM 31 13 ALU MUL, IMUL r8/m8 3 3 2 ALU0 Iatency ax=3, Iatency ax=3, dx=4 MUL, IMUL r32/m32 3 4 3 ALU0_1	DAA		12	6	6	ALU	
AAD 4 5 ALU0 AAM 31 13 ALU MUL, IMUL r8/m8 3 3 2 ALU0 Iatency ax=3, Iatency ax=3, dx=4 MUL, IMUL r32/m32 3 4 3 ALU0_1	DAS		16	7	7	ALU	
AAM MUL, IMUL r8/m8 3 3 2 ALU0 latency ax=3, MUL, IMUL r32/m32 3 4 3 ALU0_1				5			
MUL, IMUL r8/m8 3 3 2 ALU0 latency ax=3, dx=4 MUL, IMUL r16/m16 3 3 2 ALU0_1 dx=4 MUL, IMUL r32/m32 3 4 3 ALU0_1	AAM		31	13			
MUL, IMUL r16/m16 3 3 2 ALU0_1 dx=4 MUL, IMUL r32/m32 3 4 3 ALU0_1		r8/m8			2		
MUL, IMUL r32/m32 3 4 3 ALU0_1							
				3			dx=4
IMUL r16,r16/m16 2 3 2 ALU0						_	
	IMUL	r16,r16/m16	2	3	2	ALU0	

T	1		1	1	
IMUL	r32,r32/m32	2	4	2,5	ALU0
IMUL	r16,(r16),i	2	4	1	ALU0
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	11132	1	1	1/3	ALU
1					
CWD, CDQ		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		1	1	1/3	ALU, AGU
TEST	r,r	1	1	1/3	
NOT	r,m	1	1		ALU, AGU
	r			1/3	ALU
NOT	m :/OI	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	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
RCL, RCR	m,1	1	7	4	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
SHLD, SHRD	r,r,cl	7	4	3	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
BT	m,r	5		2	ALU, AGU
BTC, BTR, BTS	r,r/i	2	2	1	ALU, AGU ALU
BTC		5	7	2	
	m,i				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	r,m	20	8	8	ALU, AGU	
BSR	r,m	23	10	10	ALU, AGU	
SETcc	r	1 1	1	1/3	ALU	
SETcc	m	1		1/2	ALU, AGU	
CLC, STC		1 1		1/3	ALU	
CMC		1 1	1	1/3	ALU	
CLD		2	Ţ	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	
						low values = real
JMP	m(far)	17-21	25-33			mode
Jcc	short/near	1		1/3 - 2	ALU	rcp. t.= 2 if jump
J(E)CXZ	short	2		1/3 - 2	ALU	rcp. t.= 2 if jump
LOOP	short	7	3-4	3-4	ALU	
CALL	near	3	2	2	ALU	
						low values = real
CALL	far	16-22	23-32			mode
CALL	r	4	3	3	ALU	
CALL	m(near)	5	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	
RETF		15-23	24-35			low values = real mode
						low values = real
RETF	i	15-24	24-35			mode
IRET		32	81			real mode
INT	i	33	42			real mode
						values are for no
BOUND	m	6		2		jump
INTO		2		2		values are for no jump
						Jamp
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		values per count
REP SCAS		5	2	2		values per count
CMPS		7	6	6		values per count
REP CMPS		6	3-4	3-4		values per count
INLIF CIVIFS		0	J -4	J- 4		values per coult

		AMD K7				
Other NOP (90) Long NOP (0F 1F) ENTER	1 1 i,(1/3 1/3 12	ALU ALU 12	2 one 5 alk if 16	
LEAVE	3		3		3 ops, 5 clk if 16 bit	
CLI	8-		5			
STI CPUID	16- 19-		27			
RDTSC	5		11			
RDPMC	9		11			

Floating point x87 instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
mstruction	Operanus	Орз	Latericy	throughput	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		188		
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 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-
ENICTOW	A.V.		0.40	40	ENGO ALLI	mediately after FCOM FTST
FNSTSW	AX	2	6-12	12		
FSTSW	AX	3	6-12	12	FMISC, ALU	do.
FNSTSW	m16	2		8	FMISC, ALU	do.
FNSTCW	m16	3		1	FMISC, ALU	footowif
FLDCW	m16	14		42	FMISC, ALU	faster if
FEDGW	11110	14		42	FIVIISC, ALO	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	1-2	FMUL	
FIMUL	m	2	4	2	FMUL,FMISC	
I IIVIOL	""	-	7	_	I WIOL,I WIISC	Low values are
FDIV(R)(P)	r/m	1	11-25	8-22	FMUL	for 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		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		
FXSAVE		44		59		
FXRSTOR		85		87		

Integer MMX instructions

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/W						
	mm,r/m	1	2	1/2	FA/M	
PCMPEQ/GT B/W/D	mm,r/m	1	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	FADD	
Logic						
PAND PANDN POR						
PXOR	mm,r/m	1	2	1/2	FA/M	
PSLL/RLW/D/Q						
PSRAW/D	mm,i/mm/m	1	2	1/2	FA/M	
Othor						
Other				4.10	FAND	
EMMS		1		1/3	FANY	

Floating point XMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
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	
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	
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)

oznom monacione (oxociote)									
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes			
Move and convert instructions									
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			

				1		1
Integer instructions						
PAVGUSB	mm,mm	1	2	1/2	FA/M	
PMULHRW	mm,mm	1	3	1	FMUL	
Floating point instruction	 ns					
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. ALUO_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
				throughput	unit	
Move instructions						
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	r,r	1	1	1/3	ALU	
CMOVcc	r,m	1		1/2	ALU, AGU	
XCHG	r,r	3	2	1	ALU	
	.,.		_	-		
XCHG	r,m	3	16	16	ALU, AGU	Timing depends on hw
XLAT		2	5		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	Arry address size
SAHF		1	1	1/3	ALU	
SALC		1	1 1	1/3	ALU	
LDS, LES,	r,m	10	! 	9	ALO	
BSWAP	r	10	1	1/3	ALU	
PREFETCHNTA	m	1	'	1/3	AGU	
PREFETCHT0/1/2		1		1/2	AGU	
SFENCE	m	6			AGU	
LFENCE		1		8 5		
MFENCE		7		16		
IN	r:/DV	7 270		10		
OUT	r,i/DX					
001	i/DX,r	300				
Arithmetic instruction	 					
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	'	1/3	ALU, AGU	
INC, DEC, NEG	r	1	1	1/2	ALU, AGU ALU	
INC, DEC, NEG	m	1	7	3	ALU, AGU	
AAA, AAS	111	9	5	5	ALU, AGU ALU	
DAA		9 12	6	6	ALU	
DAS				7		
		16 4	7 5	'	ALU	
AAD		4	၂		ALU0	

AAM	1	31	13	I	ALU	1
MUL, IMUL	r8/m8	1	3	1	ALU0	
		3		2		lotonov ov=2 dv=4
MUL, IMUL	r16/m16		3-4	1	ALU0_1	latency ax=3, dx=4
MUL, IMUL	r32/m32	2 2	3		ALU0_1	lataran mana 4 maha-5
MUL, IMUL	r64/m64		4-5	2	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	11101	1	1	1/3	ALU	
CWD, CDQ, CQO		1	1	1/3	ALU	
OVID, ODQ, OQO		'	'		7120	
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		7	3	3		
I I	r,CL	/) ၁	3	ALU	
SHL,SHR,SAR,ROL,R	: /Cl	4	7	2	ALII ACII	
OR BOL BOD	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 BT BT BT BTC, BTR, BTS BTC BTR, BTS BTC BTR, BTS BSF BSF BSF BSF BSF BSF BSF CC SETCC CLC, STC CMC CLD STD	m,r,i/CL r,r/i m,i m,r r,r/i m,i m,i m,r r16/32,r r64,r r,r r16,m r32,m r64,m r,m r	8 1 1 5 2 5 4 8 21 22 28 20 22 25 28 1 1 1 1 1 2	6 1 2 7 7 5 8 8 9 10 8 9 10 10	3 1/3 1/2 2 1 2 5 3 8 9 10 8 9 10 10 1/3 1/3 1/3 1/3	ALU, AGU ALU	
Control transfer instru	ıctions					
JMP	short/near	1		2	ALU	
JMP JMP JMP	far r m(near)	16-20 1 1	23-32	2 2	ALU ALU, AGU	low values = real mode low values = real
JMP	m(far)	17-21	25-33	4/2 0	A 1 1 1	mode
Jcc J(E/R)CXZ	short/near short	1 2		1/3 - 2 1/3 - 2	ALU ALU	recip. thrp.= 2 if jump recip. thrp.= 2 if jump
LOOP	short	7	3-4	3-4	ALU	Toolp. unp. 2 in jump
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/for)	16-22	24-33			low values = real
RETN	m(far)	2	24-33 3	3	ALU	mode
RETN	i	2	3	3	ALU	
RETF	•	15-23	24-35	0	7.20	low values = real mode
			-			low values = real
RETF	i	15-24	24-35			mode
IRET		32	81			real mode
INT	i	33	42			real mode
BOUND	m	6		2 2		values are for no jump
INTO		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	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	Execution	Notes
				throughput	unit	
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 immedi-
						ately 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	
		_	•	_		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	do.
FCOM(P), FUCOM(P)	r/m	1	2	1	FADD	
FCOMPP, FUCOMPP	17111	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	T WIGO, ALO	
FPREM		1	7-10	8	FMUL	
FPREM1		1	8-11	8	FMUL	
I F INCIVIT		'	0-11	O	TIVIOL	
Math						
FSQRT		1	27	12	FMUL	
FLDPI, etc.		1		1	FMISC	
FSIN		66	140-190	'	1 101100	
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			
		, 0	170			
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		
IAROTOR		101		9 J		

Integer MMX and XMM instructions

integer with and Awith metactions									
Operands	Ops	Latency			Notes				
r32, mm	2	4	2	FMICS, ALU					
mm, r32	2	9	2	FANY, ALU					
mm,m32	1		1/2	FANY					
r32, xmm	3	2	2	FMISC, ALU					
xmm, r32	3	3	2						
xmm,m32	2		1	FANY					
m32, r	1		1	FMISC					
	r32, mm mm, r32 mm,m32 r32, xmm xmm, r32 xmm,m32	r32, mm 2 mm, r32 2 mm,m32 1 r32, xmm 3 xmm, r32 3 xmm,m32 2	Operands Ops Latency r32, mm 2 4 mm, r32 2 9 mm,m32 1 2 r32, xmm 3 2 xmm, r32 3 3 xmm,m32 2 3	Operands Ops Latency throughput Reciprocal throughput r32, mm mm, r32 2 4 2 mm, r32 2 9 2 mm,m32 1 1/2 r32, xmm 3 2 2 xmm, r32 3 3 2 xmm,m32 2 1 1	Operands Ops Latency throughput throughput Reciprocal throughput unit Execution unit r32, mm mm, r32 2 4 2 FMICS, ALU FANY, ALU FANY, ALU FANY mm,m32 1 1/2 FANY r32, xmm 3 2 2 xmm, r32 3 3 2 xmm,m32 2 1 FANY				

I	1 1		I	- 	T	h
MOVED (MOVED)	mC 4 /s	_	4		EMICO ALLI	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		
MOVDQU	m,xmm	5		2		
MOVDQ2Q	mm,xmm	1	2	1/2	FA/M	
MOVQ2DQ	xmm,mm	2	2	1	FA/M, FMISC	
MOVNTQ	m,mm	1		2	FMISC	
MOVNTDQ	m,xmm	2		3	FMISC	
PACKSSWB/DW						
PACKUSWB	mm,r/m	1	2	2	FA/M	
PACKSSWB/DW						
PACKUSWB	xmm,r/m	3	3	2	FA/M	
PUNPCKH/LBW/WD/						
DQ	mm,r/m	1	2	2	FA/M	
PUNPCKH/LBW/WD/				_		
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		
MASKMOVDQU	xmm,xmm	64		26		
PMOVMSKB	r32,mm/xmm	1	2	1	FADD	
PEXTRW	r32,mm/x,i	2	5	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	,	4		4 /0	E A /B A	
	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	
I CIVII LOJO I DIVVID	A111111,1/111	_		ļ '	I / VIVI	

PMULLW PMULHW PMULHUW PMULUDQ	mm,r/m	1	3	1	FMUL	
PMULLW PMULHW PMULHUW PMULUDQ	xmm,r/m	2	3	2	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						
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	xmm,i	2	2	1	FA/M	
Other						
EMMS		1		1/3	FANY	

Floating point XMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions	1					
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	rr	1	2	1/2	FA/M	
MOVERN 3	r,r	'		1/2	r-A/IVI	
MOVHPS/D, MOVLPS/D	r,m	1		1	FMISC	
MOVHPS/D,				4	EMICO	
MOVLPS/D	m,r	1		1	FMISC	0050
MOVDDUP	r,r	2	2	1		SSE3
MOVSH/LDUP	r,r	2	2	2		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	1
UNPCK H/L PS/D	r,r/m	2	3	3	FMUL	
ON ONTHE TOTAL	1,17111	_			TWOL	
Conversion						
CVTPS2PD	r,r/m	2	4	2	FMISC	
CVTPD2PS	r,r/m	4	8	3	FMISC	
CVTSD2SS	r,r/m	3	8	8	FMISC	
CVTSS2SD	r,r/m	1	2	1	FMISC	
CVTDQ2PS	r,r/m	2	5	2	FMISC	
CVTDQ2PD	r,r/m	2	5	2	FMISC	
CVT(T)PS2DQ	r,r/m	2	5	2	FMISC	
CVT(T)PD2DQ	r,r/m	4	8	3	FMISC	
CVTPI2PS	xmm,mm	1	4	1	FMISC	
CVTPI2PD	xmm,mm	2	5	2	FMISC	
CVT(T)PS2PI	mm,xmm	1	6	1	FMISC	
CVT(T)PD2PI	mm,xmm	3	8	2	FMISC	
CVTSI2SS	xmm,r32	3	14	2	FMISC	
CVTSI2SD	xmm,r32	2	12	2	FMISC	
CVT(T)SD2SI	r32,xmm	2	10	2	FMISC	
CVT(T)SS2SI	r32,xmm	2	9	2	FMISC	
011(1)00201	,,,,,,,,,,,,	_		_		
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	4	1	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/D	r,r/m	1	2	1	FADD	
Logic						
ANDPS/D ANDNPS/D					E	
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	
110011100	1,1/111	'	5	ļ .	I WICL	

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. ALUO_1 means that ALUO 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
				throughput	unit	
Move instructions						
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 MOVSXD MOVSXD CMOVcc CMOVcc XCHG XCHG XLAT PUSH PUSH PUSH PUSH PUSHP(D/Q) PUSHA(D) POP POP POP POP POP POP POP POP SALEA LEA LEA LAHF SAHF SALC	r,m r64,r32 r64,m32 r,r r,m r,r r,m r m DS/ES/FS/GS SS r16,[m] r32/64,[m]	1 1 1 1 1 2 2 2 1 1 2 2 9 9 1 3 6 10 28 9 2 1 1 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	4 1 4 1 4 1 21 5 6 3 10 26 16 6 3 1 2 3 1	1/2 1/3 1/2 1/3 1/2 1 19 5 1/2 1 1 3 6 1/2 1 8 16 11 6 1 1/3 1/3 2 1/3 1	ALU, AGU	Any address size ≤ 2 source operands W. scale or 3 opr.
LDS, LES, BSWAP PREFETCHNTA PREFETCHT0/1/2 PREFETCH(W) SFENCE LFENCE	r,m r m m m	10 1 1 1 1 6	1	10 1/3 1/2 1/2 1/2 8 1	ALU AGU AGU AGU	3DNow
MFENCE IN OUT Arithmetic instruction	r,i/DX i/DX,r	4 ~270 ~300		33		
ADD, SUB ADD, SUB ADD, SUB ADD, SUB ADC, SBB ADC, SBB ADC, SBB CMP CMP INC, DEC, NEG INC, DEC, NEG AAA, AAS DAA DAS AAD AAM	r,r/i r,m m,r r,r/i r,m m,r/i r,r/i r,m r	1 1 1 1 1 1 1 1 1 1 9 12 16 4 30	1 4 1 4 1 7 5 6 7 5	1/3 1/2 1 1/3 1/2 1 1/3 1/2 1/3 2 5 6 7 5	ALU ALU, AGU ALU, AGU ALU, AGU ALU, AGU ALU, AGU ALU, AGU ALU ALU, AGU ALU ALU ALU ALU ALU ALU ALU	

				1 4		1
MUL, IMUL	r8/m8	1	3	1	ALU0	
MUL, IMUL	r16/m16	3	3	2	ALU0_1	latency ax=3, dx=4
MUL, IMUL	r32/m32	2	3	1	ALU0_1	
MUL, IMUL	r64/m64	2	4	2	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	
1	, , ,		4	2		
IMUL	r16,m16,i	3			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 divi-
IDIV	r16/m16		24-39	24-39	ALU	dend. See AMD soft-
IDIV	r32/m32		24-55	24-55	ALU	ware optimization
IDIV	r64/m64		24-87	24-87	ALU	guide.
	104/11104	4				
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	ÁLU	
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		7	3	3	ALU	
	r,i	9				
RCL	r,CL		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	ÁLU	
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	ALO, AGO ALU	
BT		1	'	1/3	ALU, AGU	
1	m,i		_			
BT DTC DTC	m,r	5	7	2	ALU, AGU	
BTC, BTR, BTS	r,r/i	2	2	1/3	ALU	

BTC BTR, BTS BTC BTR, BTS BSF BSR BSF BSR POPCNT LZCNT SETcc SETcc CLC, STC	m,i m,r m,r r,r r,r r,m r,r/m r,r/m r	5 4 8 8 6 7 8 1 1 1	9 9 8 8 4 4 7 7 2 2	1,5 1,5 10 7 3 3 3 1 1 1/3 1/2 1/3	ALU, AGU ALU, AGU ALU, AGU ALU, AGU ALU ALU, AGU ALU, AGU ALU ALU ALU ALU ALU ALU	SSE4.A / SSE4.2 SSE4.A, AMD only
CMC CLD STD		1 1 2	1	1/3 1/3 2/3	ALU ALU ALU	
Control transfer instru						
JMP	short/near	1		2	ALU	
JMP	far	16-20	23-32	_		low values = real mode
JMP	r	1		2	ALU	
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	2	A1.11	low values = real mode
RETN		2 2	3 3	3	ALU	
RETN RETF	İ	15-23	ა 24-35	3	ALU	low values = real made
RETF	i	15-23	24-35 24-35			low values = real mode low values = real mode
IRET	'	32	81			real mode
INT	i	33	42			real mode
BOUND	m '	6	72	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		
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

Floating point x87	instruction	ns				
Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
				throughput	unit	
Move instructions						
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	
						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	
FLDCW	m16	12		14	FMISC, ALU	faster if unchanged
						J
Arithmetic instruction	IS					
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		
FRNDINT		6		37		

FPREM		1		7	FMUL	
FPREM1		1		7 7	FMUL	
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 XMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
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,(x)mm	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,(x)mm	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	,				, , ,	
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/	,,,,,,	•	_	,, <u>,,</u>	170101	
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		1	2		FA/M	
	xmm,xmm,i	-		1/2	FAVIVI	
MASKMOVQ	mm,mm	32		13		
MASKMOVDQU	xmm,xmm	64		24	FADD	
PMOVMSKB	r32,mm/xmm	1	3	1	FADD	
PEXTRW	r32,(x)mm,i	2	6	1		
PINSRW	(x)mm,r32,i	2	9	3	FA/M	
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	IS					
PADDB/W/D/Q						
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	111111/2111111,1/111	- 1		1/2	FAVIVI	
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			2	1/2	FA/M	
	mm/xmm,r/m	1	3			
PSADBW	mm/xmm,r/m	1	3	1	FADD	
Logic						
PAND PANDN POR	-					
PXOR	mm/xmm,r/m	1	2	1/2	FA/M	
	111111/2111111,1/111	'		1/2	I AVIVI	
PSLL/RL W/D/Q PSRAW/D	mm i/mm/m	1	2	1/2	FA/M	
	mm,i/mm/m	ı		1/2	[FAVIVI	
PSLL/RL W/D/Q PSRAW/D	v i//v/mm	1	3	1/2	FA/M	
PSLLDQ, PSRLDQ	x,i/(x)mm xmm,i		3	1/2	FA/M	
F SLLDQ, FSKLDQ	XIIIII,I	1	_ S	1/2	Γ <i>Α</i> VIVI	
1			I			

Other				
EMMS	1	1/3	FANY	

Floating point XMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions						
MOVAPS/D	r,r	1	2,5	1/2	FANY	
MOVAPS/D	r,m	1	2	1/2	?	
MOVAPS/D	m,r	2	2	1	FMUL,FMISC	
MOVUPS/D	r,r	1	2,5	1/2	FANY	
MOVUPS/D	r,m	1	2	1/2	?	
MOVUPS/D	m,r	3	3	2	FMISC	
MOVSS/D	r,r	1	2	1/2	FA/M	
MOVSS/D	r,m	1	2	1/2	?	
MOVSS/D	m,r	1	2	1	FMISC	
MOVHLPS,						
MOVLHPS	r,r	1	3	1/2	FA/M	
MOVHPS/D,						
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	1 WIGO	
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	

K10

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	r,r/m	1	2	1	FADD	
MAXPS/D MINPS/D	r,r/m	1	2	1	FADD	
CMPccSS/D	r,r/m	1	2	1	FADD	
CMPccPS/D	r,r/m	1	2	1	FADD	
COMISS/D	,					
UCOMISS/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 unit	Notes
Move and convert in	structions					
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	mm,mm	1	2	1/2	FA/M	3DNow extension
Integer instructions						
PAVGUSB	mm,mm	1	2	1/2	FA/M	
PMULHRW	mm,mm	1	3	1	FMUL	
Floating point instru	ctions					
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	

K10

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 throughput	Execution pipes	Notes
Move instructions				tinougnput	pipes	
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	r,r	2	1	1	EX01	
	,,,					Timing depends on
XCHG	r,m	2	~50	~50	EX01	hw
XLAT		2	6	2		
PUSH	r	1		1		
PUSH	i	1		1		
PUSH	m	2		1.5		
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		
LEA	r16,[m]	2	2-3		EX01	any addr. size
LEA	r32,[m]	2	2-3		EX01	16 bit addr. size
						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		
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 instructio						
ADD, SUB	r,r	1	1	0.5	EX01	
ADD, SUB	r,i	1	1	0.5	EX01	

			Danaozei		
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	111	10	6	'	LXUI
DAA		16	9		
DAS		20	10		
AAD		4	6		
AAM		9		20	
	*O/ma O		20	20	EV4
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	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
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
1.201	1,1	'		0.5	LAUI

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	
RCR	r,1	1	1		EX01	
RCR	r,i	15	8		EX01	
RCR	r,cl	16	8		EX01	
SHLD, SHRD	r,r,i	6	3	3	EX01	
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	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	
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	OOL4.2
SETcc	m	1		1	EX01	
CLC, STC		1		0.5	EX01	
CMC		1	1	0.0	EX01	
CLD		2	'	3	LXOT	
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	3	1	P1	SSE4A
Control transfer instru	ctions					
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

CALL	near	2		2	EX1	
CALL	r	2		2	EX1	
CALL		3		2	EX1	
	m			2		
RET		1			EX1	
RET	i	4		2-3	EX1	
BOUND	m	11		5		for no jump
INTO		4		24		for no jump
String instructions						
LODS		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 per 10B				Desi Case
				3		a ma a II m
REP MOVS		2n		2n		small n
REP MOVS		4 per 16B		3 per 16B		best case
SCAS		3		3		
REP SCAS		7n		4n		
CMPS		6		3		
REP CMPS		9n		4n		
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	α,δ	2		4		
CPUID		37-63		112-280		
RDTSC				42		
		36				
RDPMC		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	Floating point x87 instructions									
Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Domain, notes				
Move instructions				tinougnput	pipes					
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	P				
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	 									
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	19-65		P0	fp				
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		P0 P1 P3	
FXTRACT		12	10	5	P0 P1 P3	
F2XM1		10	64-71		P0 P1 P3	
FYL2X		10-175			P0 P1 P3	
FYL2XP1		10-175			P0 P1 P3	
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		
VMOVDQA	m256,ymm	4	5	3	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	1-2		
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	(x)mm,r/m	1	2	1	P1	
PACKUSWB	(x)mm,r/m	1	2	1	P1	
PUNPCKH/LBW/WD/D		4		_	D4	
Q	(x)mm,r/m	1	2	1	P1	

					i	
PUNPCKHQDQ	xmm,r/m	1	2	1	P1	
PUNPCKLQDQ	xmm,r/m	1	2	1	P1	
PSHUFB	(x)mm,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	(x)mm,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	7
PMOVSXBW/BD/BQ/	7011111,11,1	_		_		
WD/WQ/DQ	xmm,xmm	1	2	1	P1	SSE4.1
PMOVZXBW/BD/BQ/W	,					
D/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
Arithmetic instructions	S					
PADDB/W/D/Q/SB/SW/						
USB/USW	(x)mm,r/m	1	2	0.5	P23	
PSUBB/W/D/Q/SB/SW/						
USB/USW	(x)mm,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	(x)mm,r/m	1	2	0.5	P23	
PCMPEQQ	(x)mm,r/m	1	2	0.5	P23	SSE4.1
PCMPGTQ	(x)mm,r/m	1	2	0.5	P23	SSE4.2
PMULLW PMULHW						
PMULHUW PMULUDQ	(-)	_		4	D0	
DM III I D	(x)mm,r/m	1	4	1	P0	22511
PMULLD	xmm,r/m	1	5	2	P0	SSE4.1
PMULDQ	xmm,r/m	1	4	1	P0	SSE4.1
PMULHRSW	(x)mm,r/m	1	4	1	P0	SSSE3
PMADDWD	(x)mm,r/m	1	4	1	P0	
PMADDUBSW	(x)mm,r/m	1	4	1	P0	
PAVGB/W	(x)mm,r/m	1	2	0.5	P23	
PMIN/MAX SB/SW/ SD		_		0.5	B00	
UB/UW/UD	(x)mm,r/m	1	2	0.5	P23	22511
PHMINPOSUW	xmm,r/m	2	4	1	P1 P23	SSE4.1
PABSB/W/D	(x)mm,r/m	1	2	0.5	P23	SSSE3
PSIGNB/W/D	(x)mm,r/m	1	2	0.5	P23	SSSE3
PSADBW	(x)mm,r/m	2	4	1	P23	
MPSADBW	x,x,i	8	8	4	P1 P23	SSE4.1
VIDOONAD AAVID IO	, .				Doo	AMD XOP
VPCOMB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
VDCOMUD/M/D/O		4	_	0.5	Doo	AMD XOP
VPCOMUB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7

\(\text{\text{CD}}\)		l	I	I		1
VPHADDBW/BD/BQ/ WD/WQ/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPHADDUBW/BD/BQ/	X,X/111	ı		0.5	F23	AIVID XOP
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	1	4	1	P0	AMD XOP
VPMACSSWW/WD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSSDD	x,x,x/m,x	1	5	2	P0	AMD XOP
VPMACSSDQH/L	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMADCSWD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMADCSSWD	x,x,x/m,x	1	4	1	P0	AMD XOP
Logic						
PAND PANDN POR						
PXOR	(x)mm,r/m	1	2	0.5	P23	
PSLL/RL W/D/Q	, ,				_,	
PSRAW/D	(x)mm,r/m	1	3	1	P1	
PSLL/RL W/D/Q PSRAW/D	()		_	4	D4	
	(x)mm,i	1	2	1	P1	
PSLLDQ, PSRLDQ	xmm,i	1	2	1	P1	00544
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
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 x,x,i	7	14	3	P1 P2 P3	SSE4.2
PCMPISTRM	x,x,i x,x,i	7	7	4	P1 P2 P3	SSE4.2
	7,7,1	•		·		3021.2
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
	77-					
Other						
EMMS		1		0.25		

Floating point XMM and YMM instructions

Instruction	Operands	Ops	 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	400	_		0.5		
MOVUPS/D	x,m128	1	6	0.5		
VMOVAPS/D VMOVUPS/D	v m256	2	6	1-2		
	y,m256		0	1-2		
MOVAPS/D MOVUPS/D	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		6	0.5	101	īρ
MOVSS/D	m32/64,x		5	1		
MOVHPS/D	11132/04,X	I I	5	1		
MOVLPS/D	x,m64	1	7	1		
MOVHPS/D	m64,x	2	8	1	P1 P3	
MOVLPS/D	m64,x	1	7	1	P3	
MOVLHPS MOVHLPS	X,X	1	2	1	P1	ivec
MOVMSKPS/D	r32,x	2	10	1	P1 P3	1700
VMOVMSKPS/D	r32,y	_	10		1110	
MOVNTPS/D	m128,x	1	6	2	P3	
VMOVNTPS/D	m256,y			_		
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		
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	10	1	P1 P3	
1		I .	1	I .	1	ı l

EXTRACTPS	m32,x,i	2	14	1	P1 P3	
VEXTRACTF128	x,y,i	1	2	1	P23	ivec
VEXTRACTF128	m128,y,i	2	7	1	P23	
INSERTPS	x,x,i	1	2	1	P1	
INSERTPS	x,m32,i	1		1	P1	
VINSERTF128	y,y,x,i	2	2	1	P23	ivec
VINSERTF128	y,y,m128,i	2	9	1	P23	
VMASKMOVPS/D	x,x,m128	1	9	0.5	P01	
VMASKMOVPS/D	y,y,m256	2	9	1	P01	
VMASKMOVPS/D	m128,x,x	18	22	7	P0 P1 P2 P3	
VMASKMOVPS/D	m256,y,y	34	25	13	P0 P1 P2 P3	
Conversion						
CVTPD2PS	x,x	2	7	1	P01	fp
VCVTPD2PS	x,y	4	7	2	P01	fp
CVTPS2PD	X,X	2	7	1	P01	fp
VCVTPS2PD	y,x	4	7	2	P01	fp
CVTSD2SS	X,X	1	4	1	P0	fp
CVTSS2SD	X,X X,X	1	4	1	P0	fp
CVTDQ2PS	X,X X,X	1	4	1	P0	fp
VCVTDQ2PS		2	4	2	P0	fp
CVT(T) PS2DQ	у, у	1	4	1	P0	fp
VCVT(T) PS2DQ	X,X	2	4	2	P0	
CVTDQ2PD	y,y	2	7	1	P01	fp fn
VCVTDQ2PD	X,X	4	8	2	P01	fp fn
	y,x	2	7	1	P01	fp
CVT(T)PD2DQ	X,X	4	7	2	P01	fp
VCVT(T)PD2DQ	x,y					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
CVTSI2SS	x,r32	2	14	1	P0	fp
CVT(T)SS2SI	r32,x	2	13	1	P0	fp
CVTSI2SD	x,r32/64	2	14	1	P0	fp
CVT(T)SD2SI	r32/64,x	2	13	1	P0	fp
A!41						
Arithmetic			5.0	0.5	D04	£
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	yyy/m	2	5-6	1	P01	fma
ADDSUBPS/D	y,y,y/m x,x/m	1	5-6	0.5	P01	fma
VADDSUBPS/D		2	5-6	1	P01	fma
VADDSUBPS/D	y,y,y/m		5-6	I	PUT	IIIIa
HADDPS/D HSUBPS/D	x,x	3	10	2	P01 P1	ivec/fma
HADDPS/D HSUBPS/D VHADDPS/D	x,m128	4		2	P01 P1	ivec/fma
VHSUBPS/D VHADDPS/D	y,y,y	8	10	4	P01 P1	ivec/fma
VHSUBPS/D	y,y,m	10		4	P01 P1	ivec/fma

MIII CO MIII CD		4	.	0.5	D04	£
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						
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/						
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	.0	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	00	•	101	AMD FMA4
, an other rank transcrate						71112 1 1117 (1
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					P01	· ·
VRSQRTPS	x,x/m	1	5	1	P01	fp
	y,y/m	2	5 10	2	P01 P01	fp
VFRCZSS/SD/PS/PD	X,X	2 3		2 2		AMD XOP
VFRCZSS/SD/PS/PD	x,m	3	10	2	P01	AMD XOP
Lania						
Logic						
AND/ANDN/OR/XORPS/PD	x,x/m	1	2	0.5	P23	ivec
VAND/ANDN/OR/XOR	^,^/111	'		0.5	r ZJ	1000
PS/PD	y,y,y/m	2	2	1	P23	ivec
. 5/1 5	y, y, y/111			I I	F 20	IVEC
Other						
VZEROUPPER		9				32 bit mode
				4		
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		l
STMXCSR	m32	2	19	19	P0 P3		ĺ
FXSAVE	m4096	67	136	136	P0 P1 P2 P3		ĺ
FXRSTOR	m4096	116	176	176	P0 P1 P2 P3		ĺ
XSAVE	m	122	196	196	P0 P1 P2 P3		ĺ
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 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

Integer Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions	1					
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		1		
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	r,m	1	5	0.5	EX01	
MOVZX		1	4	0.5	EX01	
MOVSXD	r,m r64,r32	1	1	0.5	EX01	
MOVSXD			5	0.5	EX01	
	r64,m32	1				
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	
		_				Timing depends on
XCHG	r,m	2	~40	~40	EX01	hw
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		1		
POP	m	2		1		
POPF(D/Q)		34		18		
POPA(D)		14		8		
LEA	r16,[m]	2	2-3		EX01	any addr. size
LEA	r32,[m]	2	2-3		EX01	16 bit addr. size
	7					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		
BSWAP	r	1	1 1	0.5	EX01	
PREFETCHNTA		1	'	0.5	LAUI	
PREFETCHT0/1/2	m	1		0.5		
PREFETCH/W	m					DDEEETOU\A/
	m	1		0.5		PREFETCHW
SFENCE		7		81		

			Filedriver		
LFENCE		1		0,25	
MFENCE		7		81	
Arithmetic instructions	S				
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
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	111	10	6		LXOI
DAA		16	9		
DAS		20	10		
AAD		4	6		
AAM		10	15	15	
MUL, IMUL	r8/m8	10	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		4	2	EX1
IMUL	r32,r32/m32		4	2	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	r64,m64,i	2		4	EX1
DIV	r8/m8	9	17-22	13-22	EX0
DIV	r16/m16	7	13-26	13-22	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	13-71	EX0
IDIV	r16/m16	7	13-26	13-16	EX0
IDIV	r32/m32	2	13-26	13-25	EX0
IDIV	r64/m64	2	13-40	13-40	EX0
CBW, CWDE, CDQE	104/11104	1	13-71	10-71	EX01
CDQ, CQO		1	1	0.5	EX01
CDQ, CQO CWD		2	1	1 1	EX01 EX01
CAAD			'	'	EAUI
Logio inotroctione					
AND, OR, XOR	rr	1	1	0.5	EX01
AND, OR, XOR	r,r	1 1	1	0.5	EX01
AND, OR, AOR	r,i	ļ I	1	0.5	EAUI

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	7-8	1	EX01	
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	
		•		0.5		
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		1	1	0.5	EX01	
	r,r/i		ı			
BT	m,i	1		0.5	EX01	
ВТ	m,r	7		3,5	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	
	111	1		0.5		
CLC, STC			_	0.5	EX01	
CMC		1	1		EX01	
CLD		2		3		
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		BMI1
BEXTR	r,r,r	2	2	0.67		BMI1
BEXTR	r,r,i	2	2	0.67		AMD TBM
BLSI	r,r	2	2	1		BMI1
BLSMSK	r,r	2	2	1		BMI1
BLSR		2	2	1		BMI1
1	r,r	2	2	1		AMD TBM
BLCFILL	r,r					
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	r,r	2	2	1		AMD TBM
BLSI	r,r	2	2	1		AMD TBM

TIMSKC r,r 2 2 2 1 1 AMD TBM TBM T	BLSIC	r,r	2	2	1 1		AMD TBM
TZMSK							
Control transfer instructions							
JMP	Zimork	.,.	_	_	·		7 7
JMP	Control transfer instru	ctions					
JMP r 1 2 EX1 J JMP JMP m 1 2 EX1 Z I JMP JMP JMP M			1 1		2	EX1	
JMP m 1 2 EX1 2 if jumping Jused CMP+Jcc short/near 1 1-2 EX1 2 if jumping JUE/RYCXZ short 1 1-2 EX1 2 if jumping LOOP short 1 1-2 EX1 2 if jumping LOOPE LOONNE short 1 1-2 EX1 2 if jumping CALL short 1 1-2 EX1 2 if jumping CALL near 2 2 EX1 2 if jumping CALL near 2 2 EX1 2 if jumping CALL r 2 2 EX1 2 if jumping CALL near 2 2 EX1 2 if jumping CALL r 2 2 EX1 2 if jumping CALL r 2 2 EX1 2 if jumping CALL r 2 2 EX1 2 if jumping CALL <td< td=""><td>II</td><td></td><td>1 1</td><td></td><td></td><td></td><td></td></td<>	II		1 1				
JCC	II	m	1 1			EX1	
fused CMP+Jcc J(E/R)CXZ short/near short 1 1-2 EX1 2 if jumping 2		short/near	1 1			EX1	2 if iumping
J(E/R)CXZ	II		1 1				
LOOP LOOPE LOOPNE Short 1			1 1				
LOOPE LOOPNE short near 2			1				
CALL near 2 2 EX1 CALL r 2 2 EX1 CALL m 3 2 EX1 CALL m 3 2 EX1 RET 1 4 2 EX1 RET i 4 2 EX1 BOUND m 11 5 EX1 BOUND m 4 2 EX1 BOUND m 4 2 EX1 BOUND m 3 3 3 3 REP LODS m32/m64 6 3 3 3 3 3 3 3 3 3 3 3 3 3 3	II		1 1				
CALL CALL r m 2 3 3 3 4 2 2 2 2 2 2 2 2 2 2 2 3 EX1 2 2 2 2 3 EX1 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3			1				2 ii jaiiipiiig
CALL m 3 2 EX1 RET 1 2 EX1 RET i 4 2 EX1 BOUND m 11 5 EX1 BOUND m 11 5 for no jump String instructions LODS 3 3 3 REP LODS m8/m16 6n 3n 3 REP LODS m32/m64 6n 3n small REP LODS 3 3 3 small small n STOS 3 3 3 small n <	II						
RET i 4 2 EX1 RET i 4 2 EX1 BOUNDD m 11 5 for no jump INTO 4 2 EX1 EX1 LODS m3 m8/m16 6n 3n	II						
RET	II	***					
BOUND INTO		i	1 1		2		
NTO						LXI	for no jump
String instructions		111	1				
Name	INTO		4		2		ioi no jump
Name	String instructions						
REP LODS m8/m16 6n 3n 2.5n STOS 3 3 3 3 REP STOS 1n 1n 1n small n REP STOS 3 per 16B 3 per 16B best case MOVS 5 3 1n small n REP MOVS 4.5 pr 16B 3 per 16B best case SCAS 7n 3 per 16B best case SYDCHONIZATION 6 3 3 per 16B best case SYDCHONIZATION 7n 4 per 16B 3 per 16B 3 per 16B best case SYDCHONIZATION 7n 4 per 16B 3 per 16B 3 per 16B 3 per 16B 3 per 16B 4 per 16B 4 per 16B 4 per 16B			3		3		
REP LODS STOS REP STOS REP STOS REP STOS MOVS REP MOVS REP MOVS REP MOVS REP MOVS REP SCAS CAS REP SCAS CMPS REP CAS CMPS REP CAPS 1 - 3 n moder 16 mode		m8/m16	1				
STOS 3 1n 3 1n small n best case MOVS 5 3 small n best case movs small n best case small n set case sma	II		1				
REP STOS 1n 3 per 16B 3 per 16B best case MOVS 1-3n 1n 3 per 16B small n MOVS 1-3n 1n small n best case REP MOVS 4.5 pr 16B 3 per 16B best case SCAS 3 3 per 16B best case SCAS 7n 3 -4n best case CMPS 6 3 3 -4n CMPS 6 3 3 -4n CMPS 9n 4n 4n Synchronization 0 0 0 0 LOCK ADD m,r 4 20 0 LOCK XADD m,r 4 -39 0 CMPXCHG m,r8/16 5 -23 0 LOCK CMPXCHG m,r8/16 5 -40 0 CMPXCHG8B m64 18 -25 LOCK CMPXCHG8B m64 18 -22 CMPXCHG16B m128 22 -80	II	11132/11104					
REP STOS 3 per 16B 3 per 16B 3 per 16B best case MOVS 1-3n 1n small n small n REP MOVS 4.5 pr 16B 3 per 16B best case SCAS 3 3 per 16B 3 per 16B small n SCAS 3 3 per 16B best case SCAS 3 3 per 16B small n SCAS 7n 3-4n best case CMPS 6 3 3 REP CMPS 9n 4n 4n Synchronization m,r 1 40 LOCK ADD m,r 4 20 LOCK XADD m,r 4 23 LOCK CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r8/16 5 ~40 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	II						emall n
MOVS 5 1-3n 3 1n small n best case REP MOVS 4.5 pr 16B 3 per 16B 3 per 16B best case SCAS 3 3 3 per 16B best case REP SCAS 7n 3-4n 3 per 16B best case CMPS 6 3 3 per 16B best case Synchronization 6 3 4n 4n Synchronization m,r 1 -40 4n 4n Synchronization m,r 4 20 20 4n 20 4n 4n 20	II						
REP MOVS 1-3n 1n 3 per 16B small n REP MOVS 3 4n			1 '				Desi Case
REP MOVS 4.5 pr 16B 3 per 16B best case SCAS 3 3-4n 3-4n CMPS 6 3 3-4n CMPS 9n 4n Synchronization LOCK ADD m,r 1 ~40 XADD m,r 4 20 LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHGBB m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none					_		emall n
SCAS 3 3 3-4n REP SCAS 7n 3-4n 3 CMPS 6 3 3 REP CMPS 9n 4n 3 Synchronization LOCK ADD m,r 1 ~40 XADD m,r 4 20 LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	II						
REP SCAS 7n 3-4n CMPS 9n 3 REP CMPS 9n 4n Synchronization LOCK ADD m,r 1 ~40 XADD m,r 4 20 LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none							Desi Case
CMPS 6 3 REP CMPS 9n 4n Synchronization LOCK ADD m,r 1 ~40 XADD m,r 4 20 LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	II						
Synchronization DOCK ADD M,r			1				
Synchronization LOCK ADD m,r 1 ~40 XADD m,r 4 20 LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none							
LOCK ADD m,r 1 ~40 XADD m,r 4 20 LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	REP CIVIPS		911		411		
LOCK ADD m,r 1 ~40 XADD m,r 4 20 LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	Synchronization						
XADD m,r 4 20 LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r3/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none		m r	1 1	~40			
LOCK XADD m,r 4 ~39 CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	II		1				
CMPXCHG m,r8/16 5 23 LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none			1				
LOCK CMPXCHG m,r8/16 5 ~40 CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none							
CMPXCHG m,r32/64 6 20 LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none							
LOCK CMPXCHG m,r32/64 6 ~40 CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none							
CMPXCHG8B m64 18 25 LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none			1				
LOCK CMPXCHG8B m64 18 ~42 CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none			1				
CMPXCHG16B m128 22 66 LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none							
LOCK CMPXCHG16B m128 22 ~80 Other NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none							
Other 1 0.25 none NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	II						
NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	LOCK CIVIFACEG 10B	111120		~00			
NOP (90) 1 0.25 none Long NOP (0F 1F) 1 0.25 none	Other						
Long NOP (0F 1F) 1 0.25 none			1		0.25	none	
	` '						
	, ,						

ENTER	a,0	13		21	
ENTER	a,b	20+3b		16+4b	
LEAVE		2		4	
CPUID		38-64		105-271	
XGETBV		4		30	
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

Floating point x87	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	 S									
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						
Other		_		0.05		
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	(x)mm,r/m	1	2	1	P1	
PACKUSWB	(x)mm,r/m	1	2	1	P1	
PUNPCKH/LBW/WD/D						
Q	(x)mm,r/m	1	2	1	P1	
PUNPCKHQDQ	xmm,r/m	1	2	1	P1	
PUNPCKLQDQ	xmm,r/m	1	2	1	P1	
PSHUFB	(x)mm,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	(x)mm,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	332
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	1	1	P1	AMD SSE4A
PMOVSXBW/BD/BQ/	Α,Λ	'		'		7 WID OOL 47 (
WD/WQ/DQ	x,x	1	2	1	P1	SSE4.1
PMOVZXBW/BD/BQ/W	Α,Λ	'	_	'		OOL4.1
D/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
V 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	7,7,7,7,7	•	_	•		7.111.2 7.01
Arithmetic instructions	 					
PADDB/W/D/Q/SB/SW/						
USB/USW	(x)mm,r/m	1	2	0.5	P23	
PSUBB/W/D/Q/SB/SW/	, ,					
USB/USW	(x)mm,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	(x)mm,r/m	1	2	0.5	P23	
PCMPEQQ	(x)mm,r/m	1	2	0.5	P23	SSE4.1
PCMPGTQ	(x)mm,r/m	1	2	0.5	P23	SSE4.2
PMULLW PMULHW	,					
PMULHUW PMULUDQ						
	(x)mm,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	(x)mm,r/m	1	4	1	P0	SSSE3
PMADDWD	(x)mm,r/m	1	4	1	P0	
PMADDUBSW	(x)mm,r/m	1	4	1	P0	
PAVGB/W	(x)mm,r/m	1	2	0.5	P23	
PMIN/MAX SB/SW/ SD						
UB/UW/UD	(x)mm,r/m	1	2	0.5	P23	
PHMINPOSUW	x,r/m	2	4	1	P1 P23	SSE4.1

PABSB/W/D	(x)mm,r/m	1	2	0.5	P23	SSSE3
PSIGNB/W/D	(x)mm,r/m	1	2	0.5	P23	SSSE3
PSADBW	(x)mm,r/m	2	4	1	P23	
MPSADBW	x,x,i	8	8	4	P1 P23	SSE4.1
0, 2	23,23,					AMD XOP
VPCOMB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
VI COMBIVIDIQ	χ,χ,χ,τι,,		_	0.0	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/	χ,χ,χ,τι,,		_	0.0	. 20	laterity our rott
WD/WQ/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPHADDUBW/BD/BQ/	7,70111	•	_	0.0	. 20	7 11115 7 (01
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	1	4	1	P0	AMD XOP
VPMACSSWW/WD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSSDD	x,x,x/m,x	1	5	2	P0	AMD XOP
VPMACSSDQH/L	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMADCSWD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMADCSSWD	x,x,x/m,x	1	4		P0	AMD XOP
VENIADOSSVVD	A,A,A/111,A	1	7	'	10	AMD AOI
Logic						
PAND PANDN POR						
PXOR	(x)mm,r/m	1	2	0.5	P23	
PSLL/RL W/D/Q	(20)11111,17111	•	_	0.0	. 20	
PSRAW/D	(x)mm,r/m	1	3	1	P1	
PSLL/RL W/D/Q	(20)11111,17111	•				
PSRAW/D	(x)mm,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
	71,71,72111			·		7 7
String instructions						
PCMPESTRI	x,x,i	27	16	10	P1 P2 P3	SSE4.2
PCMPESTRM	x,x,i	27	10	10	P1 P2 P3	SSE4.2
PCMPISTRI	x,x,i	7	13	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
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
	, ,					
1	l .	I .	1	I .	l .	1

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						
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	

			i ilodiivoi			
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		2	2	2	P1	
	y,y,y/m					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	
VMASKMOVPS/D	m128,x,x	18	~100	~90	P0 P1 P2 P3	
VMASKMOVPS/D	m256,y,y	34	~190	~180	P0 P1 P2 P3	
VIVIA COLIVIO VI CAB	111200, y, y		100	100	10111210	
Conversion						
CVTPD2PS	VV	2	8	1	P01	ivec/fp
VCVTPD2PS	X,X	4	7	2	P01	·
	x,y					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	x,y	4	7	2	P01	fp/ivec
CVTPI2PS	x,mm	2	8	1	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
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
HADDPS/D HSUBPS/D	x,x	3	10	2	P01 P1	ivec/fma
HADDPS/D HSUBPS/D VHADDPS/D	x,m	4		2	P01 P1	ivec/fma
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
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/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/		_		_		_
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	0.7	7	P01 P23	SSE4.1
VDPPS	y,y,y,i	25	27	13	P01 P3	SSE4.1
VDPPS	y,m,i	29	4.5	13	P01 P3	SSE4.1
DPPD	X,X,İ	15	15	5	P01 P23	SSE4.1
DPPD VFMADD132SS/SD	x,m,i	17 1	5-6	6	P01 P23 P01	SSE4.1 FMA3
VFMADD1329S/SD VFMADD132PS/PD	x,x,x/m x,x,x/m	1	5-6 5-6	1	P01	FMA3
VFMADD132PS/PD	y,y,y/m	2	5-6	1	P01	FMA3
All other FMA3 instruction		l	J 3-0	'	101	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		l	1		. • .	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	x,m	3	10	2	P01	AMD XOP

Logic AND/ANDN/OR/XORPS/						
PD	x,x/m	1	2	0.5	P23	ivec
VAND/ANDN/OR/XOR PS/PD	y,y,y/m	2	2	1	P23	ivec
Other						
VZEROUPPER		9		4	P2 P3	32 bit mode
VZEROUPPER		16		5	P2 P3	64 bit mode
VZEROALL		17		6	P2 P3	32 bit mode
VZEROALL		32		10	P2 P3	64 bit mode
LDMXCSR	m32	7		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 = 256 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.

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions				<u> </u>		
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	r,m	1	3	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	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	1	EX01	
XCHG	r32,r32	2		0.5	EX01	
XCHG	·	2		0.5		
ACIIG	r64,r64	2	'	0.5	EX01	Timing depends on
XCHG	r,m	2	~38	~38	EX01	Timing depends on hw
XLAT	1,111	2	6	2	LXUI	
PUSH	r	1	0	1		
PUSH	r :					
PUSH	i	1		1 1		
	m	2 8		4		
PUSHF(D/Q)						
PUSHA(D)		9		9		
POP	r	1		1		
POPE(D(O)	m	2		1		
POPF(D/Q)		34		19		
POPA(D)		14		8		
POP	sp	1	2		E)/04	
LEA	r16,[m]	2	2-3		EX01	any addr. size
LEA	r32,[m]	1	2		EX01	16 bit addr. size
	00/045			0.5	E)/04	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		
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		
Arithmetic instructions	S					

ADD, SUB			`	Steamhone	'	
ADD, SUB	ADD. SUB	r.r	1	1	0.5	FX01
ADD, SUB	1					
ADD, SUB ADD, SUB ADD, SUB ADC, SBB R, r, r 1	1					
ADD, SUB ADC, SBB r,r 1 1 1 7 1 EX01 ADC, SBB r,r 1 1 1 1 EX01 ADC, SBB r,r 1 1 1 1 EX01 ADC, SBB r,r 1 1 1 1 EX01 ADC, SBB r,m 1 1 9 1 EX01 ADC, SBB m,r 1 9 1 EX01 ADC, SBB m,r 1 9 1 EX01 ADC, SBB m,r 1 9 1 EX01 ADC, SBB m,r 1 1 9 1 EX01 CMP r,r 1 1 1 0.5 EX01 CMP r,r 1 1 1 0.5 EX01 CMP r,r 1 1 0.5 EX01 CMP r,r 1 1 0.5 EX01 CMP r,r 1 1 0.5 EX01 CMP r,r 1 1 0.5 EX01 CMP R,r 2 EX1 CMUL R,r 1 R,r 1 R,r 2 EX1 CMUL R,r 1	1			7		
ADC, SBB	·		-			
ADC, SBB					I	
ADC, SBB						
ADC, SBB	1				_	
ADC, SBB CMP CMP CMP r,r 1 1 0.5 EX01 CMP CMP r,m 1 0.5 EX01 CMP CMP r,m 1 0.5 EX01 CMP CMP r,m 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 1 0.5 EX01 INC, DEC, NEG m 1 7 AAA, AAS DAA DAA DAA BDAS AAD AAD AAM MUL, IMUL r8/m8 10 r1 r16/m16 2 EX1 MUL, IMUL r16,r16/m16 1 R64,r64/m64 IMUL r16,(r16),i r1 r10 r16,(r16),i r1 r10 r10 r10 r10 r10 r10 r10		r,m				
CMP CMP CMP CMP CMP CMP CMP CMP CMP CMP		m,r				
CMP	ADC, SBB	m,i	1		1	EX01
CMP	CMP	r,r	1	1	0.5	EX01
CMP	CMP	r,i	1	1	0.5	EX01
CMP	CMP	r,m	1		0.5	EX01
INC, DEC, NEG r	CMP	m,i	1		0.5	EX01
INC, DEC, NEG	INC. DEC. NEG		1	1		
AAA, AAS DAA DAS DAS AAD AAD AAD AAM AAM AAM AAM AAM AAM AAM						
DAA DAS AAD AAD AAM AAM MUL, IMUL r8/m8 1					•	2,101
DAS AAD AAD AAM MUL, IMUL R8/m8 1	1					
AAD AAM AAM AAM AAM BORNA AAM AAM AAM AAM AAM AAM AAM BORNA						
AAM 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 r32/m32 1 4 2 EX1 MUL, IMUL r16/m16 1 4 2 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 r64, r64/m64 1 6 4 EX1 IMUL r32, r32/m32 1 4 2 EX1 IMUL r32, r32/m32 1 4 2 EX1 IMUL r32, r32/m32 1 4 2 EX1 IMUL r32, r32/i, i						
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 r32,r32/m32 1 4 2 EX1 IMUL r64,r64/m64 1 6 4 EX1 IMUL r16,(r16),i 2 5 2 EX1 IMUL r64,(r64),i 1 6 4 EX1 IMUL r16,m16,i 2 2 EX1 IMUL r64,m64,i 2 2 EX1 DIV r8/m8 9 17-22 13-17 EX0 DIV r6/m64 2 13-70 13-70 EX0					45	
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 r32,(r32),i 1 4 2 EX1 IMUL r16,m16,i 2 2 EX1 IMUL r16,m16,i 2 2 EX1 IMUL r16,m64,i 2 2 EX1 DIV r8/m8 9 17-22 13-17 EX0 DIV r32/m32 2 13-39 EX0 DIV r64/m64 2 13-70 13-70 EX0 IDIV r64/m		0/ 0				E)//
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 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 DIV r8/m8 9 17-22 13-17 EX0 DIV r16/m16 7 15-25 15-25 EX0 DIV r64/m64 2 13-70 13-70 EX0 IDIV<						
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 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 DIV r8/m8 9 17-22 13-17 EX0 DIV r32/m32 2 13-39 13-39 EX0 IDIV r64/m64 2 13-70 13-70 EX0 IDIV r16/						
MUL r16,r16/m16 1 4 2 EX1 MUL r32,r32/m32 1 4 2 EX1 MUL r64,r64/m64 1 6 4 EX1 MUL r16,(r16),i 2 5 2 EX1 MUL r32,(r32),i 1 4 2 EX1 MUL r32,(r32),i 1 4 2 EX1 MUL r64,(r64),i 1 6 4 EX1 MUL r16,m16,i 2 2 2 EX1 MUL r16,m16,i 2 2 2 EX1 MUL r32,m32,i 2 2 2 EX1 MUL r64,m64,i 2 4 EX1 MUL r64,m64 2 13-17 EX0 MUV r16/m16 7 15-25 15-25 EX0 MUV r32/m32 2 13-39 13-39 EX0 MUV r64/m64 2 13-70 13-70 EX0 MUV r16/m16 7 14-25 14-24 EX0 MUV r32/m32 2 13-39 13-39 EX0 MUV r32/m32 2 13-70 13-70 EX0 MUV R64/m64 2 13-70 13-70 EX						
MUL	MUL, IMUL	r64/m64	1			
MUL	IMUL	r16,r16/m16	1	4		EX1
IMUL	IMUL	r32,r32/m32	1	4	2	EX1
IMUL	IMUL	r64,r64/m64	1	6	4	EX1
IMUL	IMUL	r16,(r16),i	2	5	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 DIV r8/m8 9 17-22 13-17 EX0 DIV r16/m16 7 15-25 EX0 DIV r32/m32 2 13-39 EX0 DIV r64/m64 2 13-70 EX0 IDIV r8/m8 9 17-22 13-17 EX0 IDIV r16/m16 7 14-25 14-24 EX0 IDIV r16/m16 7 14-25 14-24 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 CWD 2 1 1	IMUL		1	4	2	EX1
IMUL	IMUL	` '	1	6	4	
IMUL r32,m32,i 2 4 EX1 EX1 DIV r8/m8 9 17-22 13-17 EX0 DIV r16/m16 7 15-25 15-25 EX0 DIV r64/m64 2 13-39 13-39 EX0 DIV r64/m64 2 13-70 13-70 EX0 IDIV r16/m16 7 14-25 14-24 EX0 IDIV r32/m32 2 13-39 13-39 EX0 IDIV r16/m16 7 14-25 14-24 EX0 IDIV r32/m32 2 13-39 13-39 EX0 IDIV r32/m32 2 13-70 13-70 EX0 IDIV r32/m32 2 13-70 13-70 EX0 EX0 IDIV r64/m64 2 13-70 13-70 EX0 EX0 EX01		` '				
IMUL r64,m64,i 2						
DIV r8/m8 9 17-22 13-17 EX0 DIV r16/m16 7 15-25 15-25 EX0 DIV r32/m32 2 13-39 13-39 EX0 DIV r64/m64 2 13-70 13-70 EX0 IDIV r8/m8 9 17-22 13-17 EX0 IDIV r16/m16 7 14-25 14-24 EX0 IDIV r32/m32 2 13-39 13-39 EX0 IDIV r64/m64 2 13-70 13-70 EX0 CBW, CWDE, CDQE 1 1 EX01 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 7 1 EX01						
DIV r16/m16 7 15-25 EX0 DIV r32/m32 2 13-39 13-39 EX0 DIV r64/m64 2 13-70 13-70 EX0 IDIV r8/m8 9 17-22 13-17 EX0 IDIV r16/m16 7 14-25 14-24 EX0 IDIV r32/m32 2 13-39 13-39 EX0 IDIV r64/m64 2 13-70 13-70 EX0 CBW, CWDE, CDQE 1 1 EX01 EX01 CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 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 7 1 EX01 AND, OR, XOR m,r 1 7 1 EX01 AN				17-22		
DIV r32/m32 2 13-39 13-39 EX0 DIV r64/m64 2 13-70 13-70 EX0 IDIV r8/m8 9 17-22 13-17 EX0 IDIV r16/m16 7 14-25 14-24 EX0 IDIV r32/m32 2 13-39 13-39 EX0 IDIV r64/m64 2 13-70 13-70 EX0 CBW, CWDE, CDQE 1 1 EX01 EX01 CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 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 1 EX01 AND, OR, XOR m,i 1 7 1 EX01						
DIV r64/m64 2 13-70 13-70 EX0 r8/m8 9 17-22 13-17 EX0 IDIV r16/m16 7 14-25 14-24 EX0 IDIV r32/m32 2 13-39 13-39 EX0 IDIV r64/m64 2 13-70 13-70 EX0						
IDIV r8/m8 9 17-22 13-17 EX0 IDIV r16/m16 7 14-25 14-24 EX0 IDIV r32/m32 2 13-39 13-39 EX0 IDIV r64/m64 2 13-70 13-70 EX0						
IDIV						
IDIV r32/m32 2 13-39 13-39 EX0						
IDIV CBW, CWDE, CDQE 1						
CBW, CWDE, CDQE 1 1 0.5 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 1 EX01 AND, OR, XOR m,i 1 7 1 EX01						
CDQ, CQO 1 1 0.5 EX01 CWD 2 1 1 EX01 Logic instructions		r64/m64			13-70	
CWD 2 1 1 EX01 Logic instructions			-			
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 1 EX01 AND, OR, XOR m,i 1 7 1 EX01 AND, OR, XOR m,i 1 7 1 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 1 EX01 AND, OR, XOR m,i 1 7 1 EX01	CWD		2	1	1	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 1 EX01 AND, OR, XOR m,i 1 7 1 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 1 EX01			_	_	2.5	EV04
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			-			
AND, OR, XOR m,r 1 7 1 EX01 AND, OR, XOR m,i 1 7 1 EX01			-	1		
AND, OR, XOR m,i 1 7 1 EX01			-			
		m,r	1			
TEST		m,i				
	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	
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			7		EX01	
	r,i	15				
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	
			I.			
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	
			2	3		
BSF	r,r	6	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		
STD		2		4		
POPCNT	r16/32,r16/32	1	4	2		SSE4.2
POPCNT	r64,r64		4			SSE4.2
		1		4	5 1/0	
LZCNT	r,r	1	2	2	EX0	LZCNT
TZCNT	r,r	2	2	2		BMI1
BEXTR	r,r,r	2	2	1		BMI1
BEXTR	r,r,i	2	2	1		AMD TBM
BLSI	r,r	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
BLCI	r,r	2	2	1		AMD TBM
BLCIC	r,r	2	2	1		AMD TBM
BLCMSK		2	2			AMD TBM
	r,r			1		
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		2	2	1		AMD TBM
	r,r					
TZMSK	r,r	2	2	1		AMD TBM

Control transfer instru	ctions]
JMP	short/near	1 1		2	EX1	
JMP	r	1 1		2	EX1	
JMP				2	EX1	
	m					0 16 1
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
CALL	near	2		2	EX1	
CALL	r	2		2	EX1	
CALL	m	3		2	EX1	
RET		1		2	EX1	
RET	i	4		2	EX1	
BOUND	m	11		5	LXI	for no jump
	111			5 2		
INTO		4		2		for no jump
String instructions						
LODS		3		3		
REP LODS	m8/m16	6n		3n		
REP LODS	m32/m64	6n		2.5n		
STOS	11102/11104	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		
Complementies						
Synchronization LOCK ADD			- 20			
	m,r	1 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			
LOOK GIVIF ACTIG TOD	111120	4	-60			
Other						
NOP (90)		1		0.25	none	
Long NOP (0F 1F)		1		0.25	none	
PAUSE		8		4		
ENTER	a,0	13		21		
ENTER	a,b	11+5b		20-30		
·	u,b	11.00		_0 00		1

LEAVE		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							
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	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	 						
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	

I	İ	ĺ			ĺ
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	4	1	P2	
MOVD	mm/x, r32/64	2	5	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	xmm,xmm	1	0	0.25	none	inherit domain
MOVDQA	xmm,m	1	2	0.5		
MOVDQA	m,xmm	1	3	1	P2	
VMOVDQA	ymm,ymm	2	2	0.5	P02	
VMOVDQA	ymm,m256	2	3	1		
VMOVDQA	m256,ymm	2	4	1	P2	
MOVDQU	xmm,xmm	1	0	0.25	none	inherit domain
MOVDQU	xmm,m	1	2	0.5		
MOVDQU	m,xmm	1	3	1	P2	
LDDQU	xmm,m	1	2	0.5		
VMOVDQU	ymm,m256	2	3	1		
VMOVDQU	m256,ymm	2	4	1		
MOVDQ2Q	mm,xmm	1	1	0.5	P02	
MOVQ2DQ	xmm,mm	1	1	0.5	P02	
MOVNTQ	m,mm	1	3	1	P2	
MOVNTDQ	m,xmm	1	3	1	P2	
MOVNTDQA	xmm,m	1	2	0.5		

PACKUSWB/DW (x)mm,r/m							
PUNPCKH/LBW/WD/D Q	PACKSSWB/DW	(x)mm,r/m	1	2	1	P1	
Q	PACKUSWB	(x)mm,r/m	1	2	1	P1	
PUNPCKHQDQ	PUNPCKH/LBW/WD/D						
PUNPCKLQDQ	Q	(x)mm,r/m	1		1	P1	
PSHUFB	PUNPCKHQDQ	xmm,r/m	1		1	P1	
PSHUFD	PUNPCKLQDQ	xmm,r/m	1	2	1	P1	
PSHUFU/HW	PSHUFB	(x)mm,r/m	1	3	1	P1	
PSHUFL/HW	PSHUFD	xmm,xmm,i	1	2	1	P1	
PALIGNR	PSHUFW	mm,mm,i	1	2	1	P1	
PBLENDW	PSHUFL/HW	xmm,xmm,i	1		1	P1	
MASKMOVQ	PALIGNR	(x)mm,r/m,i	1	2	1	P1	
MASKMOVDQU	PBLENDW	xmm,r/m	1	2	0.5	P02	SSE4.1
PMOVMSKB	MASKMOVQ	mm,mm	31	32	16	P2	
PEXTRB.W/D/Q	MASKMOVDQU	xmm,xmm	65	45	31	P0 P1 P2	
PINSRB/W/D/Q	PMOVMSKB	r32,mm/x	2	5	1	P1 P2	
EXTRQ	PEXTRB/W/D/Q	r,x/mm,i	2	5	1	P1 P2	SSE4.1
EXTRQ	PINSRB/W/D/Q	x/mm,r,i	2	6	1	P1	
INSERTQ	EXTRQ	x,i,i	1	3	1	P1	AMD SSE4A
INSERTQ	EXTRQ	x,x	1	1	1	P1	AMD SSE4A
PMOVSXBW/BD/BQ/ WD/WQ/DQ	INSERTQ	x,x,i,i	1	1	1	P1	AMD SSE4A
WD/WQ/DQ	INSERTQ	x,x	1	1	1	P1	AMD SSE4A
PMOVZXBW/BD/BQ/W D/WQ/DQ	PMOVSXBW/BD/BQ/						
D/WQ/DQ	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 Arithmetic instructions PADDB://IVID/Q/SB/SW// USB/USW/ USB/USW (x)mm,r/m 1 2 0.5 P02 PSUBB/W/D/Q/SB/SW// USB/USW/ (x)mm,r/m 1 2 0.5 P02 PHADD/SUB(S)W/D x,x 3 5 2 P02 2P1 SSE3 PCMPEQ/GT B/W/D (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPEQQ (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULDQ (x)mm,r/m 1 4 1 P0 SSE4.1 PMADDWD (x)mm,r/m 1 4 1	PMOVZXBW/BD/BQ/W						
VPCMOV VPPERM y,y,y/m x,x,x,x/m 2 2 2 P1 AMD XOP AMD XOP Arithmetic instructions PADDB/W/D/Q/SB/SW/ USB/USW (x)mm,r/m 1 2 0.5 P02 PSUBB/W/D/Q/SB/SW/ USB/USW (x)mm,r/m 1 2 0.5 P02 PHADD/SUB(S)W/D PCMPEQ/GT B/W/D PCMPEQQ x,x 3 5 2 P02 2P1 SSSE3 PCMPEQQ (x)mm,r/m 1 2 0.5 P02 P02 PCMPEQQ (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULHUW PMULUDQ (x)mm,r/m 1 4 1 P0 SSE4.1 PMULDQ x,r/m 1 4 1 P0 SSE4.1 PMULHRSW (x)mm,r/m 1 4 1 P0 SSE3 PMADDUBSW (x)mm,r/m 1 4 1 P0 P02 PHMINPOSUW	D/WQ/DQ	x,x	1		1	P1	SSE4.1
VPPERM	VPCMOV	x,x,x,x/m				P1	AMD XOP
Arithmetic instructions	VPCMOV	y,y,y,y/m	2		2	P1	AMD XOP
PADDB/W/D/Q/SB/SW/USB/USW USB/USW USB/USW (x)mm,r/m 1 2 0.5 P02 PSUBB/W/D/Q/SB/SW/USB/USW (x)mm,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 (x)mm,r/m 1 2 0.5 P02 PCMPEQQ (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULUDQ (x)mm,r/m 1 4 1 P0 PMULD x,r/m 1 4 1 P0 SSE4.1 PMULDQ x,r/m 1 4 1 P0 SSE4.1 PMULHRSW (x)mm,r/m 1 4 1 P0 SSE4.1 PMULHRSW (x)mm,r/m 1 4 1 P0 SSSE3 PMADDWD (x)mm,r/m 1 4 1 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 PMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P	VPPERM	x,x,x,x/m	1	2	1	P1	AMD XOP
PADDB/W/D/Q/SB/SW/USB/USW USB/USW USB/USW (x)mm,r/m 1 2 0.5 P02 PSUBB/W/D/Q/SB/SW/USB/USW (x)mm,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 (x)mm,r/m 1 2 0.5 P02 PCMPEQQ (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULUDQ (x)mm,r/m 1 4 1 P0 PMULD x,r/m 1 4 1 P0 SSE4.1 PMULDQ x,r/m 1 4 1 P0 SSE4.1 PMULHRSW (x)mm,r/m 1 4 1 P0 SSE4.1 PMULHRSW (x)mm,r/m 1 4 1 P0 SSSE3 PMADDWD (x)mm,r/m 1 4 1 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 PMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P							
USB/USW		3					
PSUBB/W/D/Q/SB/SW/USB/USW		(-)	_		0.5	Doo	
USB/USW (x)mm,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 (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPEQQ (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULHW PMULHUW PMUL		(x)mm,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 (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPEQQ (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULUDQ (x)mm,r/m 1 4 1 P0 PMULDQ x,r/m 1 4 1 P0 PMULHRSW (x)mm,r/m 1 4 1 P0 SSE4.1 PMADDWD (x)mm,r/m 1 4 1 P0 SSSE3 PMADDUBSW (x)mm,r/m 1 4 1 P0 P0 PMIN/MAX SB/SW/ SD (x)mm,r/m 1 2 0.5 P02 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 S		()	_	_	0.5	DOO	
PCMPEQ/GT B/W/D (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPEQQ (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULUDQ (x)mm,r/m 1 4 1 P0 PMULDD x,r/m 1 5 2 P0 SSE4.1 PMULDQ x,r/m 1 4 1 P0 SSE4.1 PMULHRSW (x)mm,r/m 1 4 1 P0 SSSE3 PMADDWD (x)mm,r/m 1 4 1 P0 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 P0 PMIN/MAX SB/SW/ SD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3		` '	_	_	_		00050
PCMPEQQ (x)mm,r/m 1 2 0.5 P02 SSE4.1 PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULUDQ (x)mm,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 (x)mm,r/m 1 4 1 P0 SSSE3 PMADDWD (x)mm,r/m 1 4 1 P0 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3	, ,						555E3
PCMPGTQ (x)mm,r/m 1 2 0.5 P02 SSE4.2 PMULLW PMULHW PMULUDQ (x)mm,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 (x)mm,r/m 1 4 1 P0 SSSE3 PMADDWD (x)mm,r/m 1 4 1 P0 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 P02 PHMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02			•				CCE4.4
PMULLW PMULHW PMULUDQ			-				
PMULHUW PMULUDQ		(X)[[[[[],[/[[]	I		0.5	P02	33E4.2
Name							
PMULLD x,r/m 1 5 2 P0 SSE4.1 PMULDQ x,r/m 1 4 1 P0 SSE4.1 PMULHRSW (x)mm,r/m 1 4 1 P0 SSSE3 PMADDWD (x)mm,r/m 1 4 1 P0 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 PMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3	I WOLITOW I WOLODQ	(y)mm r/m	1	4	1	PΛ	
PMULDQ x,r/m 1 4 1 P0 SSE4.1 PMULHRSW (x)mm,r/m 1 4 1 P0 SSSE3 PMADDWD (x)mm,r/m 1 4 1 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 PMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3	PMIIIID	. ,					SSF4 1
PMULHRSW (x)mm,r/m 1 4 1 P0 SSSE3 PMADDWD (x)mm,r/m 1 4 1 P0 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 PMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3							
PMADDWD (x)mm,r/m 1 4 1 P0 PMADDUBSW (x)mm,r/m 1 4 1 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 PMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3		· ·					
PMADDUBSW PAVGB/W (x)mm,r/m 1 4 1 P0 PAVGB/W (x)mm,r/m 1 2 0.5 P02 PMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3			-				000L0
PAVGB/W (x)mm,r/m 1 2 0.5 P02 PMIN/MAX SB/SW/ SD UB/UW/UD PHMINPOSUW (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3			-				
PMIN/MAX SB/SW/ SD UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3		` '	-		· •		
UB/UW/UD (x)mm,r/m 1 2 0.5 P02 PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3		(^);;;;;;;;	'	_	0.5	1 02	
PHMINPOSUW x,r/m 2 4 1 P1 P02 SSE4.1 PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3		(x)mm r/m	1	2	0.5	P02	
PABSB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3 PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3		` '					SSE4.1
PSIGNB/W/D (x)mm,r/m 1 2 0.5 P02 SSSE3		· ·			· ·		

MPSADBW	x,x,i	8	8	4	P1 P02	SSE4.1
	, ,					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	x,x,x/m,i	1	2	0.5	P02	latency 0 if i=6,7
VPHADDBW/BD/BQ/						
WD/WQ/DQ	x,x/m	1	2	0.5	P02	AMD XOP
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
VPMACSDQH/L	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSSWW/WD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMACSSDD	x,x,x/m,x	1	5	2	P0	AMD XOP
VPMACSSDQH/L	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMADCSWD	x,x,x/m,x	1	4	1	P0	AMD XOP
VPMADCSSWD	x,x,x/m,x	1	4	1	P0	AMD XOP
Logic						
PAND PANDN POR						
PXOR	(x)mm,r/m	1	2	0.5	P02	
PSLL/RL W/D/Q						
PSRAW/D	(x)mm,r/m	1	3	1	P1	
PSLL/RL W/D/Q						
PSRAW/D	(x)mm,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/111,1 x,x,x,i	7	11	7	P1	pclmul
PCLMULQDQ	x,x,x,i x,x,m,i	8	''	7	P1	pclmul
AESDEC	X,X,111,1 X,X	2	5	1	P01	aes
AESDECLAST	X,X X,X	2	5	1	P01	aes
AESENC	X,X X,X	2	5	1	P01	aes
AESENCLAST	x,x X,X	2	5	1	P01	aes
AESIMC	x,x X,X	1	5	1	P01	aes
AESKEYGENASSIST	x,x x,x,i	1	5	1	P0	aes
ALUNE I GENAGOIO I	۸,۸,۱	'	3	1	FU	acs
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	P02	ivec
MOVAPS/D						
MOVUPS/D	x,m128	1	2	0.5		
VMOVAPS/D						
VMOVUPS/D	y,m256	2	2	1		
MOVAPS/D						
MOVUPS/D	m128,x	1	3	1	P2	
VMOVAPS/D	m256,y	2	3	2	P2	
VMOVUPS/D	m256,y	2	3	2	P2	
MOVSS/D	X,X	1	2	0.5	P01	fp
MOVSS/D	x,m32/64	1	2	0.5		
MOVSS/D	m32/64,x	1	3	1	P2	
MOVHPS/D	x,m64	1	3	1	P1	
MOVLPS/D	x,m64	1	3	0.5	P01	
MOVHPS/D	m64,x	2	4	1	P1 P2	
MOVLPS/D	m64,x	1	3	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	ľ
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	у,у	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	y,111120 X,X	1	2	1	P1	ivec
MOVSH/LDUP	x,m128	1	_	0.5		1000

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	
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	
11111101111101111011	200,,,,					
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	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	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
CVT(1)55251	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		4	7	2	P0 P1	F16C
VCVTP32PH VCVTPH2PS	x/m,y,i	2	7	2	P0 P1	F16C
	x,x/m	4	7	2	P0 P1	
VCVTPH2PS	y,x/m	4	/		PUPI	F16C
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
VADDDO (D. VOLIDDO (D	,,l	_	5 0		D04	£
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

		1	1			
HADDPS/D HSUBPS/D	X,X	4	10	2	P0 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	9-17	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
VDIVPD	y,y,y/m	2	9-32	9-27	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	3,3,3				-	I*
UCOMISS/D	x,x/m	2		1	P01 P2	fp
MAXSS/SD/PS/PD	- 1,5 4 1 1	_			.	٠٣
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/						-
PD	y,y/m,i	2	4	2	P0	fp
DPPS	x,x,i	9	25	4	P0 P1	SSE4.1
DPPS	x,m,i	10		5	P0 P1	SSE4.1
VDPPS	y,y,y,i	13	25	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,m,i	8		4	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	y,y,y/m	2	5-6	1	P01	FMA3
All other FMA3 instruction		bove	I			FMA3
VFMADDSS/SD	x,x,x,x/m	1 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		bove	I			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
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						
AND/ANDN/OR/XORPS/						
PD	x,x/m	1	2	0.5	P02	ivec

VAND/ANDN/OR/XOR PS/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	
FXSAVE	m4096	59-67		78	P0 P1 P2	
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 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 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 1/2 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	IO/1	
MOV	r,i	1		0.5	IO/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	IO/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	IO/1	
CMOVcc	r,m	1		1		
XCHG	r,r	2	1	1	10/1	
XCHG	r,m	3	20			Timing dep. on hw

			Вов	J		
XLAT		2	5			
PUSH	r	1		1		
PUSH	i	1		1		
PUSH	m	3		2		
PUSHF(D/Q)		9		6		
PUSHA(D)		9		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					10/1	1 -
I	r32/64,[m]	1	1	0.5		no scale, no offset
LEA	r32/64,[m]	1	2-4	1	10	w. scale or offset
LEA	r64,[m]	1		0.5	10/1	RIP relative
LAHF		4	4	2		
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 m	1		1	AGU	AMD only
SFENCE	'''	4		~45	AGU	AND OHIY
LFENCE		1		1	AGU	
MFENCE		4		~45	AGU	
Arithmetic instructions	S					
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	I0/1	
CMP		1		1	10/1	
INC, DEC, NEG	r,m	1	4	0.5	10/1	
	r		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	
II		2				
IMUL	r16,(r16),i		4	3	10	
IMUL	r32,(r32),i	1	3	1	10	
IMUL	r64,(r64),i	1	7	4	10	
DIV	r8/m8	1	27	27	10	

DIV	r16/m16	1	33	33	10	
DIV	r32/m32	1	49	49	10	
		_				
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		10/1	
CWD, CDQ, CQC		'	· •		10/1	
Logic instructions						
AND, OR, XOR	rr	1	1	0.5	I0/1	
	r,r		I		10/1	
AND, OR, XOR	r,m	1		1		
AND, OR, XOR	m,r	1		1		
TEST	r,r	1	1	0.5	I0/1	
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				1	10/1	
	r,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		7	4	4		
· ·	r,r,cl					
SHLD, SHRD	m,r,i/CL	8	18	15		
BT	r,r/i	1		0.5		
ВТ	m,i	1		1		
ВТ	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		
втс	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
				J		
LZCNT	r,r/m	8	5	0.5		SSE4.A, AMD only
SETcc	r	1	1	0.5		
SETcc	m	1		1		
CLC, STC		1		0.5	I0/1	
CMC		1	1	0.5	10/1	
CLD		1		1	10	
STD		2		2	10,11	
1		I .	1	t .	· *	ı

Control transfer instru	ictions					
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		
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

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	

1		1		1	1
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	20	2	FP0
FLDCW					
FLDCVV	m16	12		10	FP1
Arithmetic instructions	5				
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	FP0,FP1
FMUL(P)	r	1	5	3	FP1
FMUL(P)	·	1	5	3	FP1
` '	m m	2	5	J	FP1
FIMUL	m	1	40	40	FP1
FDIV(R)(P)	r		19	19	
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
I I IXLIVII		'	11-13		
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
				49-103	
FSCALE		5	8		FP0, FP1
FXTRACT		7	9		FP0, FP1
F2XM1		30-56	~60		FP0, FP1
FYL2X		8	29		FP0, FP1
FYL2XP1		12	44		FP0, FP1
Other					
FNOP		1	0	0.5	FP0, FP1
(F)WAIT		1	0	0.5	ALU
FNCLEX		9		30	FP0, FP1
FNINIT		26		78	FP0, FP1
FNSAVE	m	85		163	FP0, FP1
INSAVE	m	ု ၀၁	l	103	FFU, FFI

FRSTOR	m	80	123	FP0, FP1	
FXSAVE	m	71	105	FP0, FP1	
FXRSTOR	m	111	118	FP0, FP1	

Integer MMX and XMM instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes
Move instructions				<u> </u>		
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,(x)mm	1	6	2	FP1	
	- ,()					Moves 64 bits.
MOVD (MOVQ)	r64,(x)mm	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 1	0.5	FP0/1	
MOVQ	xmm,xmm	2	1 1	1	FP0/1	
MOVQ	mm,m64	1	5	1	FP0/1	
MOVQ	xmm,m64	2	5	1	FP1	
MOVQ	m64,(x)mm	1	6	2	FP1	
MOVDQA	xmm,xmm	2	1	1	FP0/1	
MOVDQA	xmm,m	2	6	2	AGU	
MOVDQA	m,xmm	2	6	3	FP1	
MOVDQU, LDDQU	xmm,m	2	6-9	2-5.5	AGU	
MOVDQU		2	6-9	2-3.5 3-6	FP1	
MOVDQ2Q	m,xmm	1		0.5	FP0/1	
MOVQ2DQ	mm,xmm	2	1 1	0.5	FP0/1	
MOVNTQ	xmm,mm		13	-	FP1	
	m,mm	1		1,5		
MOVNTDQ	m,xmm	2	13	3	FP1	
PACKSSWB/DW	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	1	4	0.5	ED0/4	
PACKUSWB	mm,r/m	1	1	0.5	FP0/1	
PACKSSWB/DW PACKUSWB				0	ED0/4	
	xmm,r/m	3	2	2	FP0/1	
PUNPCKH/LBW/WD/D Q	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	1	4	0.5		
	mm,r/m	1	1	0.5		
PUNPCKH/LBW/WD/D	vmm r/m		4	1		
Q DUNDOKHODO	xmm,r/m	2	1 1	1	EDO ED4	
PUNPCKHQDQ	xmm,r/m	2	1 1	1	FP0, FP1	
PUNPCKLQDQ	xmm,r/m	1	1	0.5	FP0/1	0
PSHUFB	mm,mm	1	2	1	FP0/1	Suppl. SSE3
PSHUFB	xmm,xmm	6	3	3	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	
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,(x)mm	1	8	2	FP0	

					1	
PEXTRW	r32,(x)mm,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
LXTIC	XIIIII,XIIIIII,I,I	•	_	_	11 0/1	OOL+.7 (, 7 (IVID OTH)
Arithmetic instruction						
PADDB/W/D/Q	5					
PADDSB/W						
PADDUSB/W						
PSUBB/W/D/Q						
PSUBSB/W						
PSUBUSB/W	mm,r/m	1	1	0.5	FP0/1	
PADDB/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						
PMULUDQ	xmm,r/m	2	2	2	FP0	
PMULHRSW	mm,r/m	1	2	1	FP0	Suppl. SSE3
PMULHRSW	xmm,r/m	2	2	2	FP0	Suppl. SSE3
PMADDWD	mm,r/m	1	2	1	FP0	
PMADDWD	xmm,r/m	2	2	2	FP0	
PMADDUBSW	mm,r/m	1	2	1	FP0	Suppl. SSE3
PMADDUBSW	xmm,r/m	2	2	2	FP0	Suppl. SSE3
PAVGB/W	mm,r/m	1	1	0.5	FP0/1	очри оо <u>г</u> о
PAVGB/W	xmm,r/m	2	1	1	FP0/1	
PMIN/MAX SW/UB	mm,r/m	1	1	0.5	FP0/1	
PMIN/MAX SW/UB	xmm,r/m	2	1	1	FP0/1	
PABSB/W/D	mm,r/m	1	1	0.5	FP0/1	Suppl. SSE3
PABSB/W/D	xmm,r/m	2	1	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	FP0/1	Suppl. SSE3
PSADBW	mm,r/m	1	2	2	FP0/1	ουμμι. σοπο
PSADBW	·	2	2	2	FP0, FP1	
IL OUDIN	xmm,r/m	2			FFU, FF1	
Logio						
Logic						
PAND PANDN POR PXOR	mm r/m	4	1	0.5	FP0/1	
	mm,r/m	1	1	0.5	FFU/ I	
PAND PANDN POR	vmm r/m	2	4	4	EDO/4	
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 PSRAW/D PSLLDQ, PSRLDQ	xmm,i/xmm/m xmm,i	2 2	1 1	1 1	FP0/1 FP0/1	
Other EMMS		1		0.5	FP0/1	

Floating point XMI	Floating 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											
	r,r	1	1	0.5	FP0/1						
MOVHPS/D,											
MOVLPS/D	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,											
MOVSLDUP	r,r	2	1	1	FP0/1						
MOVSHDUP,											
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						
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	3	12	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	
ADDPS/D SUBPS/D	r,r/m	2	3	2	FP0	
ADDSUBPS/D	r,r/m	2	3	2	FP0	SSE3
HADDPS/D						
HSUBPS/D	r,r/m	2	3	2	FP0	SSE3
MULSS	r,r/m	1	2	1	FP1	
MULSD	r,r/m	1	4	2	FP1	
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						
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	
RSQRTPS	r,r/m	2	3	2	FP1	
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 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 1/2 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 pipeline.

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 dif-

ferent execution pipes.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes
Move instructions						
MOV	r,r	1	1	0.5	I0/1	
MOV	r,i	1		0.5	10/1	
MOV	r8/16,m	1	4	1	AGU	Any addressing mode
MOV	m,r8/16	1	4	1	AGU	Any addressing mode
MOV	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		

			9-			
MOVSXD	r64,m32	1	3	1		
CMOVcc	r,r	1	1	0.5	10/1	
CMOVcc	r,m	1		1	1071	
I			_		10/4	
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		
		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		
	OI .			18		
POPF(D/Q)		29				
POPA(D)		9		8		
LEA	r16,[m]	2	3	2	10	Any address size
LEA	r32/64,[m]	1	1	0.5	I0/1	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	, []	4	3	2	10, 1	1
SAHF		1	1	0.5	10/1	
I					10/1	
SALC		1	1	1	10/4	
BSWAP	r	1	1	0.5	10/1	
MOVBE	r,m	1		1		MOVBE
MOVBE	m,r	1		1		MOVBE
PREFETCHNTA	m	1		~100	AGU	
PREFETCHT0/1/2	m	1		~100	AGU	
PREFETCHW	m	1		~100	AGU	
LFENCE		1		0.5	AGU	
MFENCE		4		~45	AGU	
SFENCE		4		~45	AGU	
Arithmetic instructions				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	6	1		
ADC, SBB	r,r/i	1	1	1	I0/1	
ADC, SBB	r,m	1		1		
ADC, SBB	m,r/i	1	8			
CMP	r,r/i	1	1	0.5	10/1	
CMP		1	'	1	15/1	
I	r,m	1	4		10/1	
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		
1 ***	l	1			I	1

			2 - 9 -			
MUL, IMUL	r8/m8	1	3	1	10	
MUL, IMUL	r16/m16	3	3	3	10	
MUL, IMUL	r32/m32	2	3	2	10	
MUL, IMUL	r64/m64	2	6	5	10	
IMUL	r16,r16/m16	1	3	1	10	
1	· ·		3	-		
IMUL	r32,r32/m32	1		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	
IDIV	r16/m16	2	12-19	12-19	10	
IDIV		2	12-19	12-19		
	r32/m32				10	
IDIV	r64/m64	2	12-43	12-43	10	
CBW, CWDE, CDQE		1	1		10/1	
CWD, CDQ, CQO		1	1		I0/1	
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			4	0.5	I0/1	Divili
	r,i	1	1			
TEST	r,r	1	1	0.5	I0/1	
TEST	r,m	1		1		
NOT	r	1	1	0.5	I0/1	
NOT	m	1	6	1		
SHL, SHR, SAR	r,i/CL	1	1	0.5	I0/1	
ROL, ROR	r,i/CL	1	1	0.5	10/1	
RCL, RCR	r,1	1	1	1	I0/1	
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,	.,02	•				
ROR	m,i /CL	1	6	1		
RCL, RCR	m,1	1		1		
		10				
RCL	m,i			11		
RCR	m,i	9		11		
RCL	m,CL	9		11		
RCR	m,CL	8		11		
SHLD, SHRD	r,r,i	6	3	3		
SHLD, SHRD	r,r,cl	7	4	4		
SHLD, SHRD	m,r,i/CL	8		11		
ВТ	r,r/i	1		0.5		
ВТ	m,i	1		1		
BT	m,r	5		3		
ı-·	''','	J	l	1	İ	I I

			9-			
BTC, BTR, BTS	r,r/i	2	2	1		
BTC	m,i	5		11		
BTR, BTS	m,i	4		11		
BTC, BTR, BTS	m,r	8		11		
BSF	r,r	7	4	4		
BSR	r,r	8	4	4		
BSF, BSR	r,m	8		4		
POPCNT	r,r/m	1	1	0.5		SSE4A/SSE4.2
LZCNT	r,r	1	1	0.5		SSE4A/LZCNT
		1				
TZCNT	r,r	2	2	1		BMI1
BLSI BLSR	r,r	2	2	1		BMI1
BLSI BLSR	r,m	3		2		BMI1
BLSMSK	r,r	2	2	1		BMI1
BLSMSK	r,m	3		2		BMI1
BEXTR	r,r,r	1	1	0.5		BMI1
BEXTR		2	'	1		BMI1
	r,m,r		4			DIVILI
SETcc	r	1	1	0.5		
SETcc	m	1		1		
CLC, STC		1		0.5	IO/1	
CMC		1 1	1		10/1	
CLD		1		1	10	
STD		2		2	10,11	
010				_	10,11	
0 1 4 5 1 1	- 4.9					
Control transfer instru		↓ .		_		
JMP	short/near	1		2		
JMP	r	1		2		
JMP	m(near)	1		2		
Jcc	short/near	1 1		0.5 - 2		2 if jumping
J(E/R)CXZ	short	2		1 - 2		2 if jumping
LOOP	short	8		5		2 ii jaiiipiiig
LOOPE LOOPNE	short	10		6		
CALL	near	2		2		
CALL	r	2		2		
CALL	m(near)	5		2		
RET		1 1		3		
RET	i	4		3		
1121		"		J		values are for no
BOLIND	m	8		4		
BOUND	m			4		jump
						values are for no
INTO		4		2		jump
String instructions						
LODS		4		2		
REP LODS		~5n		~3n		
STOS		4		2		
						for anall a
REP STOS		~2n		~n		for small n
REP STOS		2/16B		1/16B		best case
MOVS		7		4		
REP MOVS		~2n		~1.5n		for small n
REP MOVS		2/16B		1/16B		best case
SCAS		5		3		
REP SCAS		~6n		~3n		
		1				
CMPS		7		4		

REP CMPS		~6n		~3n		
Synchronization						
LOCK ADD	m,r	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 throughput	Execution pipe	Notes
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	

			ŭ			
FLDCW	m16	12		9	FP1	
Arithmetic instructions	 e					
FADD(P),FSUB(R)(P)	r	1	3	1	FP0	
1		1	3	1	FP0	
FADD(P),FSUB(R)(P)	m	2		2		
FIADD,FISUB(R)	m		_	3	FP0,FP1	
FMUL(P)	r	1	5		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	
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 m	80	136-150	136-130	FP0, FP1	
FROIUR	m	00	130	130	FFU, FFI	

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, 132 x,m32	1	4	1	AGU	
MOVD	m32,(x)mm	1	3	1	FP1	
MOVD	11102,(X)111111	'		ı	111	Moves 64 bits.Name
MOVD / MOVQ	r64,(x)mm	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	mm,mm	1	1	0.5	FP0/1	
MOVQ	x,x	1	1	0.5	FP0/1	
MOVQ	(x)mm,m64	1	4	1	AGU	
MOVQ	m64,(x)mm	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	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	0.5	FP0/1	
PACKSSWB/DW						
PACKUSWB	x,r/m	1	2	0.5	FP0/1	
PUNPCKH/LBW/WD/D						
Q	mm,r/m	1	1	0.5	FP0/1	
PUNPCKH/LBW/WD/D			_			
Q	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	0.5	FP0/1	
PSHUFL/HW	x,x,i	1	1	0.5	FP0/1	0
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	
MASKMOVDQU	X,X	64	43-2210	34	FP0, FP1	
PMOVMSKB	r32,(x)mm	1	3	1	FP0	
PEXTRW	r32,(x)mm,i	1	4	1	FP0	
PINSRW	mm,r32,i	2	8 7	1	FP0/1	
PINSRB/W/D/Q	x,r,i	2	'	1 1	FP0/1 FP0/1	
PINSRB/W/D/Q	x,m,i	-	3			00544
PEXTRB/W/D/Q PEXTRB/W/D/Q	r,x,i m v i	1 1	3	1 1	FP0 FP1	SSE4.1 SSE4.1
INSERTQ	m,x,i	3	2	2	FP0, FP1	SSE4.1 SSE4A, AMD only
INSERTQ	x,x x,x,i,i	3	2	2	FP0, FP1 FP0, FP1	SSE4A, AMD only
EXTRQ	X,X,I,I X,X	1	1	0.5	FP0, FF1	SSE4A, AMD only
LATING	^,^	ı '	· '	0.0	110/1	OCETA, AND ONLY

			Ŭ			
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
PMOVZXBW/BD/BQ/						
WD/WQ/DQ	x,x	1	2	0.5	FP0/1	SSE4.1
Arithmetic instruction	 S					
PADDB/W/D/Q						
PADDSB/W						
ADDUSB/W PSUBB/W/D/Q						
PSUBSB/W						
PSUBUSB/W	(x)mm,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	(x)mm,r/m	1	1	0.5	FP0/1	SSE4.1
PCMPGTQ	(x)mm,r/m	1	1	0.5	FP0/1	SSE4.2
PMULLW PMULHW						
PMULHUW						
PMULUDQ	(x)mm,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 PMADDWD	(x)mm,r/m	1	2 2	1 1	FP0 FP0	Suppl. SSE3
PMADDUBSW	(x)mm,r/m (x)mm,r/m	1	2	1	FP0	Suppl. SSE3
PAVGB/W	(x)mm,r/m	1	1	0.5	FP0/1	Suppl. 33L3
PMIN/MAX SW/UB	(x)mm,r/m	1	1	0.5	FP0/1	
PABSB/W/D	(x)mm,r/m	1	1	0.5	FP0/1	Suppl. SSE3
PSIGNB/W/D	(x)mm,r/m	1	1	0.5	FP0/1	Suppl. SSE3
PSADBW	(x)mm,r/m	1	2	0.5	FP0/1	
MPSADBW	x,x,i	3	4	1	FP0/1	SSE4.1
Logic						
PAND PANDN POR						
PXOR	(x)mm,r/m	1	1	0.5	FP0/1	
PSLL/RL W/D/Q						
PSRAW/D	mm,i/mm/m	1	1	0.5	FP0/1	
PSLL/RL W/D/Q					ED0 //	
PSRAW/D	X,X	1	2	0.5	FP0/1	
PSLL/RL W/D/Q PSRAW/D	v:	1	1	0.5	FP0/1	
PSLLDQ, PSRLDQ	x,i x,i	1	1 2	0.5	FP0/1 FP0/1	
PTEST	x,x/m	1	3	1	FP0/1	SSE4.1
1 1201	Χ,ΧΙΙΙ	'		'	110	OOL4.1
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							
PCLMULQDQ	x,x/m,i	1	3	1	FP0	PCLMUL	
AESDEC	x,x	2	5	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 XMM instructions

Floating point XMM			T			
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/D	m,x	1	4	1	FP1	
MOVNTPS/D	m,x	1	429	1	FP1	
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 AVX VUNPCK H/L PS/D y,y,y 2 2 1 FP0/1 AVX EXTRACTPS r32,x,i 1 3 1 FP0 FP0/1 EXTRACTPS m32,x,i 1 3 1 FP1 AVX VEXTRACTF128 x,y,i 1 1 0.5 FP0/1 AVX VEXTRACTF128 m128,y,i 1 12 1 FP1 AVX VEXTRACTF128 x,x,i 1 12 1 FP0/1 AVX INSERTPS x,x,i 1 1 FP0/1 AVX VINSERTF128 y,y,x,i 2 1 1 FP0/1 AVX VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks </th
UNPCK H/L PS/D
VUNPCK H/L PS/D y,y,y 2 2 1 FP0/1 AVX EXTRACTPS r32,x,i 1 3 1 FP0 FP0 EXTRACTPS m32,x,i 1 3 1 FP1 AVX 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 FP0/1 INSERTF128 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 masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
EXTRACTPS r32,x,i 1 3 1 FP0 EXTRACTPS m32,x,i 1 3 1 FP1 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 FP0/1 INSERTPS x,m32,i 1 6 1 FP0/1 AVX VINSERTF128 y,y,x,i 2 1 1 FP0/1 AVX VMASKMOVPS/D x,x,m128 1 15 1 FP0/1 >300 clk if masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
EXTRACTPS m32,x,i 1 3 1 FP1 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 FP0/1 INSERTPS x,m32,i 1 6 1 FP0/1 AVX VINSERTF128 y,y,x,i 2 1 1 FP0/1 AVX VMASKMOVPS/D x,x,m128 1 15 1 FP0/1 >300 clk if masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
EXTRACTPS m32,x,i 1 3 1 FP1 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 FP0/1 INSERTPS x,m32,i 1 6 1 FP0/1 AVX VINSERTF128 y,y,x,i 2 1 1 FP0/1 AVX VMASKMOVPS/D x,x,m128 1 15 1 FP0/1 >300 clk if masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
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 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 masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
VEXTRACTF128 m128,y,i 1 12 1 FP1 AVX INSERTPS x,x,i 1 1 FP0/1 FP0/1 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 masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
INSERTPS
INSERTPS x,m32,i 1 6 1 FP0/1 AVX 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 masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
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 masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
VINSERTF128 y,y,m128,i 2 13 2 FP0/1 AVX VMASKMOVPS/D x,x,m128 1 15 1 FP0/1 >300 clk if masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
VMASKMOVPS/D x,x,m128 1 15 1 FP0/1 >300 clk if masks VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
VMASKMOVPS/D y,y,m256 2 15 2 FP0/1 >300 clk if masks
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
VCVTPD2PS
CVTSD2SS
CVTSS2SD
CVT(T)PS2DQ
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
VCVTPS2PH x/m,x,i 1 4 1 FP1 F16C
VCVTPS2PH x/m,y,i 3 6 2 FP0, FP1 F16C
VCVTPH2PS
VCVTPH2PS y,x/m 2 5 2 FP1 F16C
Arithmetic
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 FP1

			ouge	iai		
MULSD/PD	x,x/m	1 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	
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	y,y/m	2	2	2	FP0	
(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	y, y/111,1 X,X,İ	5	11	4	FP0, FP1	SSE4.1
DPPS	x,m,i	6	11	4	FP0, FP1	SSE4.1
VDPPS	y,y,y,i	10	12	7	FP0, FP1	SSE4.1
VDPPS	y,y,y,i y,m,i	12	12	7	FP0, FP1	SSE4.1
DPPD	y,111,1 X,X,İ	3	9	3	FP0, FP1	SSE4.1
DPPD	x,x,i x,m,i	4	9	3	FP0, FP1	SSE4.1
שררט	Х,111,1	4		3	FFU, FFT	33E4.1
Logio						
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	
VANDES/D, etc.	у,у/111		ı	'	FFU/I	
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		2	2	2	FP1	
VROURIFO	y,y/m		2		FFI	
Other						
LDMXCSR	m	12	9	8	FP0, FP1	
STMXCSR	m	3	13	12	FP0, FP1	
VZEROUPPER	111	21	13	30	1 1 0, 1 1 1	32 bit mode
VZEROUPPER		37		46		64 bit mode
VZEROUPPER		41		46 58		32 bit mode
		73				
VZEROALL			66	90		64 bit mode
FXSAVE		66	66 50	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

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	i .		l <u> </u>
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 DIV r6/m8 17 np DIV r16/m16 25 np DIV r16/m16 25 np DIV r32/m32 41 np IDIV r16/m16 30 np IDIV r16/m16 13 np IDIV<				
DIV	MUL IMUL	r8/r16/m8/m16	11	np
DIV	MUL IMUL	all other versions	9 d)	np
DIV	DIV	r8/m8	17	np
DIV	DIV	r16/m16	25	np
IDIV	DIV	r32/m32	41	-
IDIV IDIV	IDIV		22	-
IDIV				· ·
CBW CWDE 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 m,i 3 u ROR ROL RCR RCL r/m, i 1/3 u ROR ROL RCR RCL r/m, i(><1)				-
CWD CDQ SHR SHL SAR SAL SHR SHL SAR SAL SHR SHL SAR SAL SHR SHL SAR SAL SHR SHL SAR SAL T/m, CL ROR ROL RCR RCL ROR ROL ROR ROL ROR ROL ROR ROL T/m, i(><1) 1/3 ROR ROL ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROL ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL T/m, i(><1) ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROL T/m, i(><1) ROR ROL ROR ROR		102/11102		-
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, i(><1)				-
SHR SHL SAR SAL m, i 3 u SHR SHL SAR SAL r/m, CL 4/5 np ROR ROL RCR RCL r/m, i(><1)		r i		-
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)				
ROR ROL RCR RCL ROR ROL ROR ROL ROR ROL ROR ROL ROR ROL ROR ROL ROR ROL ROR ROL RCR RCL RCC RC RCL RCC RC RCL RCC RC RCL RCC RC RC RC RCL RCC RC RCC RC RCC RCC RCC RCC RCC RCC R				
ROR ROL ROR ROL ROR ROL RCR RCL RCR RCL RCR RCL SHLD SHRD SHLD SHRD RT RCR RCL SHLD SHRD RT RCR RCL SHLD SHRD RT RCR RCL RCR RCL SHLD SHRD RT RCR RCL SHLD SHRD RT RCR RCL SHLD SHRD RT RCR RCL SHLD SHRD RT RCR RCL SHLD SHRD RT RCR RCL SHLD SHRD RT RCR RCL SHLD SHRD RT RCR RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD SHRD RT RCL SHLD RT RCL SHLD RT RCL SHLD RT RCL SHLD RT RCL SHLD RT RCL SHLD RT RCT RCT RCT RCT RCT RCT RCT RCT RCT		· ·		-
ROR ROL RCR RCL RCR RCL RCR RCL RCR RCL RCR RCL RCH RCR RCL RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCR RCL RCH RCH RCR RCL RCH RCH RCH RCR RCL RCH RCH RCH RCH RCH RCH RCH RCH RCH RCH		·		
RCR RCL r/m, i(><1)		, ,		· ·
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 8 a) np BT m, i 8 a) np BT m, i 8 a) np BT m, i 8 a) np BT m, i 8 a) np BT m, r/m 1/2 a) np MP CALL short/near 1 e) v CALL JMP 2/5 e np RETN		· ·		-
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) np BT m, i 9 a) np BTR BTS BTC m, i 7 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, r 14 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, i 8 a) np BFT MCALL sh		, , ,		-
SHLD SHRD m, i/CL 5 a) np BT r, r/i 4 a) np BT m, i 4 a) np BT m, i 9 a) np BTR BTS BTC r, r/i 7 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, r 14 a) np BSF BSR r, r/m 7-73 a) np SETcc r/m 1/2 a) np JMP CALL short/near 1 e) v JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np SETcc r/m 1/2 a) np JMP CALL short/near 1 e) v JMP CALL short/near 1 e) v JMP CALL short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN i 3/6 e) np RETN i		· ·		-
BT r, r/i 4 a) np BT m, i 4 a) np BT m, i 9 a) np BTR BTS BTC r, r/i 7 a) np BTR BTS BTC m, r 14 a) np BTR BTS BTC m, r 14 a) np BSF BSR r, r/m 7-73 a) np SETCc r/m 1/2 a) np JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np RETN i <t< td=""><td></td><td></td><td>,</td><td>np</td></t<>			,	np
BT		· ·	,	np
BT m, i 9 a) np BTR BTS BTC r, r/i 7 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, r 14 a) np BSF BSR r, r/m 7-73 a) np SETCC r/m 1/2 a) np JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN 2/5 e np np RETN 1 3/6 e) np RETF i 5/8 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np BOUND r, m 8 np CLI STI 6-9 np LODS 7+3*n g) np STOS 3 np REP STOS		r, r/i	,	np
BTR BTS BTC r, r/i 7 a) np BTR BTS BTC m, i 8 a) np BTR BTS BTC m, r 14 a) np BSF BSR r, r/m 7-73 a) np SETCC r/m 1/2 a) np JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN 2/5 e np np RETN i 3/6 e) np RETF i 5/8 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np BOUND r, m 8 np CLI STI 6-9 np LODS 7+3*n g) np REP LODS 7+3*n g) np STOS 3 np REP MOVS 12		m, i	4 a)	np
BTR BTS BTC m, i 8 a) np BSF BSR r, r/m 7-73 a) np SETcc r/m 1/2 a) np JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN i 3/6 e) np RETN i 3/6 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CL STC CMC CLD STD 2 np CLI STI 6-9 np LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np SCAS 4 np <	BT	m, i	9 a)	np
BTR BTS BTC m, r 14 a) np BSF BSR r, r/m 7-73 a) np SETcc r/m 1/2 a) np JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN i 3/6 e) np RETN i 3/6 e) np RETF i 5/8 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CL STC CMC CLD STD 2 np CLI STI 6-9 np LODS 7+3*n g) np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) n	BTR BTS BTC	r, r/i	7 a)	np
BSF BSR SETCC r, r/m 7-73 a) np JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN 2/5 e np RETN i 3/6 e) np RETF 4/7 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CL STC CMC CLD STD 2 np CLI STI 6-9 np LODS 7+3*n g) np STOS 3 np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 5 np CMPS 8+4*n g) np	BTR BTS BTC	m, i	8 a)	np
SETCC r/m 1/2 a) np JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN 2/5 e np RETN i 3/6 e) np RETF 4/7 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 7+3*n g) np STOS 3 np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np	BTR BTS BTC	m, r	14 a)	np
JMP CALL short/near 1 e) v JMP CALL far >= 3 e) np conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN 2/5 e np RETN 4/7 e) np RETF 4/7 e) np RETF 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 3 np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 5 np CMPS 8+4*n g) <t< td=""><td>BSF BSR</td><td>r , r/m</td><td>7-73 a)</td><td>np</td></t<>	BSF BSR	r , r/m	7-73 a)	np
JMP CALL far >= 3 e) np conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN 2/5 e np RETN i 3/6 e) np RETF 4/7 e) np np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 5 np CMPS 8+4*n g) np BSWAP <td>SETcc</td> <td>r/m</td> <td>1/2 a)</td> <td>np</td>	SETcc	r/m	1/2 a)	np
conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN i 3/6 e) np RETN i 3/6 e) np RETF 4/7 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 8+4*n g) np BSWAP r 1 a) np	JMP CALL	short/near	1 e)	V
conditional jump short/near 1/4/5/6 e) v CALL JMP r/m 2/5 e np RETN i 3/6 e) np RETN i 3/6 e) np RETF 4/7 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 8+4*n g) np BSWAP r 1 a) np	JMP CALL	far	>= 3 e)	np
CALL JMP r/m 2/5 e np RETN i 3/6 e) np RETN i 3/6 e) np RETF 4/7 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np	conditional jump	short/near	1/4/5/6 e)	V
RETN i 3/6 e) np RETF i 3/6 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np		r/m	2/5 e	np
RETN i 3/6 e) np RETF 4/7 e) np RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np	RETN		2/5 e	-
RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r , m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np	RETN	i	3/6 e)	-
RETF i 5/8 e) np J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				-
J(E)CXZ short 4-11 e) np LOOP short 5-10 e) np BOUND r , m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np		i	,	
LOOP short 5-10 e) np BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np		short	,	-
BOUND r, m 8 np CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				-
CLC STC CMC CLD STD 2 np CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np			,	-
CLI STI 6-9 np LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np		, ,		· ·
LODS 2 np REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				
REP LODS 7+3*n g) np STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				
STOS 3 np REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				-
REP STOS 10+n g) np MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				-
MOVS 4 np REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				-
REP MOVS 12+n g) np SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				-
SCAS 4 np REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				
REP(N)E SCAS 9+4*n g) np CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np			σ,	
CMPS 5 np REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np				-
REP(N)E CMPS 8+4*n g) np BSWAP r 1 a) np	1		•	-
BSWAP r 1 a) np				-
			•	np
CPUID 13-16 a) np		r	1 a)	np
	CPUID		13-16 a)	np

RDTSC		6-13 a) j)	np	
Notes:				
a	This instruction ha code on a P1 unle			e clock cycle extra to de- instruction.
b	versions with FS a	ind GS have a	OFH prefix. see	e note a.
С	versions with SS,	FS, and GS hav	ve a 0FH prefix	k. see note a.
d	versions with two	operands and n	io immediate h	ave a 0FH prefix, see
е	निर्श्निम िश्वlues are fo	r mispredicted j	umps/branche	S.
f	only pairable if reg	jister is AL, AX	or EAX.	
g	add one clock cycle multi-cycle instruct			fix unless preceded by a
h	pairs as if it were v	writing to the ac	cumulator.	
i	9 if SP divisible by	4 (imperfect pa	airing).	
j	on P1: 6 in privileg mode. On PMMX:	•	•	ivileged; error in virtual

Floating point instructions (Pentium and Pentium MMX)

Explanation of column headings

Operands r = register, m = memory, m32 = 32-bit memory operand, etc. The numbers are minimum values. Cache misses, misalignment, **Clock cycles**

denormal operands, and exceptions may increase the clock counts

considerably.

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

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

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
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

r

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

n 1 if the overlapping instruction is also an FMUL.
o 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	Operands µops							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)		.0		2		8	8		
POPA(D)				2	8				
LAHF SAHF				1					
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	3		1				23	21
DIV IDIV	r32	3		1				39	37
DIV IDIV	m8	2		1	1			19	12
DIV IDIV	m16	2		1	1			23	21
DIV IDIV	m32	2		1	1			39	37
CBW CWDE				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	1	
JMP	short/near		1	ļ '		'	'		2
JMP	far	21	'		1				
JMP	r	21	1		1				2
	-				4				2
JMP	m(near)	24	'		1				
JMP	m(far)	21	4		2				
conditional jump	short/near		1	,			_		2 2
CALL	near	00	1	1	_	1	1		2
CALL	far	28	١,		1	2	2		
CALL	r		1	2	_	1	1		2
CALL	m(near)		1	4	1	1	1		2
CALL	m(far)	28	١.		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				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	'n					
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	1	'					•
RDTSC		31	I						
IN		18						>300	
		18							
OUT		18			4			>300	
PREFETCHTO(4/2 d)	m m				1				
PREFETCHT0/1/2 d)	m				1				
SFENCE d)						1	1		6

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	μops						Latency	Reciprocal
		p0	p1	p01		рЗ	p4	1	throughput
FLD	r	1			-		ļ .		
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		
FADD(P) FSUB(R)(P)	r	1						3	1
FADD(P) FSUB(R)(P)	m	1			1			3-4	1
FMUL(P)	r	1						5	2 g)
FMUL(P)	m	1			1			5-6	2 g)
FDIV(R)(P)	r	1						38 h)	37
FDIV(R)(P)	m	1			1			38 h)	37
FABS		1							
FCHS		3						2	
FCOM(P) FUCOM	r	1						1	
FCOM(P) FUCOM	m	1			1			1	
FCOMPP FUCOMPP		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				
FFREE	r	1				
FFREEP	r	2				
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-

and 1)

tency-1).

i) Faster for lower precision.

Integer MMX instructions (Pentium II and Pentium III)

Instruction	Operands			μ	ops		Latency	Reciprocal	
		p0	p1	p01	p2	рЗ	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

Pentium II and III

PMOVMSKB d)	r32,mm		1					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			μ	ops			Latency	Reciprocal
		p0	p1	p01	p2	рЗ	p4		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	1
MOVHPS MOVLPS	m64,xmm					1	1	1	1
MOVLHPS MOVHLPS	xmm,xmm			1				1	1
MOVMSKPS	r32,xmm	1						1	1
MOVNTPS	m128,xmm					2	2		2 - 15
CVTPI2PS	xmm,mm		2					3	1
CVTPI2PS	xmm,m64		2		1			4	2
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

Pentium II and III

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
MAXPS 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
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.

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

ment 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.

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	Operands µops µops unfused domain fused								Recipro- cal
		domain	p0	p1	p01	p2	рЗ	p4		through- put
Move instructions										
MOV	r,r/i	1			1					0,5
MOV	r,m	1				1				1 1
MOV	m,r	1					1	1		1 1
MOV	m,i	2					1	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												
NUMBER N	XCHG	r.m	7			4	1	1	1	hịah h)		
PUSH		.,								g 5/	1	
PUSH		_				'	'	1	4	4		
PUSH								1	-			
PUSHF(D)								1	-			
PUSHA(D)		m					1	1	1	2	1	
PUSHA(D)	PUSH	sr	2			1		1	1			
PUSHA(D)	PUSHF(D)		16	3		11		1	1		6	
POP	` '							8	8	8		
POP	` '	r				-	1					
POP												
POP		, ,				2						
POPF(D)		m m						1	1	2	1	
POPA(D)		sr	10			9	1					
LAHF SÄHF SALC 2	POPF(D)		17	10		6	1				16	
LAHF SÄHF	POPA(D)		10			2	8			7	7	
SALC						1					1	
LEA				1	1	'						
BSWAP		rm		1	'					4		
DS LES LFS LGS LSS				1						'	'	
PREFETCHNTA				1								
PREFETCHT0/1/2		m	11			8	3					
SFENCE/LFENCE/MFENCE 18	PREFETCHNTA	m	1				1				1	
N	PREFETCHT0/1/2	m	1				1				1	
N	SFENCE/LFENCE/MFENCE		2					1	1		6	
Arithmetic instructions			_	18				-	-	>300		
Arithmetic instructions												
ADD SUB r,r/i 1 1 1 1 1 2 1 0,5 ADD SUB r,m 1	001			10						/300		
ADD SUB r,r/i 1 1 1 1 1 2 1 0,5 ADD SUB r,m 1												
ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADD SUB ADC SBB r,r/i ADC SBB ADC SBB r,r/i ADC SBB ADC SBB r,r/i ADC SBB r,r/i ADC SBB ADC SB 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 SBB ADC SBB ADC SBB ADC SBB ADC SBB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC SB ADC		,,	_								0.5	
ADD SUB ADC SBB ADC SBB I,r/i ADC SAMP I,r/i ADC SAMP I		·										
ADC SBB ADC SA SA SA SA SA SA SA SA SA SA SA SA SA		r,m				1	1			2	1	
ADC SBB ADC SBB ADC SBB ADC SBB CMP CMP CMP CMP CMP CMP M,r/i 1 1 1 1 1 1 1 1 1 1 1 0,5 CMP CMP M,r 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	ADD SUB	m,r/i	3			1	1	1	1		1	
ADC SBB CMP CMP CMP CMP CMP CMP CMP C	ADC SBB	r,r/i	2		1	1				2	2	
ADC SBB CMP CMP CMP CMP CMP CMP CMP C	ADC SBB	r,m	2		1	1	1					
CMP r,r/i 1 </td <td></td> <td></td> <td></td> <td></td> <td></td> <td>4</td> <td>1</td> <td>1</td> <td>1</td> <td></td> <td></td> <td></td>						4	1	1	1			
CMP m,r 1 <td></td> <td></td> <td></td> <td></td> <td></td> <td>1 -</td> <td>•</td> <td></td> <td></td> <td>1</td> <td>0.5</td> <td></td>						1 -	•			1	0.5	
CMP m,i 2 1 1 1 1 1 0,5 INC DEC NEG NOT INC DEC NEG NOT AAA AAS DAA DAS m 3 1 0,5 1 1 1 1 0,5 1 1 1 0,5 1 1 1 0,5 1 1 1 1 1 1 0,5 1 1 1 1 1 0,5 1<						1 -	4					
INC DEC NEG NOT						١.				'		
INC DEC NEG NOT						-	1				· ·	
AAA AAS DAA DAS 1 1 1 2 2 AAM 4 1 1 2 15 MUL IMUL r8 1 1 4 1 MUL IMUL r16/r32 3 3 5 1 IMUL r,r 1 1 4 1 IMUL r,r,i 1 1 4 1 MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 1 15-16 c) 12-20 c) DIV IDIV r8						1 -				1	0,5	
AAD 3 1 2 2 AAM 4 1 1 2 15 MUL IMUL r8 1 1 4 1 MUL IMUL r16/r32 3 3 5 1 IMUL r,r 1 1 4 1 IMUL r,r,i 1 1 4 1 MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12-20 c) DIV IDIV r32 4 3 1		m				1	1	1	1			
AAM 4 1 1 2 4 1 1 2 4 1 MUL IMUL r16/r32 3 3 3 5 1 IMUL r,r 1 1 4 1 IMUL r,r,i 1 1 4 1 MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IDIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12	AAA AAS DAA DAS		1		1							
MUL IMUL r8 1 1 4 1 MUL IMUL r16/r32 3 3 5 1 IMUL r,r 1 1 4 1 IMUL r,r,i 1 1 4 1 MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12	AAD		3	1		2				2		
MUL IMUL r8 1 1 4 1 MUL IMUL r16/r32 3 3 5 1 IMUL r,r 1 1 4 1 IMUL r,r,i 1 1 4 1 MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12	AAM		4	1	1	2				15		
MUL IMUL r16/r32 3 3 5 1 IMUL r,r 1 1 4 1 IMUL r,r,i 1 1 4 1 MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c)		r8	1	1							1	
IMUL r,r 1 1 4 1 IMUL r,r,i 1 1 4 1 MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r16 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12			l									
IMUL r,r,i 1 1 4 1 MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r16 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12												
MUL IMUL m8 1 1 1 4 1 MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r16 4 3 1 15-39 c) 12-20 c) DIV IDIV r32 4 3 1 1 15-16 c) 12												
MUL IMUL m16/m32 3 3 1 5 1 IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r16 4 3 1 15-24 c) 12-20 c) DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12												
IMUL r,m 1 1 1 4 1 IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r16 4 3 1 15-24 c) 12-20 c) DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12												
IMUL r,m,i 2 1 1 4 1 DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r16 4 3 1 15-24 c) 12-20 c) DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12												
DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r16 4 3 1 15-24 c) 12-20 c) DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12		r,m		1							1	
DIV IDIV r8 5 4 1 15-16 c) 12 DIV IDIV r16 4 3 1 15-24 c) 12-20 c) DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12	IMUL	r,m,i	2	1			1			4	1	
DIV IDIV r16 4 3 1 15-24 c) 12-20 c) DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12	DIV IDIV		5	4		1				15-16 c)	12	
DIV IDIV r32 4 3 1 15-39 c) 12-20 c) DIV IDIV m8 6 4 1 1 15-16 c) 12						1 .				,		
DIV IDIV m8 6 4 1 1 1 15-16 c) 12												
						-	4					
אוטו אוטן mio 5 3 1 1 15-24 c) 12-20 c)												
	אוטו אוטן	m 16	5	ა		1	1		1	15-∠4 C)	12-20 C)	

DIV IDIV CBW CWDE	m32	5 1	3	1	1	1			15-39 c)	12-20 c)
CWD CDQ		1		1					1 1	1
CWD CDQ		ı		1					'	1
Logic instructions										
AND OR XOR	r,r/i	1			1				1	0,5
AND OR XOR	r,m	1			1	1			2	1
AND OR XOR	m,r/i	3			1	1	1	1		1
TEST	r,r/i	1			1				1	0,5
TEST	m,r	1			1	1			1	1
TEST	m,i	2			1	1				1
SHR SHL SAR ROR ROL	r,i/CL	1	1						1	1
SHR SHL SAR ROR ROL	m,i/CL	3	1			1	1	1		-
RCR RCL	r,1	2	1		1		•		2	2
RCR	r8,i/CL	9	5		4				11	_
RCL	r8,i/CL	8	4		4				10	
RCR RCL	r16/32,i/CL	6	3		3				9	9
RCR RCL	m,1	7	2		2	1	1	1		
RCR	m8,i/CL	12	6		3	1	1	1		
RCL	m8,i/CL	11	5		3	1	1	1		
RCR RCL	m16/32,i/CL	10	5		2	1	1	1		
SHLD SHRD	r,r,i/CL	2	2		_	'	'	'	2	2
SHLD SHRD	m,r,i/CL	4	1		1	1	1	1		_
BT	r,r/i	1	'	1	'	'	'	'	1	1
BT	m,r	8		'	7	1			'	•
BT	m,i	2		1	'	1				
BTR BTS BTC	r,r/i	1		1		'				
BTR BTS BTC	m,r	10		'	7	1	1	1	6	
BTR BTS BTC	m,i	3		1	'	1	1	1		
BSF BSR	r,r	2		1	1	'	'	'		
BSF BSR	r,m	2		1	1	1				
SETcc	r .,	1		1	'	'				
SETcc	m '	2		1			1	1		
CLC STC CMC		1		1			'	'		1
CLD STD		4		'	4					7
OLD OTD		7			-					'
Control transfer instruction	 no									
JMP	short/near	1		1						1
JMP	far	22	21	'		1				28
JMP	r	1	21	1		'				1
JMP	m(near)	2		1		1				2
JMP	m(far)	25	23	'		2				31
conditional jump	short/near	1	25	1		_				1
J(E)CXZ	short	2		1	1					1
LOOP	short	11	2	1	8					6
LOOP(N)E	short	11	2	1	8					6
CALL	near	4		1	1		1	1		2
CALL	far	32	27	'	'	1	2	2		27
CALL	r iai	32 4	~ '	1	2	'	1	1		9
CALL	m(near)	4		1	_	1	1	1		2
CALL	m(far)	35	29	'		2	2	2		30
RETN	in(iai)	2	29	1	2	1		_		2
1.2-1.14	I	_	1	'	_	'		1	1	_

RETN RETF RETF BOUND INTO	i i r,m	3 27 27 15 5	24 24 7	1	6 5	1 3 3 2				2 30 30 8 4
String instructions LODS REP LODS		2 6n			10+6r	2				4 0,5
STOS		3				1	1	1		1
REP STOS		5n		(a. 5r	a)				0,7
MOVS		6			1	3	1	1		0,7
REP MOVS		6n		(ca. 6r	,				0,5
SCAS		3			1	2				1,3
REP(N)E SCAS CMPS		7n 6			12+7r 4	1 2				0,6 0,7
REP(N)E CMPS		9n			12+9r					0,7
Other										
NOP (90)		1			1					0,5
Long NOP (0F 1F)		1			1					1
PAUSE		2			2					
CLI			9							
STI ENTER	i,0	12	17		10		1	1		
ENTER	a,b	12	ca.	18	+4b		b-1	2b		
LEAVE	α,υ	3	Ca.	10	2	1	D-1	20		
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".

tel, AMD and VIA CI US.

b) Has an implicit LOCK prefix.

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

nstruction	Operands	µops fused	μ	ops	unfus	sed d	Latency	Recipro- cal		
		domain	p0	p1	p01	p2	р3	p4		through-
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 FIST(P) FISTTP g)	m m m	4 4 4	3 2 2			1	1 1	1 1	5 5 5	2 2 2
FLDZ		1	1							
FLD1 FLDPI FLDL2E etc.		2	2							
FCMOVcc	r	2	2						2 7	
FNSTSW	AX	3	3						/	3
FNSTSW	m16	2	1		_		1	1		4.0
FLDCW	m16	3	1		1	1		_		19
FNSTCW	m16	3	1				1	1	_	3
FINCSTP FDECSTP		1	1						1	
FFREE	r	1	1							1
FFREEP	r	2	2							2
FNSAVE		142	142							131
FRSTOR		72	72							91
Arithmetic instructions										
FADD(P) FSUB(R)(P)	r	1			1				3	1
FADD(P) FSUB(R)(P)	m	1			1	1			3	1
FMUL(P)	r	1	1						5	2
FMUL(P)	m	1	1			1			5	2
FDIV(R)(P)	r	1	1						9-38 c)	8-37 c)
FDIV(R)(P)	m	1	1			1			9-38 c)	8-37 c)
FABS		1	1						1	1
FCHS		1	1						1	1
FCOM(P) FUCOM	r	1		1					1	1
FCOM(P) FUCOM	m	1		1		1			1	1
FCOMPP FUCOMPP		2		1	1				1	1
FCOMI(P) FUCOMI(P)	r	1		1					1	1
FIADD FISUB(R)	m	6	3	1	1	1			3	3
FIMUL	m m	6	5			1			5	3
FIDIV(R)	m m	6	5			1			9-38 c)	8-37 c)
FICOM(P)	m m	6	3	2		1				4
FTST		1		1						1
FXAM		1		1						1
FPREM FPREM1		26	26						37	
FRNDINT		15	15						19	
Math										
FSCALE		28	28						43	
FXTRACT		15			15				9	
FSQRT		1	1						9 h)	8
FSIN FCOS		80-100	80-10	00				80-1		
FSINCOS		90-110	90-11	10				100-	130	
F2XM1		~ 20	~20					~45		
FYL2X		~ 40	~40					~60		
FYL2XP1		~ 55	~55					~65		
FPTAN		~ 100	~100					~140		
FPATAN		~ 85	~85					~140		
Other										
FNOP		1	1							1
WAIT		2		1	1					1

FNCLEX	3	3		13
FNINIT	14	14		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

any port.

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

Integer MMX and XMM instructions

Instruction	Operands	μοps fused	μ	ops	unfus	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					1	1		2
MOVNTDQ	m128,xmm	4					2	2		2 3
PACKSSWB/DW	,									
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 PUNPCKLQDQ PSHUFW PSHUFW PSHUFD PSHUFD PSHUFL/HW MASKMOVQ MASKMOVDQU PMOVMSKB PMOVMSKB PEXTRW PEXTRW	xmm,xmm xmm, m128 mm,mm,i mm,m64,i xmm,xmm,i xmm,m128,i xmm,xmm,i xmm, m128,i mm,mm xmm,xmm r32,mm r32,xmm r32,xmm,i	1 1 1 2 3 4 2 3 8 1 1 2	1 1 2 1 1 1 1 2	1 1 1 1 1 1 j) 1 2	1	1 1 2 2	1 2	1 2	1 1 2 1 1 2 3	1 1 1 2 2 1 1 1
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 PADD/SUB(U)(S)B/W/D	mm,mm mm,m64	1 1			1 1	1			1	0,5 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	mm,mm	2			2				2	1
PADDQ PSUBQ	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	mm,mm	1			1				1	0,5
PCMPEQ/GTB/W/D	mm,m64	1			1	1				1
PCMPEQ/GTB/W/D	xmm,xmm	2			2				1	1
PCMPEQ/GTB/W/D	xmm,m128	2			2	2				2
PMULL/HW PMULHUW	mm,mm	1			1				3	1 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 PAND(N) POR PXOR PAND(N) POR PXOR	mm,mm mm,m64 xmm,xmm	1 1 2			1 1 2	1		1	0,5 1 1	
PAND(N) POR PXOR	xmm,m128	4			2	2		'	2	
PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q	mm,mm/i mm,m64	1 1	1 1			1		1	1 1	
PSLL/RL/RAW/D/Q	xmm,i	2	2			•		2	2	
PSLL/RL/RAW/D/Q PSLL/RL/RAW/D/Q	xmm,xmm xmm,m128	3 3	2	1 1		2		2	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)

Also uses some execution units under port 1. j)

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	loma	in	Latency	Recipro- cal
		domain	p0	p1	p01	p2	рЗ	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	7	2
CVTSS2SD		2	2		_	'	2	2
	xmm,xmm		2			,		2
CVTSS2SD	xmm,m64	3	2		_	1	_	2
CVTDQ2PS	xmm,xmm	2			2		3	
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 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		1
CVT(T)SD2SI	r32,xmm	2	'	1	1	'	4	1 1
CVT(T)SD2SI	r32,m64	3		1	1	1	-	1
CV1(1)3D231	132,11104	3		I	ı	'		1
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 1
MULSD	xmm,xmm	1	1				5	2
MULSS	xmm,m32	2	1			1	4	1
MULSD	xmm,m64	2	1			1	5	2
MULPS	xmm,xmm	2	2			'	4	2
MULPD	xmm,xmm	2	2				5	4
MULPS	xmm,m128	4	2			2	4	2
MULPD	xmm,m128	4	2			2	5	4
	· ·		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	xmm,m128	4	2			2 2			16-34 c)	16-34 c)
DIVPD	xmm,m128	4	2		,	2			16-62 c)	16-62 c)
CMPccSS/D	xmm,xmm	1			1	,			3	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
	7.1111,111120	т			_					'
Other										
LDMXCSR	m32	9	9							20
STMXCSR	m32	6	6							12
FXSAVE	m4096	118	32				43	43		63
FXRSTOR	m4096	87	43			44	70	70		72
I AROTOR	1117030	01	170		<u> </u>					1 4

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

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.

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	μοps fused	μops	un	fuse	d d	oma	ain		Unit	Laten- cy	Reci- 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	8	4	x	x	x	4			int		16
MOV	sr,m	8	3	×		x	5			int		16
MOVNTI	m,r	2						1	1	int		2
MOVSX MOVZX	,									_		
MOVSXD	r,r	1	1	×	x	X				int	1	0,33
MOVSX MOVZX	r,m	1	-				1			int		1
CMOVcc	r,r	2	2	x	Х	X	-			int	2	1
CMOVcc	r,m	2	2	X	X	X	1			int	_	
XCHG	r,r	3	3	X	X	X	•			int	2	2
XCHG	r,m	7	x	^		``	1	1	1	int	high b)	_
XLAT	.,	2	1				1	_	•	int	4	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)	31	17	15	x	x	x		1	1	int		7
PUSHA(D) i)		18	9	^	^	^		1	8	int		8
POP	r	10					1	l '	"	int	2	1
POP	(E/R)SP	4	3				1			int		'
POP	1	2	٦				1	1	1	int		1.5
POP	m	10	9				1	'	'	int		1,5 17
POPF(D/Q)	sr	24	23		\ ,		1			int	20	17
, ,		10	23	X	X	X	8			int	20	7
POPA(D) i) LAHF SAHF		10	1		\ ,		0			int	1	0,33
SALC i)		2	2	X	X	X				int	4	1
LEA a)	r	1	1	1	X	X				int	1	1
BSWAP	r,m	2	2	1		1					4	1
I	r	11	11	'		'	4			int int	4	
LDS LES LFS LGS LSS PREFETCHNTA	m m	1	''				1			int		17 1
	m m											
PREFETCHT0/1/2	m	1 2					1	4	4	int		1
LFENCE MFENCE		2						1	1	int int		8 9
SFENCE		2										9
CLFLUSH	m8	4	2		\ ,			1	1	int int	240	117
IN	1110	4	4	X	X	X		'	'	int	240	117
OUT										int		
A vith we object in a two objects												
Arithmetic instructions ADD SUB	r,r/i	1	1		\ ,	,				int	1	0,33
ADD SUB	1	1	1	X	X	X	1			int	'	
ADD SUB	r,m m,r/i	2	1	X	X	X	1	1	1	int	6	1 1
ADC SBB	1	2	2		X			'	'	int	2	2
ADC SBB	r,r/i	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	
CMP	m,r/i	1	1	X	X	X		'	'	int	1	0,33
CMP	r,r/i	1 1	1	X	X	X	1			int	1	0,33 1
INC DEC NEG NOT	m,r/i			X	X	X						
	r	1	1	X	X	X	4	4	4	int	1	0,33
INC DEC NEG NOT	m	3	1	X	X	X	1	1	1	int	6	1

AAA AAS DAA DAS i)		1	1		1					int		1
AAD i)		3	3	х	х	Х				int		1
AAM i)		4	4							int	17	
MUL IMUL	r8	1	1		1					int	3	1
MUL IMUL	r16	3	3	x	х	х				int	5	1,5
MUL IMUL	r32	3	3	х	х	х				int	5	1,5
MUL IMUL	r64	3	3	x	x	х				int	7	4
IMUL	r16,r16	1	1		1					int	3	1 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 1
IMUL	r32,r32,i	1	1		1					int	3	1 1
IMUL	r64,r64,i	1	1	1	'					int	5	2
MUL IMUL	m8	1	1	١.	1		1			int	3	1 1
MUL IMUL	m16	3	3	X	, X	х	1			int	5	1,5
MUL IMUL	m32	3	3	X	X	X	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		1			int	3	1 1
IMUL	r64,m64			1	'					int	5	2
IMUL	r164,11164	1 1	1 1	'	4		1				5	2 2
I					1		-			int		
IMUL	r32,m32,i	1	1	4	1		1			int		1
IMUL	r64,m64,i	1	1	1			1			int	40	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	X	Х				int	1	
CWD CDQ CQO		1	1	X		Х				int	1	
Logic instructions	_											
AND OR XOR	r,r/i	1	1	X	X	Х				int	1	0,33
AND OR XOR	r,m	1	1	X	X	Х	1			int		1
AND OR XOR	m,r/i	2	1	X	X	Х	1	1	1	int	6	1
TEST	r,r/i	1	1	Х	Х	Х				int	1	0,33
TEST	m,r/i	1	1	Х	Х	Х	1			int		1
SHR SHL SAR	r,i/cl	1	1	Х		х				int	1	0,5
SHR SHL SAR	m,i/cl	3	2	х		Х	1	1	1	int	6	1
ROR ROL	r,i/cl	1	1	х		Х				int	1	1
ROR ROL	m,i/cl	3	2	Х		Х	1	1	1	int	6	1
RCR RCL	r,1	2	2	Х	Х	Х				int	2	2
RCR	r8,i/cl	9	9	Х	Х	х				int	12	
RCL	r8,i/cl	8	8	х	х	х				int	11	
RCR RCL	r16/32/64,i/cl	6	6	Х	Х	х				int	11	
RCR RCL	m,1 ^	4	3	Х	Х	х	1	1	1	int	7	
RCR	m8,i/cl	12	9	Х	Х	х	1	1	1	int	14	
RCL	m8,i/cl	11	8	X	X	Х	1	1	1	int	13	
ı	, - 1		-	1	1	ı	ı	I	ı l	-	-	ı 1

RCR RCL SHLD SHRD SHLD SHRD BT BT BT BT BTR BTS BTC BTR BTS BTC BTR BTS BTC BSF BSR SSF BSR 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 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 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	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 0,33 4 14
Control transfer instruction JMP JMP i) JMP JMP JMP JMP Conditional jump Fused compare/test and b J(E/R)CXZ LOOP LOOP(N)E CALL 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 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	1 1 1 1 1 X X X X	1 2 1 1 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 68 1 1-2 5 5 2 75 2 75 2 78 78 8
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 - 3 8 1 7+7n - 4 7+8n - 7 7+10n -	2 20+1.: 5 1 13+n 3 17+7r	 2n 1 1		 5 	1	 1 	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 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.

Floating point x87 instructions

Instruction	Operands	μορs fused	I							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				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	x	Х		2	2	float	4	5
FBSTP	m80	170	166	Х	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)	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
FCHS		1	1	1				float	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	μορs fused	μops	un	fuse	ed d	oma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	р5	p2	p3	p4			through- put
Move instructions												
MOVD k)	r32/64,(x)mm	1	1	Х	Х	x				int	2	0,33
MOVD k)	m32/64,(x)mm	1 1						1	1		3	1
MOVD k)	(x)mm,r32/64	1	1	Х		Х				int	2	0,5
MOVD k)	(x)mm,m32/64	1 1					1			int	2	1

MOVQ	(v)mm (v)mm	1	1	,	v		I	l	I	int	4	0.22
MOVQ	(x)mm, (x)mm	1 1	1	X	X	Х	1			int	1 2	0,33
	(x)mm,m64						'	4	4	int		1 1
MOVQ	m64, (x)mm	1	1		١.,	.,		1	1	:4	3	1
MOVDQA	xmm, xmm	1	'	X	Х	Х	4			int	1	0,33
MOVDQA	xmm, m128	1					1			int	2	1 1
MOVDQA	m128, xmm	1						1	1		3	1 1
MOVDQU	m128, xmm	9	4	Х	Х	Х	1	2	2		3-8	4
MOVDQU	xmm, m128	4	2	Х		Х	2			int	2-8	2
LDDQU g)	xmm, m128	4	2	Х		Х	2			int	2-8	2
MOVDQ2Q	mm, xmm	1	1	Х	X	Х				int	1	0,33
MOVQ2DQ	xmm,mm	1	1	X	X	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 1
PSHUFB h)	mm,mm	1	1			1				int	1	1
PSHUFB h)	mm,m64	2	1			1	1			int		1 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 1
PSHUFW	mm,m64,i	2	1			1	1			int		1 1
PSHUFD	xmm,xmm,i	2	2	х	x	1				flt→int	3	1 1
PSHUFD	xmm,m128,i	3	2	x	x	1	1			int		1 1
PSHUFL/HW	xmm,xmm,i	1	1			1				int	1	1 1
PSHUFL/HW	xmm, m128,i	2	1			1	1			int		1 1
PALIGNR h)	mm,mm,i	2	2	x	x	Х				int	2	1
PALIGNR h)	mm,m64,i	2	2	х	х	х	1			int		1 1
PALIGNR h)	xmm,xmm,i	2	2	Х	Х	х				int	2	1 1
PALIGNR h)	xmm,m128,i	2	2	X	X	Х	1			int	_	1 1
MASKMOVQ	mm,mm	4								int		2-5
MASKMOVDQU	xmm,xmm	10								int		6-10
PMOVMSKB	r32,(x)mm	1	1	1						int	2	1
PEXTRW	r32,mm,i	2	2	-						int	3	1 1
PEXTRW	r32,xmm,i	3	3							int	5	1 1
PINSRW	mm,r32,i	1	1			1				int	2	1
PINSRW	mm,m16,i	2	1			1	1			int	_	1
PINSRW	xmm,r32,i	3	3	X	х	Х	ľ			int	6	1,5
PINSRW	xmm,m16,i	4	3	X	X	X	1			int		1,5
I IIVOIVV	XIIIII,III TO,I	7		_ ^	^	^						1,0
Arithmetic instructions												
PADD/SUB(U)(S)B/W/D	(x)mm, (x)mm	1	1	X		х				int	1	0,5
PADD/SUB(U)(S)B/W/D	(x)mm,m	1	1	X		X	1			int		1
PADDQ PSUBQ	(x)mm, (x)mm	2	2	X		X				int	2	1
PADDQ PSUBQ	(x)mm,m	2	2	X		X	1			int	_	1
	(23)111113111	_	_	1 1	I	. ^	' '	I	1		I	ı '

PHADD(S)W	1		1	l	1	l	I				
PHSUB(S)W h)	mm,mm	5	5						int	5	4
PHADD(S)W	''''','''''	3	3						IIIC	3	7
PHSUB(S)W h)	mm,m64	6	5				1		int		4
PHADD(S)W	11111,11104	O	"						1110		7
PHSUB(S)W h)	xmm,xmm	7	7						int	6	4
PHADD(S)W	\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \	,	'						1111	0	7
PHSUB(S)W h)	xmm,m128	8	7				1		int		4
PHADDD PHSUBD h)	mm,mm	3	3						int	3	2
PHADDD PHSUBD h)	mm,m64	4	3				1		int	3	2
· · · · · · · · · · · · · · · · · · ·	·	5					'			5	3
PHADDD PHSUBD h)	xmm,xmm		5				4		int	5	3
PHADDD PHSUBD h)	xmm,m128	6	5				1		int	4	
PCMPEQ/GTB/W/D	(x)mm,(x)mm	1	1	Х		Х			int	1	0,5
PCMPEQ/GTB/W/D	(x)mm,m	1	1	X		Х	1		int		1
PMULL/HW PMULHUW	(x)mm,(x)mm	1	1		1				int	3	1
PMULL/HW PMULHUW	(x)mm,m	1	1		1		1		int	_	1
PMULHRSW h)	(x)mm,(x)mm	1	1		1				int	3	1
PMULHRSW h)	(x)mm,m	1	1		1		1		int		1
PMULUDQ	(x)mm,(x)mm	1	1		1				int	3	1
PMULUDQ	(x)mm,m	1	1		1		1		int		1
PMADDWD	(x)mm,(x)mm	1	1		1				int	3	1
PMADDWD	(x)mm,m	1	1		1		1		int		1
PMADDUBSW h)	(x)mm,(x)mm	1	1		1				int	3	1
PMADDUBSW h)	(x)mm,m	1	1		1		1		int		1
PAVGB/W	(x)mm,(x)mm	1	1	x		х			int	1	0,5
PAVGB/W	(x)mm,m	1	1	x		х	1		int		1
PMIN/MAXUB/SW	(x)mm,(x)mm	1	1	x		х			int	1	0,5
PMIN/MAXUB/SW	(x)mm,m	1	1	x		х	1		int		1
PABSB PABSW PABSD	(x)mm,(x)mm	1	1	x		х			int	1	0,5
h)	(x)mm,m	1	1	x		х	1		int		1
PSIGNB PSIGNW	(x)mm,(x)mm	1	1	x		х			int	1	0,5
PSIGND h)	(x)mm,m	1	1	x		х	1		int		1
PSADBW	(x)mm,(x)mm	1	1		1				int	3	1
PSADBW	(x)mm,m	1	1		1		1		int		1
	(11)	•									•
Logic instructions											
PAND(N) POR PXOR	(x)mm,(x)mm	1	1	х	х	x			int	1	0,33
PAND(N) POR PXOR	(x)mm,m	1	1	X	X	X	1		int	·	1
PSLL/RL/RAW/D/Q	mm,mm/i	1	1	1	^	^	ļ '		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	X	x				int	2	1
PSLL/RL/RAW/D/Q	xmm,m128	3	2	X	X		1		int	_	1
PSLL/RLDQ	xmm,i	2	2				'		int	2	1
F OLL/NLDW	AIIIII,I	2	~	X	X				1111		ı
Other											
EMMS		11	11	x	x	x			float		6
Notes:		1.1	1 11	_ ^	_ ^	_^			noat		U

Notes:

g) SSE3 instruction set.

h) 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

nstruction	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												
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	Х	1	2	2		3-4	4
MOVSS/D	xmm,xmm	1	1	Х	х	Х				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				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	'	ļ '				1	1	liout	'	2-3
SHUFPS	xmm,xmm,i	3	3		3			'	'	flt→int	3	2-3
SHUFPS	xmm,m128,i	4	3		3		1			flt→int		2
SHUFPD		1	1	1	3		'			float	1	1
	xmm,xmm,i	2	1	1			1				'	1 1
SHUFPD	xmm,m128,i		1							float		
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				int	1	1
MOVSH/LDUP g)	xmm,m128	2	1			1	1			int		1
UNPCKH/LPS	xmm,xmm	3	3		3					flt→int	3	2
JNPCKH/LPS	xmm,m128	4	3		3		1			int		2
UNPCKH/LPD	xmm,xmm	1	1	1						float	1	1
JNPCKH/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	xmm,m128	2	2				1			float	'	1

1			ı	ı	1	1 1	1 1			1 1	
CVTPI2PS	xmm,mm	1	1		1				float	3	3
CVTPI2PS	xmm,m64	1	1		1	1			float		3
CVT(T)PS2PI	mm,xmm	1	1		1				float	3	1
CVT(T)PS2PI	mm,m128	1	1		1	1			float		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 CVTSI2SD	xmm,m32	2	1	'	1	1			float	4	3
	1 '	1	1			'				3	1
CVT(T)SD2SI	r32,xmm				1				float	3	
CVT(T)SD2SI	r32,m64	1	1		1	1			float		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
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	1 1	1	1	1		1			float		1
DIVSS	xmm,m128	-				'				6 10 4)	•
	xmm,xmm	1	1	1					float	6-18 d)	5-17 d)
DIVSS	xmm,m32	1	1	1		1			float	0 00 4/	5-17 d)
DIVSD	xmm,xmm	1	1	1					float	6-32 d)	5-31 d)
DIVSD	xmm,m64	1	1	1		1			float		5-31 d)
DIVPS	xmm,xmm	1	1	1					float	6-18 d)	5-17 d)
DIVPS	xmm,m128	1	1	1		1			float		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
U.C.C.D WIII (CO/D	**************************	•		I	' '	I I	1 1	I	oat	-	•

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:

Round divisors give low values. SSE3 instruction set. d)

g)

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

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.

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

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	fused	μορs p015		p2	p4	Unit	•	Reci- procal through- put
Move instructions									

MOV a) MOV a) MOV MOV MOV MOV MOV MOV MOV MOV MOVSX MOVZX MOVSXD MOVSX MOVZX MOVSX MOVSXD CMOVCC CMOVCC CMOVCC XCHG XLAT PUSH PUSH PUSH PUSH PUSH PUSH PUSH PUSH	r,r/i r,m m,r m,r m,r m,sr sr,r sr,m m,r r,r r16/32,m r64,m r,r r,m r,m r,m r,m r,m r,m r m m m m	1 1 1 1 1 1 1 2 8 8 2 1 1 2 2 2 3 7 2 1 1 1 2 1 2 1 1 1 1 2 1 2 1 1 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 2 2 4 2 2 2 2	1 1 4 3 1 1 2 2 3 x 1 1 15 9 3 9 23 2 1 2 1 2 11 2 11	x x x x x x x 1 1	x	x x x x x x x 1 1	1 1 1 1 1 1 1 1 1 8 8	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 2 3 3 3 3 1 1 2 2 high b) 4 3 2 2 20 1 4 1 4 1 4	0,33 1 1 1 1 1 16 16 2 0,33 1 1 1 1 7 8 1 1,5 17 7 0,33 1 1 1 1 8 6 9 90
IN OUT Arithmetic instructions		4	2	'		, ,		'	'		
ADD SUB ADD SUB ADD SUB ADC SBB ADC SBB ADC SBB CMP CMP	r,r/i r,m m,r/i r,r/i r,m m,r/i r,r/i m,r/i	1 1 2 2 2 4 1	1 1 1 2 2 3 1	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 6 2 2 7 1	0,33 1 1 2 2 0,33 1

INC DEC NEG NOT	r	1	1	х	Х	х				1	0,33
INC DEC NEG NOT	m	3	1	Х	Х	х	1	1	1	6	1
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	х				5	1,5
MUL IMUL	r32	3	3	Х	Х	х				5	1,5
MUL IMUL	r64	3	3	Х	Х	х				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,r16,i	1	1		1					3	1
IMUL	r32,r32,i	1	1		1					3	1
IMUL	r64,r64,i	1	1	1						5	2
MUL IMUL	m8	1	1		1		1			3	1
MUL IMUL	m16	3	3	х	Х	х	1			5	1,5
MUL IMUL	m32	3	3	Х	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	х				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	х	х	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	х	х	x	1			14-23 c)	
DIV	m64	32	31	х	х	x	1			34-88 c)	
IDIV	m64	56	55	х	х	x	1			39-72 c)	
CBW CWDE CDQE		1	1	х	x	×				1	
CWD CDQ CQO		1	1	х		x				1	
Logic instructions											
AND OR XOR	r,r/i	1	1	Х	Х	x				1	0,33
AND OR XOR	r,m	1	1	Х	Х	х	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		x				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	х				11	
RCR RCL	m,1	4	3	х	х	х	1	1	1	7	

RCR RCL RCR RCL SHLD SHRD SHLD SHRD BT BT BT BT BTR BTS BTC BTR BTS BTC BTR BTS BTC BTR BTS BTC CBTR BTS BTC CC BSF BSR SETCC SETCC CLC STC CMC CLD STD	m8,i/cl m8,i/cl m8,i/cl r,r,i/cl r,r,i/cl r,r/i m,r m,i r,r/i m,r r,r/i m,r m,i r,r m,i r,r m,i r,r	12 11 10 2 3 1 9 3 1 10 3 2 1 2 1 6 6	9 8 7 2 1 8 2 1 7 1 2 2 1 1 6 6	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	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 1 1	14 13 13 2 7 1 1 5 6 2	1 1 4 1 1 1 0,33 3 14
Control transfer instructi JMP JMP i) JMP JMP JMP Conditional jump Fused compare/test and bi J(E/R)CXZ LOOP LOOP(N)E CALL CALL i) 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 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 1 30 30 13 5	x x x	x x x	1	1 2 1 1 2 2 2 2	1 1 1	1 1 1	0 0 0 0	1-2 76 1-2 1-2 68 1 1-2 5 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		3 4+7n-1 4 8+5n-2 8 1 7+7n-1 4 7+8n-1	2 0+1.2 5 1 3+n 3	 - 1		5	1 1 2	 1 	 1 	1+5n-2 ⁻ 7+2n-0. 1+3n-0. 3+8n-23	1 55n 63n 1

REP(N)E CMPS		7+10n-	7+9n	l	I	I	I	I	l	2+7n-22	2+5n
Other NOP (90) Long NOP (0F 1F) PAUSE ENTER ENTER	i,0 a,b	1 1 3 12	1 1 3 10	x x x	x x x	x x x		1	1		0,33 1 8 8
LEAVE CPUID RDTSC RDPMC	4,5	3 53-117 13 23	2				1				53-211 32 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. 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)

Not available in 64 bit mode. i)

Floating point x87 instructions

Instruction	Operands	μοps fused	d C							Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	р5	p2	p3	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	X	x	X	7	23	23	float		142

FDIV(R)(P)	STOR	m	78	51	х	х	x	27	float		177	
FADD(P) FSUB(R)(P) r 1	thmetic instructions											
FADD(P) FSUB(R)(P)		r	1	1		1			float	3	1	
FMUL(P)		m	1	1		1		1	float		1	
FMUL(P)		r	1	1	1				float	5	2	
FDIV(R)(P)	` '	m	1	1	1			1	float		2	
FDIV(R)(P)	` '	r	1	1	1				float	6-21 d)	5-20 d)	
FABS FCHS FCHS FCOM(P) FUCOM FCOM(P) FUCOM FCOM(P) FUCOM FCOM(P) FUCOM FCOM(P) FUCOM FCOM(P) FUCOM(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) FCOMI(P) FUCOMI(P) ' ' '	m	1	1	1			1	float	6-21 d)	5-20 d)		
FCOM(P) FUCOM			1	1	1				float			
FCOM(P) FUCOM	HS		1	1	1				float	1	1	
FCOM(P) FUCOM	OM(P) FUCOM	r	1	1		1			float		1	
FCOMI(P) FUCOMI(P) r 1 2 2 2	` '	m	1	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 float 6-21 5-20 FICOM(P) m 2 2 2 1 float 6-21 5-20 FTST 1 1 1 1 float 2 2 FXAM 1 1 1 1 float 1 1 FPREM 26-29 x x x x float 13-40 FPREM1 28-35 x x x x float 10-22 Math 28 28 x x x float 43 FSCALE 53-84 x x x float ~170	OMPP FUCOMPP		2	2	1	1			float			
FIMUL m 2 2 1 1 1 float 5 2 FIDIV(R) m 2 2 1 1 1 float 6-21 5-20 FICOM(P) m 2 2 2 1 float 6-21 5-20 FTST 1 1 1 1 float 2 2 FXAM 1 1 1 1 float 1 1 FPREM 26-29 x x x x float 13-40 FPREM1 28-35 x x x x float 10-22 Math 28 28 x x x x float 43 FXTRACT 53-84 x x x x float ~170	OMI(P) FUCOMI(P)	r	1	1		1			float		1	
FIMUL m 2 2 1 1 1 float 5 2 FIDIV(R) m 2 2 1 1 1 float 6-21 5-20 FICOM(P) m 2 2 2 1 float 6-21 5-20 FTST 1 1 1 1 float 2 2 FXAM 1 1 1 1 float 1 1 FPREM 26-29 x x x x float 13-40 FPREM1 28-35 x x x x float 10-22 Math 17-19 x x x x float 43 FSCALE 28 28 x x x float ~170	.DD FISUB(R)	m	2	2		2		1	float	3	2	
FICOM(P) FTST 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		m	2	2	1	1		1	float	5	2	
FICOM(P) m 2 2 2 1 float	IV(R)	m	2	2	1	1		1	float	6-21	5-20 d)	
FTST 1	` '	m	2	2		2		1	float			
FPREM 26-29 x	` '		1	1		1			float		1	
FPREM1	4M		1	1		1			float		1	
The image	REM		26-29	'	Х	х	х		float	13-40		
Math 28 28 x x x x float 43 FXTRACT 53-84 x x x x math x <td>REM1</td> <td></td> <td>28-35</td> <td></td> <td>Х</td> <td>х</td> <td>х</td> <td></td> <td>float</td> <td>18-41</td> <td></td> <td></td>	REM1		28-35		Х	х	х		float	18-41		
FSCALE 28 28 x x x	NDINT		17-19	I	х	х	х		float	10-22		
FSCALE 28 28 x x x	th											
FXTRACT 53-84 x x x float ~170			28	28	Х	Х	x		float	43		
			1				x					
			1	1	1				float	6-20		
FSIN 18-85 x x x x float 32-85			18-85	I	Х	х	х		float			
FCOS 76-100 x x x	os		76-100		х	х	х		float	70-100		
18-			18-									
FSINCOS 105 x x x x 107	NCOS		105		х	х	х		float	38-107		
F2XM1 19 19 x x x x float 45	KM1		19	19	Х	х	х		float	45		
FYL2X FYL2XP1 57-65 x x x x float 50-100	_2X FYL2XP1		57-65		Х	Х	х		float	50-100		
FPTAN 19-100 x x x float 40-130	ΓΑΝ		19-100		Х	х	х		float	40-130		
FPATAN 23-87 x x x	ATAN		23-87	I	х	х	х		float	55-130		
Other	ner											
FNOP 1 1 1 1 float 1	OP		1	1	1				float		1	
WAIT 2 2 x x x float 1	JT		2	2	Х	х	x		float		1	
FNCLEX 4 4 x x float 15	CLEX		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	µops	μορs unfused domain	Unit	Laten-	Reci-
		fused			су	procal

		do- main	p015	p0	p1	p5	p2	рЗ	p4			through- put
Move instructions												
MOVD k)	r,(x)mm	1	1	Х	Х	x				int	2	0,33
MOVD k)	m,(x)mm	1						1	1		3	1
MOVD k)	(x)mm,r	1	1	Х		x				int	2	0,5
MOVD k)	(x)mm,m	1					1			int	2	1
MOVQ	v,v ´	1	1	Х	x	x				int	1	0,33
MOVQ	(x)mm,m64	1					1			int	2	1
MOVQ	m64, (x)mm	1					-	1	1		3	1
MOVDQA	xmm, xmm	1	1	x	X	X		•	-	int	1	0,33
MOVDQA	xmm, m128	1	'		``		1			int	2	1
MOVDQA	m128, xmm	1					'	1	1		3	1 1
MOVDQU	m128, xmm	9	4	x	x	X	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		x	2			int	2-8	2
MOVDQ2Q	'	1	1				-					
	mm, xmm	1 1	1	X	X	X				int	1	0,33
MOVQ2DQ	xmm,mm		'	X	X	X		1	1	int	I	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												
PACKUSWB	mm,m64	1	1	1			1			int		1
PACKSSWB/DW												
PACKUSWB	xmm,xmm	1	1			1				int	1	1
PACKSSWB/DW												
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			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/ZXVQ j)	xmm,xmm	1	1			1	'			int	1	1
PMOVSX/ZXDQ j)	1	1				1	1			int	'	1
• ,	xmm,m64						'				1	1
PSHUFB h)	mm,mm	1	1			1	4			int	'	
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													
PSHUFD		mm,mm,i		1			1				int	1	1
PSHUFL/HW	PSHUFW	mm,m64,i	2	1			1	1			int		1
PSHUFL/HW	PSHUFD	xmm,xmm,i	1	1			1				int	1	1
PSHUFL/HW	PSHUFD	xmm,m128,i	2	1			1	1			int		1
PALIGNR h)	PSHUFL/HW	xmm,xmm,i	1	1			1				int	1	1
PALIGNR h)	PSHUFL/HW	x, m128,i	2	1			1	1			int		1
PALIGNR h)	PALIGNR h)	mm,mm,i	2	2			2				int	2	1
PALLENDVB	PALIGNR h)	mm,m64,i	3	3			3	1			int		1
PBLENDVB j)	PALIGNR h)	xmm,xmm,i	1	1			1				int	1	1 1
PBLENDVB j)	PALIGNR h)	xmm,m128,i	1	1			1	1			int		1 1
PBLENDVB	PBLENDVB j)	x,x,xmm0	2	2			2				int	2	2
PBLENDW	PBLENDVB j)	x,m,xmm0	2	2			2	1			int		
PBLENDW		1 ' '	1	1			1				int	1	1 1
MASKMOVQ mm,mm 4 1 <t< td=""><td></td><td></td><td>1</td><td>1</td><td></td><td></td><td>1</td><td>1</td><td></td><td></td><td></td><td></td><td>1 1</td></t<>			1	1			1	1					1 1
MASKMOVDQU		1 1	4	1	1			1	1	1	int		2-5
PMOVMSKB		· · ·			1		3	2	2	3			
PEXTRB j)		· · ·		1	1							2	
PEXTRB j)		1 ' ' '	-		-	x	×						
PEXTRW r32,(x)mm,i 2 2 x x x 1 int 3 1 PEXTRW j) m16,(x)mm,i 2 2 2 7 1 1 1 int 3 1 PEXTRD j) r32,xmm,i 2 2 x x x x int 3 1 PEXTRD j) m32,xmm,i 2 2 x x x x int 3 1 PEXTRQ j,m) r64,xmm,i 2 2 x x x int 3 1 PEXTRQ j,m) m64,xmm,i 2 1 1 1 int 1 PEXTRQ j,m) m64,xmm,i 2 1 1 1 int 1 PINSRB j) xmm,m32,i 1 1 1 int 1 PINSRB j) xmm,m8,i 2 1 1 1 int 1 PINSRW (x)mm,m16,i 2 1 1 1 int 1 PINSRD j) xmm,m32,i 1 1 1 int 1 PINSRD j) xmm,m32,i 1 1 1 int 1 PINSRD j) xmm,m32,i 2 1 1 1 int 1 PINSRD j) xmm,m64,i 2 1 1 1 int 1 PINSRQ j,m) xmm,r64,i 1 1 1 int 1 PINSRQ j,m) xmm,m64,i 2 1 1 1 int 1 PADD/SUB(U)(S)B/W/D xmm,m64,i 2 1 1 1 int 1 PADDQ PSUBQ (x)mm,m 2 2 x x 1 int 1 PADDQ PSUBQ (x)mm,m 2 2 x x 1 int 2 PHADD(S)W PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) (x)mm,m64 4 3 1 2 1 int 1 2 PCMPEQ/GTB/W/D (x)mm,m64 4 3 1 2 1 int 1 1 PCMPEQQ j) xmm,xmm 1 1 x x x 1 int 1 1 PMULL/HW PMULHUW v,v 1 1 1 int 3 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 int 1 int 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 int 1 int 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 int int 1 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 int int 1 1 1 PMULHRSW h) v,v 1 1 1 int int 1 int 1 1 PMULHRSW h) v,v 1 1 1 int int 1 int 1 1 int 1 1 1 int 1 1 1 1 int 1 1 1 1 1 1 1 1 1	37	1 ' ' 1		1									1 1
PEXTRW j)	37			1				1					1 1
PEXTRD j)		1 '\ ' '		1				'	1	1			
PEXTRD j)	3,	1 ' '		1	1 -	-	'		'	'		3	
PEXTRQ j,m					^	^			1	1			
PEXTRQ j,m		1 1		1			-		'	'	-	3	
PINSRB j)				1	^	^			1	1		3	
PINSRB j)									'	'		1	
PINSRW								1					
PINSRW								'				2	
PINSRD j)		1 ' '	-	1			-	1				_	
PINSRD j)		1,,						'				1	1 1
PINSRQ j,m xmm,r64,i	37		-	1			-	4			-	'	
PINSRQ j,m		1 ' 1		1			-				-	4	· .
Arithmetic instructions v,v 1 1 x x int 1 0,5 PADD/SUB(U)(S)B/W/D v,v 1 1 x x 1 int 1 0,5 PADDQ PSUBQ v,v 2 2 x x 1 int 2 1 PADDQ PSUBQ (x)mm,m 2 2 x x 1 int 1 1 PHADD(S)W PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) v,v 1 1 x x int 3 2 PCMPEQ/GTB/W/D v,v 1 1 x x 1 int 1 0,5 PCMPEQQ j) xmm,mm1								4				1	· .
PADD/SUB(U)(S)B/W/D v,v 1 1 x x 1 int 1 0,5 PADD/SUB(U)(S)B/W/D (x)mm,m 1 1 x x 1 int 1 1 PADDQ PSUBQ v,v 2 2 x x 1 int 2 1 PADDQ PSUBQ (x)mm,m 2 2 x x 1 int 2 1 PHADD(S)W PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADD(S)W PHSUB(S)W h) (x)mm,m64 4 3 1 2 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 1 int 3 2 PHADDD PHSUBD h) v,v 1 1 x x int 1 0,5 PCMPEQ/GTB/W/D v,v 1 1 x x 1 1 1 1	PINSKQ J,III)	XIIIII,III04,I	2	'			'				ШЦ		
PADD/SUB(U)(S)B/W/D v,v 1 1 x x 1 int 1 0,5 PADD/SUB(U)(S)B/W/D (x)mm,m 1 1 x x 1 int 1 1 PADDQ PSUBQ v,v 2 2 x x 1 int 2 1 PHADD(S)W PHADD(S)W v,v 3 3 1 2 int 3 2 PHADD(S)W h) v,v 3 3 1 2 int 3 2 PHADD(S)W h) (x)mm,m64 4 3 1 2 int 3 2 PHADD PHSUBD h) v,v 3 3 1 2 1 int 3 2 PHADDD PHSUBD h) v,v 1 1 x x int 1 0,5 PCMPEQ/GTB/W/D v,v 1 1 x x 1 int 1 0,5 PCMPEQQ j)	A rith motio in atrustions												
PADD/SUB(U)(S)B/W/D (x)mm,m 1 1 x x 1 int 1 1 x x 1 int 2 1 1 1 x x 1 int 2 1 1 1 x x 1 int 2 1 1 int 2 1 int 3 2 2 2 2 2 2 2 2 3 3 1 2 2 int 3 2 2 2 2 3 3 1 2 2 3 3 1 2 1 3 2 2 3 3 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1		- ,,,	1	4	\ \						int	4	0.5
PADDQ PSUBQ v,v 2 2 x x int 2 1 PADDQ PSUBQ (x)mm,m 2 2 x x 1 int 1 PHADD(S)W PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADD(S)W PHADD(S)W (x)mm,m64 4 3 1 2 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) (x)mm,m64 4 3 1 2 1 int 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0,5 PCMPEQQ j) xmm,xmm 1 1 1 int 1 1 PMULL/HW PMULHUW v,v 1 1 1 int 3 1 PMULHRSW h) v,v 1 1 1 1 int <td></td> <td></td> <td>=</td> <td></td> <td></td> <td></td> <td></td> <td>4</td> <td></td> <td></td> <td>-</td> <td>'</td> <td></td>			=					4			-	'	
PADDQ PSUBQ (x)mm,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) (x)mm,m64 4 3 1 2 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) (x)mm,m64 4 3 1 2 int 3 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0,5 PCMPEQQ j) xmm,xmm 1 1 1 int 1 1 PCMPEQQ j) xmm,m128 1 1 1 int 3 1 PMULL/HW PMULHUW v,v 1 1 1 int 3 1 PMULHRSW h) v,v 1 1 1 int 3 1	, , , ,	` '	-	1							-	2	· .
PHADD(S)W PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADD(S)W PHSUB(S)W h) (x)mm,m64 4 3 1 2 1 int 3 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) (x)mm,m64 4 3 1 2 int 3 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0,5 PCMPEQ/GTB/W/D (x)mm,m 1 1 1 int 1 1 PCMPEQQ j) xmm,xmm 1 1 1 int 1 1 PCMPEQQ j) xmm,m128 1 1 1 int 3 1 PMULL/HW PMULHUW v,v 1 1 1 int 3 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 int 3 1				1				4			-		
PHSUB(S)W h) v,v 3 3 1 2 int 3 2 PHADD(S)W PHSUB(S)W h) (x)mm,m64 4 3 1 2 1 int 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) (x)mm,m64 4 3 1 2 1 int 3 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0,5 PCMPEQ/GTB/W/D (x)mm,m 1 1 1 int 1 1 PCMPEQQ j) xmm,xmm 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 1 1		(X)!!!!!,!!!	2	2	\		X				ШЦ		
PHADD(S)W PHSUB(S)W h) (x)mm,m64 4 3 1 2 1 int 2 PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) (x)mm,m64 4 3 1 2 1 int 3 2 PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0,5 PCMPEQ/GTB/W/D (x)mm,m 1 1 x x 1 int 1 1 PCMPEQ/GTB/W/D (x)mm,mm 1 1 1 int 1 <td></td> <td></td> <td>2</td> <td>2</td> <td>1</td> <td></td> <td>2</td> <td></td> <td></td> <td></td> <td>int</td> <td>2</td> <td>2</td>			2	2	1		2				int	2	2
PHSUB(S)W h) (x)mm,m64 4 3 1 2 1 int 2 PHADDD PHSUBD h) v,v 3 3 1 2 1 int 3 2 PHADDD PHSUBD h) (x)mm,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 (x)mm,m 1 1 x x 1 int 1 1 PCMPEQ/GTB/W/D (x)mm,mm 1 1 1 int 1 1 PCMPEQQ j) xmm,xmm 1 1 1 int 1 1 PMULL/HW PMULHUW v,v 1 1 1 int 3 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 int 3 1 PMULHRSW h) v,v 1 1 1 int 3 1 <	` ' '	V,V	3	ا ا	'		~				IIIL	3	
PHADDD PHSUBD h) v,v 3 3 1 2 int 3 2 PHADDD PHSUBD h) (x)mm,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 (x)mm,m 1 1 x x 1 int 1 1 PCMPEQ/GTB/W/D xmm,xmm 1 1 1 int 1 <td< td=""><td></td><td>(v)mm m64</td><td>4</td><td>2</td><td>1</td><td></td><td>2</td><td>4</td><td></td><td></td><td>int</td><td></td><td>2</td></td<>		(v)mm m64	4	2	1		2	4			int		2
PHADDD PHSUBD h) (x)mm,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 (x)mm,m 1 1 x x 1 int 1 PCMPEQQ j) xmm,xmm 1 1 1 int 1 1 PMULL/HW PMULHUW v,v 1 1 1 int 3 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 int 3 1 PMULHRSW h) v,v 1 1 1 1 int 3 1	` ' '	` '		1	1 .			'			-	_	I
PCMPEQ/GTB/W/D v,v 1 1 x x int 1 0,5 PCMPEQ/GTB/W/D (x)mm,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 int 1 1 PMULL/HW PMULHUW v,v 1 1 1 int 3 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 int 3 1 PMULHRSW h) v,v 1 1 1 int 3 1	,			1	1 -			4				3	
PCMPEQ/GTB/W/D (x)mm,m 1 1 x x 1 int 1 PCMPEQQ j) xmm,xmm 1 1 1 1 int 1 1 PCMPEQQ j) xmm,m128 1 1 1 1 int 1 PMULL/HW PMULHUW v,v 1 1 1 int 3 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 1 int 1 PMULHRSW h) v,v 1 1 1 1 int 3 1	,	` '			'			1			-		
PCMPEQQ j) xmm,xmm 1			-	1				_			-	l I	
PCMPEQQ j) xmm,m128 1		1 ' '			X			1			-		
PMULL/HW PMULHUW v,v 1 1 1 1 int 3 1 PMULL/HW PMULHUW (x)mm,m 1 1 1 1 int 1 PMULHRSW h) v,v 1 1 1 int 3 1	37											1	
PMULL/HW PMULHUW (x)mm,m 1	3,						1	1					
PMULHRSW h) v,v 1 1 1 1 int 3 1			•	1		'						3	
		1 ' '	=					1			-		
PMULHRSW n) (x)mm,m 1 1 1 1 int 1	,		=								_	3	
	PMULHRSW h)	(x)mm,m	1	1		1		1			ınt		1

PAND(N) POR PXOR (x)mm,m 1 PTEST j) xmm,xmm 2 PSLL/RL/RAW/D/Q mm,mm/i 1 PSLL/RL/RAW/D/Q mm,m64 1 PSLL/RL/RAW/D/Q xmm,i 1 PSLL/RL/RAW/D/Q xmm,xmm 2 PSLL/RL/RAW/D/Q xmm,m128 3 PSLL/RLDQ xmm,i 1	1 1 2 2 1 1 1 2 2 1	x x 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	1 1 1	int int int int int int int int	1 1 1 1 2	0,33 1 1 1 1 1 1 1 1
PTEST j) xmm,xmm 2 PTEST j) xmm,m128 2 PSLL/RL/RAW/D/Q mm,mm/i 1 PSLL/RL/RAW/D/Q mm,m64 1 PSLL/RL/RAW/D/Q xmm,i 1 PSLL/RL/RAW/D/Q xmm,xmm 2 PSLL/RL/RAW/D/Q xmm,m128 3	1 2 2 1 1 1 2 2	x 1 1 1 1 1 x x	X X	x x x	1 1	int int int int int int int	1 1 1 2	1 1 1 1 1 1 1
PTEST j) xmm,xmm 2 PTEST j) xmm,m128 2 PSLL/RL/RAW/D/Q mm,mm/i 1 PSLL/RL/RAW/D/Q mm,m64 1 PSLL/RL/RAW/D/Q xmm,i 1 PSLL/RL/RAW/D/Q xmm,xmm 2 PSLL/RL/RAW/D/Q xmm,m128 3	1 2 2 1 1 1 2 2	x 1 1 1 1 1 x x	X X	x x x	1 1	int int int int int int int	1 1 1 2	1 1 1 1 1 1 1
PTEST j) xmm,xmm 2 PTEST j) xmm,m128 2 PSLL/RL/RAW/D/Q mm,mm/i 1 PSLL/RL/RAW/D/Q mm,m64 1 PSLL/RL/RAW/D/Q xmm,i 1 PSLL/RL/RAW/D/Q xmm,xmm 2	1 2 2 1 1 1 2	x 1 1 1 1 1 x	X X	x x x	1 1	int int int int int int int	1 1 1	1 1 1 1 1 1
PTEST j) xmm,xmm 2 PTEST j) xmm,m128 2 PSLL/RL/RAW/D/Q mm,mm/i 1 PSLL/RL/RAW/D/Q mm,m64 1 PSLL/RL/RAW/D/Q xmm,i 1	1 2 2 1 1 1	x 1 1 1 1	X X	x x x	1	int int int int int int	1 1 1	1 1 1 1 1
PTEST j) xmm,xmm 2 PTEST j) xmm,m128 2 PSLL/RL/RAW/D/Q mm,mm/i 1 PSLL/RL/RAW/D/Q mm,m64 1	1 2 2 1 1	x 1 1 1	X X	x x	1	int int int int int int	1	1 1 1 1 1
PTEST j) xmm,xmm 2 PTEST j) xmm,m128 2 PSLL/RL/RAW/D/Q mm,mm/i 1	1 2 2 1	x 1 1	X X	x x	1	int int int int int	1	1 1 1 1
PTEST j) xmm,xmm 2 PTEST j) xmm,m128 2	1 2 2	x 1 1	X X	x x		int int int int	1	1 1 1
PTEST j) xmm,xmm 2	1 2	x 1	X X	x x		int int int		1 1
	1	Х	х	Х	1	int int		1
PAND(N) POR PXOR (y)mm m 1					1	int	1	
	1	x	Y	×			1	0.33
PAND(N) POR PXOR v,v 1								
Logic instructions						1110		
IVII SADDVV J) XIIIII,III,I 4	J		1		1 1			
MPSADBW j) xmm,m,i 4	3		1	2	1	int	5	2
PSADBW (x)mm,m 1 MPSADBW j) xmm,xmm,i 3	1 3		1	2	1	int int	5	1 2
	.		.		1		٥	l .
	1	Х	1	Χ	1	int	3	1
PSIGND h) (x)mm,m 1	1	Ţ		x	1	int		1
PSIGND h) v,v 1 1 PSIGNB PSIGNW	1	Х		Х		int	1	0,5
PSIGNB PSIGNW PSIGND h) v.v 1	,					int	1	0.5
,	1	Х		Х	1	int		1
PABSB PABSW PABSD (x)mm.m 1	1				1	int		1
.,	1	Х		X		int	1	0,5
PHMINPOSUW j) xmm,m128 4 PABSB PABSW PABSD h) vv 1	4			-	1	int	1	
PHMINPOSUW j) xmm,xmm 4				4 4	,	int	4	4
PMIN/MAXUD j) xmm,m128 1	1 4	1		,	1	int	4	1 4
- 1/		1			4	int	'	
,,	1	1			1	int	1	1
	1	1			1		'	
1, , , , , , , , , , , , , , , , , , ,	1	1			1	int int	1	1
,	1	'			4		'	1 1
PMIN/MAXSW (x)mm,m 1 PMIN/MAXUW j) xmm,xmm 1	1	X 1		Х	1	int	1	1
	1				1	int	'	1
PMIN/MAXSW v,v 1	1	X		X	1	int	1	0,5
PMIN/MAXUB (x)mm,m 1	1	x		X	1	int	'	1
PMIN/MAXUB v,v 1	1	X		x	1	int	1	0,5
PMIN/MAXSB j) xmm,m128 1	1	1			1	int	'	
PMIN/MAXSB j) (x)min,m 1 1	1	1		^	'	int	1	
PAVGB/W (x)mm,m 1	1	x		x	1	int	'	1
PAVGB/W v,v 1	1	x	'	x	'	int	1	0,5
PMADDUBSW h) (x)mm,m 1	i		1		1	int		1
PMADDUBSW h) v,v 1	1		1		'	int	3	1
PMADDWD (x)mm,m 1	1		1		1	int		1
PMADDWD v,v 1	1		1		'	int	3	1
PMULUDQ (x)mm,m 1	1		1		1	int		
PMULUDQ v,v 1	1		1		'	int	3	1
PMULDQ j) xmm,m128 1	1		1		1	int	3	1
PMULDQ j) xmm,xmm 1	1	'	1	_	'	int	3	1
PMULLD j) xmm,xmm 4 PMULLD j) xmm,m128 6	4 5	1	2	2 2	1	int int	5 5	2 4

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	Operands	μορs fused	μops	un	fuse	ed d	loma	ain		Laten- cy	Reci- procal	
		do-	p015	p0	p1	р5	p2	рЗ	p4			through-
Move instructions		main										put
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	Х	1	2	2		3-4	4
MOVSS/D	xmm,xmm	1	1	Х	х	Х				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	1			2-3
SHUFPS	xmm,xmm,i	1	1			1				int	1	1
SHUFPS	xmm,m128,i	2	1			1	1			int		1
SHUFPD	xmm,xmm,i	1	1	1						float	1	1
SHUFPD	xmm,m128,i	2	1	1			1			float		1
BLENDPS/PD j)	xmm,xmm,i	1	1			1				int	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						int	1	1
MOVDDUP g)	xmm,m64	2	1	1			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	х				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

1	1	ı		1				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	
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	2	2	1	1		1		float		1	
CVT(T)PD2DQ	xmm,xmm	2	2	1	1				float	4	1	
CVT(T)PD2DQ	xmm,m128	2	2	1	1		1		float		1	
CVTPI2PS	xmm,mm	1	1		1				float	3	3	
CVTPI2PS	xmm,m64	1	1		1		1		float		3	
CVT(T)PS2PI	mm,xmm	1	1		1				float	3	1	
CVT(T)PS2PI	mm,m128	1	1		1		1		float		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		'		float	3	1	
CVT(T)SD2SI	r32,m64	1			1		1		float		1	
0 1 (1)00201	102,11104	'	'		'		'		iloat		'	
Arithmetic												
ADDSS/D SUBSS/D	xmm,xmm	1	1		1				float	3	1	
ADDSS/D SUBSS/D	x,m32/64		1		1		1		float		1	
ADDSS/D SUBPS/D	xmm,xmm	1			1		'		float	3	1	
ADDPS/D SUBPS/D	xmm,m128	1	1				1		float		1	
ADDSUBPS/D g)	xmm,xmm	1	1		1		'		float	3	1	
ADDSUBPS/D g)	xmm,m128	1	1				1		float	3	1	
HADDPS HSUBPS g)	xmm,xmm	3	3		1	2			float	7	3	
HADDPS HSUBPS g)	xmm,m128	4	3			2	1		float	'	3	
HADDPD HSUBPD g)	xmm,xmm	3	3		X		'		float	6	1,5	
	xmm,m128	4	3	X		X	1					
HADDPD HSUBPD g) MULSS	· · · · · · · · · · · · · · · · · · ·) 1	X	X	X	'		float float	4	1,5	
	xmm,xmm	1		1						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	6-13 d)	5-12 d)	
DIVSS	xmm,m32	1	1	1			1		float		5-12 d)	
DIVSD	xmm,xmm	1	1	1					float	6-21 d)	5-20 d)	
DIVSD	xmm,m64	1	1	1			1		float		5-20 d)	
DIVPS	xmm,xmm	1	1	1					float	6-13 d)	5-12 d)	

DIVPS	xmm,m128	1	1	1			1			float		5-12 d)
DIVPD	xmm,xmm	1	1	1			'			float	6-21 d)	5-20 d)
DIVPD	xmm,m128	1	1	1			1			float	0 2 1 u)	5-20 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	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		١.			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		١.			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		١.			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	X	X	x				float	9	3
DPPD j)	xmm,m128,i	4	4	X	X	X	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	Х	х	1			int		1
Other	_											
LDMXCSR	m32	13	12	Х	Х	х	1					38
STMXCSR	m32	10	8	Х	Х	х		1	1			20
FXSAVE	m4096	151	67	Х	Х	х	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, (x)mm = 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 independent instructions of the same kind in the same thread.

Integer instructions

Instruction	Operands	μορs fused	μops	un	fus	ed d	loma	ain		Do- main	Laten- cy	Reci- procal
		do-	p015	p0	p1	p5	p2	р3	p4			through-
Move instructions		main										put
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	Х	Х	Х		1	1	int		1
PUSHA(D) i)		18	2	Х	1	Х		8	8	int		8
POP	r	1					1			int	2	1
POP	(E/R)SP	3	2	Х	1	Х	1			int		5
POP	m	2					1	1	1	int		1
POP	sr	7	2				5			int		15
POPF(D/Q)		8	7	Х	Х	х	1			int		14
POPA(D) i)		10	2				8			int		8
LAHF SAHF		1	1	Х	Х	Х				int	1	0.33
SALC i)		2	2	x	х	Х				int	4	1
LEA a)	r,m	1	1		1					int	1	1
BSWAP	r32	1	1		1					int	1	1
BSWAP	r64	1	1		1					int	3	1
LDS LES LFS LGS LSS	m	9	3	x	х	Х	6			int		15
PREFETCHNTA	m	1					1			int		1

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

				l		I	I					
Logic instructions												
AND OR XOR	r,r/i	1	1	х	Х	Х				int	1	0.33
AND OR XOR	r,m	1	1	Х	Х	Х	1			int		1
AND OR XOR	m,r/i	2	1	х	Х	Х	1	1	1	int	6	1
TEST	r,r/i	1	1	Х	Х	Х				int	1	0.33
TEST	m,r/i	1	1	Х	Х	Х	1			int		1
SHR SHL SAR	r,i/cl	1	1	х		х				int	1	0.5
SHR SHL SAR	m,i/cl	3	2	х		х	1	1	1	int	6	1
ROR ROL	r,i/cl	1	1	х		х				int	1	1
ROR ROL	m,i/cl	3	2	х		х	1	1	1	int	6	1
RCR RCL	r,1	2	2	х	Х	х				int	2	2
RCR	r8,i/cl	9	9	х	Х	х				int	13	
RCL	r8,i/cl	8	8	х	Х	х				int	11	
RCR RCL	r16/32/64,i/cl	6	6	х	Х	х				int	12-13	12-13
RCR RCL	m,1	4	3	x	Х	Х	1	1	1	int	7	
RCR	m8,i/cl	12	9	x	х	х	1	1	1	int	16	
RCL	m8,i/cl	11	8	х	х	х	1	1	1	int	14	
RCR RCL	m16/32/64,i/cl	10	7	x	х	х	1	1	1	int	15	
SHLD	r,r,i/cl	2	2	х	х	Х				int	3	1
SHLD	m,r,i/cl	3	2	X	Х	Х	1	1	1	int	8	-
SHRD	r,r,i/cl	2	2	x	х	х				int	4	1
SHRD	m,r,i/cl	3	2	х	х	Х	1	1	1	int	9	
ВТ	r,r/i	1	1	х		Х				int	1	1
ВТ	m,r	9	8	х		Х	1			int		5
ВТ	m,i	2	2	х		Х	1			int		1
BTR BTS BTC	r,r/i	1	1	х		х				int	1	1
BTR BTS BTC	m,r	10	7	х	Х	х	1	1	1	int	6	
BTR BTS BTC	m,i	3	3	x		х	1	1	1	int	6	
BSF BSR	r,r	1	1		1					int	3	1
BSF BSR	r,m	2	1		1		1			int	3	1
SETcc	r	1	1	х		Х				int	1	1
SETcc	m	2	1	х	Х	х		1	1	int		1
CLC STC CMC		1	1	х	Х	х				int	1	0.33
CLD		2	2	Х	Х	Х				int		4
STD		2	2	Х	Х	Х				int		5
Control transfer instructi												
JMP	short/near	1	1			1				int	0	2
JMP i)	far	31	31							int		67
JMP	r	1	1			1				int	0	2
JMP	m(near)	1	1			1	1			int	0	2
JMP	m(far)	31	31				11			int		73
Conditional jump	short/near	1	1			1				int	0	2
Fused compare/test and b	· '	1	1			1				int	0	2
J(E/R)CXZ	short	2	2	Х	Х	1				int		2
LOOP	short	6	6	Х	Х	Х				int		4
LOOP(N)E	short	11	11	Х	Х	Х				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	1	int		2
CALL	m(far)	47	47				1			int		79

RETN RETN RETF RETF BOUND i) INTO i)	i i r,m	1 3 39 40 15 4	1 2 39 40 13 4			1 1	1 1 2			int int int int int int		2 2 120 124 7 5
String instructions												
LODS		2	1	x	х	х	1			int		1
REP LODS		11+4n	•	^	^	^	'	l	'	int	40+12n	· •
STOS		3	1	Y	¥	х		1	1	int	70.1211	1
REP STOS	small n	60+n	•	^	^	_ ^		' '	' '	int	12+n	'
REP STOS	large n	2.5/16	hvtes							int	1 clk / 1	6 bytes
MOVS	la go ii	5		x	х	x	1	1	1	int	l ont / i	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	3		Х	Х	х	1			int		1
REP SCAS		37+6n	l	ı	ll.	ļ	1	1	'	int	40+2n	l
CMPS		5	3	Х	Х	Х	2			int		4
REP CMPS		65+8n	ļ!	1		ı	1	1	'	int	42+2n	l
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.

e) SSE4.2 instruction set.

Floating point x87 instructions

Instruction	Operands	μοps fused	μops unfused domain							Do- main	Laten- cy	procal
		do-	p015	p0	p1	р5	р2	р3	p4			through-
Move instructions		main										put
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

			criaic	•••								
FSTP	m80	7	3	Х	Х	Х		2	2	float	5	5
FBSTP	m80	208	204	х	Х	х		2	2	float	242	245
FXCH	r	1	0 f)							float	0	1
FILD	m	1	1		1		1			float	6	1
FIST(P)	m m	3	1		1		ļ .	1	1	float	7	1
FISTTP g)		3	1		1			1		float	7	1
FLDZ	m m			4	'				'		'	
		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	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	X	Х	8	23	23	float	178	178
FRSTOR	m m	79	52	X	X	X	27		-0	float	156	156
INSTOR	'''	13	32	^	^	^	21			iloat	130	130
Arithmetic instructions												
FADD(P) FSUB(R)(P)		1	1		1					float	3	1
	r						4) J	
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	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	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 m	2	2		2		1			float	3	2
FIMUL	m m	2	2	1	1		1			float	5	2
FIDIV(R)	m m	2	2	1	1		1			float	7-27 d)	
		2	2	'	2		1				1-21 u)	•
FICOM(P)	m						'			float		1
FTST		1	1		1					float		1
FXAM		1	1		1					float		1
FPREM		25	25	Х	Х	X				float	14	
FPREM1		35	35	Х	Х	Х				float	19	
FRNDINT		17	17	Х	Χ	Х				float	22	
Math												
FSCALE		24	24	Х	Х	Х				float	12	
FXTRACT		17	17	Х	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	Х				float	~110	
F2XM1		19	19	X	X	X				float	58	
FYL2X FYL2XP1		~55	~55	X	X	X				float	~80	
FPTAN		~100	~100	x	X	X				float	~115	
FPATAN		~82	~82							float	~120	
FEATAIN		~02	~62	Х	X	X				แบลเ	~120	

Other										
FNOP		1	1	1			fl	loat	1	
WAIT		2	2	Х	Χ	Х	fl	loat	1	
FNCLEX		3	3		Χ	Х	fl	loat	17	
FNINIT	~	190	~190	х	Х	Х	fi	loat	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	μοps fused	μops	un	fus	ed d	oma	ain		Do- main	Laten- cy	Reci- procal
		do-	p015	р0	p1	р5	p2	р3	p4			through-
Move instructions		main										put
MOVD k)	r32/64,(x)mm	1	1	Х	Х	х				int	1+1	0.33
MOVD k)	m32/64,(x)mm	1						1	1		3	1
MOVD k)	(x)mm,r32/64	1	1	х	Х	Х				ivec	1+1	0.33
MOVD k)	(x)mm,m32/64	1					1				2	1
MOVQ	(x)mm, (x)mm	1	1	Х	Х	х				ivec	1	0.33
MOVQ	(x)mm,m64	1					1				2	1
MOVQ	m64, (x)mm	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	Х	Х	х				ivec	1	0.33
MOVQ2DQ	xmm,mm	1	1	Х	Х	х				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		Х	1					2
PACKUSDW j)	xmm,xmm	1	1	Х		Х				ivec	1	2
PACKUSDW j)	xmm,m	1	1	Х		Х	1					2
PUNPCKH/LBW/WD/DQ	(x)mm, (x)mm	1	1	х		х				ivec	1	0.5
PUNPCKH/LBW/WD/DQ	(x)mm,m	1	1	Х		х	1					2
PUNPCKH/LQDQ	xmm,xmm	1	1	х		Х				ivec	1	0.5
PUNPCKH/LQDQ	xmm, m128	2	1	х		Х	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		Х				ivec	1	1
PMOVSX/ZXBD j)	xmm,m32	1	1	х		Х	1					2

PMOVSX/ZXBQ j)	xmm,xmm	1	1	Х		Х				ivec	1	1
PMOVSX/ZXBQ j)	xmm,m16	1	1	х		Х	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	х		х	1					2
PSHUFB h)	(x)mm, (x)mm	1	1	x		Х				ivec	1	0.5
PSHUFB h)	(x)mm,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		X	1				-	1
PSHUFD	xmm,xmm,i	1	1	X		X	•			ivec	1	0.5
PSHUFD	xmm,m128,i	2	1	X		X	1			1,00		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)	(x)mm,(x)mm,i	1	1	x		X	'			ivec	1	1
PALIGNR h)	(x)mm,m,i	2	1	X		X	1			1000	'	1
PBLENDVB j)	x,x,xmm0	2	2	1		1	'			ivec	2	1
3,		3	2	1		1	1			IVEC		1
PBLENDVB j)	xmm,m,xmm0	1	1	1		_	!			ivoo	1	•
PBLENDW j)	xmm,xmm,i	2		X		X	4			ivec	ı	0.5
PBLENDW j)	xmm,m,i		1	X		Х	1	4	,			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,(x)mm	1	1	1						float	2+2	1
PEXTRB j)	r32,xmm,i	2	2	X	Х	Х				ivec	2+1	1
PEXTRB j)	m8,xmm,i	2	2	Х		Х						1
PEXTRW	r32,(x)mm,i	2	2	X	Х	Х				ivec	2+1	1
PEXTRW j)	m16,(x)mm,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	X		Х		1	1			1
PEXTRQ j,m)	r64,xmm,i	2	2	Х	Х	Х				ivec	2+1	1
PEXTRQ j,m)	m64,xmm,i	2	1	X		Х		1	1			1
PINSRB j)	xmm,r32,i	1	1	X		Х				ivec	1+1	1
PINSRB j)	xmm,m8,i	2	1	Х		Х	1					1
PINSRW	(x)mm,r32,i	1	1	Х		Х				ivec	1+1	1
PINSRW	(x)mm,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
Arithmetic instructions												
PADD/SUB(U)												
(S)B/W/D/Q	(x)mm, (x)mm	1	1	Х		Х				ivec	1	0.5
PADD/SUB(U)												
(S)B/W/D/Q	(x)mm,m	1	1	Х		Х	1					2
PHADD/SUB(S)W/D h)	(x)mm, (x)mm	3	3	Х		х				ivec	3	1,5
PHADD/SUB(S)W/D h)	(x)mm,m64	4	3	Х		х	1					3
PCMPEQ/GTB/W/D	(x)mm,(x)mm	1	1	Х		х				ivec	1	0.5
PCMPEQ/GTB/W/D	(x)mm,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
• • • • • • • • • • • • • • • • • • • •		l .	1	1			1	1			1	l

l-0	1 1			1	1 .	ı	1	1 1			
PCMPGTQ ()	xmm,xmm	1	1		1				ivec	3	1
PCMPGTQ ()	xmm,m128	1	1		1		1				1
PMULL/HW PMULHUW	(x)mm,(x)mm	1	1		1				ivec	3	1
PMULL/HW PMULHUW	(x)mm,m	1	1		1		1				1
PMULHRSW h)	(x)mm,(x)mm	1	1		1				ivec	3	1
PMULHRSW h)	(x)mm,m	1	1		1		1				1
PMULLD j)	xmm,xmm	2	2		2				ivec	6	2
PMULLD j)	xmm,m128	3	2		2		1				
PMULDQ j)	xmm,xmm	1	1		1				ivec	3	1
PMULDQ j)	xmm,m128	1	1		1		1				1
PMULUDQ	(x)mm,(x)mm	1	1		1				ivec	3	1
PMULUDQ	(x)mm,m	1	1		1		1				1
PMADDWD	(x)mm,(x)mm	1	1		1				ivec	3	1
PMADDWD	(x)mm,m	1	1		1		1				1
PMADDUBSW h)	(x)mm,(x)mm	1	1		1				ivec	3	1
PMADDUBSW h)	(x)mm,m	1	1		1		1				1
PAVGB/W	(x)mm,(x)mm	1	1	x		х			ivec	1	0.5
PAVGB/W	(x)mm,m	1	1	X		X	1			•	1
PMIN/MAXSB j)	xmm,xmm	1	1	X		X	'		ivec	1	1
PMIN/MAXSB j)	xmm,m128	1	1	X		X	1		1100	•	2
PMIN/MAXUB	(x)mm,(x)mm	1	1	X		X	l '		ivec	1	0.5
PMIN/MAXUB	(x)mm,m	1	1	x		X	1		1000	•	2
PMIN/MAXSW	(x)mm,(x)mm	1	1	x		X	'		ivec	1	0.5
PMIN/MAXSW	(x)mm,m	1	1	^		X	1		IVEC	ı	2
	' '	1	1				'		ivoo	1	1
PMIN/MAXUW j)	xmm,xmm	1	1	X		X	1		ivec	I	2
PMIN/MAXUW j)	xmm,m	1		X		X	'			4	
PMIN/MAXU/SD j)	xmm,xmm	-	1	X		X	,		ivec	1	1
PMIN/MAXU/SD j)	xmm,m128	1	1	X		Х	1			0	2
PHMINPOSUW j)	xmm,xmm	1	1		1				ivec	3	1
PHMINPOSUW j)	xmm,m128	1	1		1		1				3
PABSB PABSW PABSD	(-)(-)										0.5
h)	(x)mm,(x)mm	1	1	X		Х			ivec	1	0.5
PABSB PABSW PABSD	()										4
h)	(x)mm,m	1	1	X		Х	1				1
PSIGNB PSIGNW											
PSIGND h)	(x)mm,(x)mm	1	1	X		Х			ivec	1	0.5
PSIGNB PSIGNW											_
PSIGND h)	(x)mm,m	1	1	X		Х	1				2
PSADBW	(x)mm,(x)mm	1	1		1				ivec	3	1
PSADBW	(x)mm,m	1	1		1		1				3
MPSADBW j)	xmm,xmm,i	3	3	Х	Х	Х			ivec	5	1
MPSADBW j)	xmm,m,i	4	3	Х	Х	Х	1				2
PCLMULQDQ n)	xmm,xmm,i									12	8
AESDEC, AESDECLAST,											
AESENC, AESENCLAST											
n)											
	xmm,xmm									~5	~2
AESIMC n)	xmm,xmm									~5	~2
AESKEYGENASSIST n)	xmm,xmm,i									~5	~2
Logic instructions	, , , , ,										
PAND(N) POR PXOR	(x)mm,(x)mm	1	1	X	Х	Х			ivec	1	0.33
PAND(N) POR PXOR	(x)mm,m	1	1	X	Х	Х	1				1

PTEST j)	xmm,xmm	2	2	х	Х	Х		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 ()	xmm,xmm,i	8	8	х	Χ	Х		ivec	14	5
PCMPESTRI ℓ)	xmm,m128,i	9	8	Х	Χ	Х	1	ivec	14	6
PCMPESTRM ()	xmm,xmm,i	9	9	х	Χ	Х		ivec	7	6
PCMPESTRM ()	xmm,m128,i	10	10	х	Χ	Х	1	ivec	7	6
PCMPISTRI ()	xmm,xmm,i	3	3	Х	Χ	Х		ivec	8	2
PCMPISTRI ()	xmm,m128,i	4	4	х	Χ	Х	1	ivec	8	2
PCMPISTRM ℓ)	xmm,xmm,i	4	4	х	Χ	Х		ivec	7	2
PCMPISTRM ℓ)	xmm,m128,i	6	5	Х	Χ	Х	1	ivec	7	5
Other										
EMMS		11	11	Х	Χ	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

e) SSE4.2 instruction setm) Only available in 64 bit moden) Only available on newer models

Floating point XMM instructions

Instruction	Operands	µops fused	ed							Do- main	Laten- cy	Reci- procal
		do-	p015	p0	p1	р5	p2	р3	p4	1		through-
Move instructions		main										put
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)	xmm,m128,i	2 2	1 2			1 2	1			float float	2	1 2
BLENDVPS/PD j)	x,x,xmm0	3	2			2	4				2	2
BLENDVPS/PD j)	xmm,m,xmm0		1			1	1			float	4	1
MOVDDUP g)	xmm,xmm	1	I			'	_			float	1	_
MOVDDUP g)	xmm,m64	1					1			6 1 4	2	1
MOVSH/LDUP g)	xmm,xmm	1	1			1				float	1	1
MOVSH/LDUP g)	xmm,m128	1					1					1
UNPCKH/LPS/D	xmm,xmm	1	1			1				float	1	1
UNPCKH/LPS/D	xmm,m128	1	1			1	1			float		1
EXTRACTPS j)	r32,xmm,i	1	1			1				float	1+2	1
EXTRACTPS j)	m32,xmm,i	2	1			1		1	1			1
INSERTPS j)	xmm,xmm,i	1	1			1				float	1	1
INSERTPS j)	xmm,m32,i	3	2			2	1			float		2
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		1
CVTPI2PS	xmm,mm	1	1		1		'			float	3+2	3
CVTPI2PS	xmm,m64	1	1		1		1			float	0.2	3
CVT(T)PS2PI	mm,xmm	1	1		1		'			float	3+2	1
i_ i_ i	mm,m128						1			float	312	1
CVTDI2DD	· ·	1 2	1		1	4	1				6	1
CVTPI2PD	xmm,mm		2		1	1	4			ivec/float	6	
CVT/TV PD2DI	xmm,m64	2	2		1	1	1			G //	c	
CVT(T) PD2PI	mm,xmm	2	2	X	1	X	_			float/ivec	6	1
CVT(T) PD2PI	mm,m128	2	2	X	1	Х	1			6 1 4	0.0	1
CVTSI2SS	xmm,r32	1	1		1					float	3+2	3
CVTSI2SS	xmm,m32	1	1		1		1			float		3
CVT(T)SS2SI	r32,xmm	1	1		1					float	3+2	1
CVT(T)SS2SI	r32,m32	1	1		1		1			float		1
CVTSI2SD	xmm,r32	2	2	1	1					float	4+2	3
CVTSI2SD	xmm,m32	2	1		1		1			float		3
CVT(T)SD2SI	r32,xmm	1	1		1					float	3+2	1
CVT(T)SD2SI	r32,m64	1	1		1		1			float		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

A D D O I I D D O I D	I	1 4		I		l	1	ı		.		4
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		ļ '			float	3	2
RCPSS/PS	· ·	1	1		1		1			float		2
	xmm,m	'	'				'			iloat		2
CMPccSS/D CMPccPS/D	\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\	4	4		4					floot		4
	xmm,xmm	1	1		1					float	3	1
CMPccSS/D CMPccPS/D										<i>c</i>		
	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
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
ROUNDSS/D												
ROUNDPS/D j)	xmm,xmm,i	1	1		1					float	3	1
ROUNDSS/D												
ROUNDPS/D j)	xmm,m128,i	2	1		1		1			float		1
DPPS j)	xmm,xmm,i	4	4	1	2	1				float	11	2
DPPS j)	xmm,m128,i	6	5	Х	Х	х	1			float		
DPPD j)	xmm,xmm,i	3	3	Х	Х	Х				float	9	1
DPPD j)	xmm,m128,i	4	3	X	X	X	1			float		3
Di 1 D J)	XIIIII,III 120,I				^		'			nout		J
Math												
SQRTSS/PS	xmm,xmm	1	1	1						float	7-18	7-18
		2	-	[4			float	7-10	
SQRTSS/PS	xmm,m		1	1			1				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	х	х	1					5
STMXCSR	m32	2	1	^	^	1	'	1	1			1
	m4096	141	141	v	v	-	F	38	38		90	90
FXSAVE				X	X	X	5	30	၂၁၀		90	
FXRSTOR	m4096	112	90	Х	Χ	X	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, (x)mm = 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	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											
MOV	r,r/i	1	1	Х	Х	х			1		

MOV				- ,	- 5							
MOV	MOV	r,m	1					1		2	0.5	dressing
MOV												modes
MOVNTI										3		
MOVSX MOVZX		m,i							1 1			
MOVSX		m,r						1	1	~350	1	
MOVSXD		r,r	1	1	X	Х	Х			1		
CMOVcc		r,m	1					1			0.5	
CMOVcc	CMOVcc	r,r	2	2	X	х	х			2	1	
XCHG	CMOVcc		2	2	X	х	х	1			1	
XCHG				3	×	х	x			2		
XLAT PUSH PUSH r 1 PUSH pUSH m 2 PUSH(D/Q) PUSHA(D) POP (E/R)SP 1 POP (E/R)SP 1 POPF(D/Q) POP (E/R)SP 1 POP M 2 POPA(D) M 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1								2	1		-	implicit
PUSH		.,						_	•			
PUSH			3	2				1				
PUSHF(D/Q)			1					1	1	3		
PUSHF(D/Q) 13 2 x x x 1 <td< td=""><td> </td><td>i</td><td></td><td></td><td></td><td></td><td></td><td>1</td><td>1</td><td></td><td>1</td><td></td></td<>		i						1	1		1	
PUSHA(D) POP	PUSH	m	2					2	1		1	
POP (E/R)SP 1 0 1 1 2 0.5 POP (E/R)SP 1 0 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	PUSHF(D/Q)		3	2	X	Х	х	1	1		1	
POP POP POP POP POP POP POPF (D/Q) (E/R)SP m 1 0 0 9 8 x x x x 1 1 2 1 1 18 18 9 not 64 bit 18 10 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	PUSHA(D)		16	0				8	8		8	not 64 bit
POP (D/Q) m 2 2 1	POP	r	1					1		2	0.5	
POP (D/Q) m 2 2 1	POP	(E/R)SP	1	0				1			0.5	
POPF(D/Q)			2					2	1			
POPA(D)				8	×	x	x		•			
LAHF SAHF SALC LEA IT IT IT IT IT IT IT IT IT I	1 '				^			-				not 64 bit
SALC										1		
LEA r,m 1 1 x x 1 1 0.5 simple complex or rip relative BSWAP r32 1												not 64 hit
LEA		r m			_	v						
BSWAP BSWAP RF64 RF64 RFEFETCHNTA RFEFETCHT0/1/2 RFEFETCHT0/1/2 RFENCE RFFENCE R					^							
BSWAP BSWAP RFEFETCHNTA RFEFETCHTO/1/2 RFEFETCHTO/1/2 RFENCE RFENCE RFENCE RFIN		1,111		'		'				0	'	
BSWAP REFETCHNTA RM												
BSWAP REFETCHNTA RM	BSWAP	r32	1	1		1				1	1	
PREFETCHNTA m 1 1 1 0.5 PREFETCHT0/1/2 m 1 1 0.5 LFENCE 2 1 1 4 MFENCE 3 1 1 1 4 MFENCE 2 1 1 1 33 SFENCE 2 1 1 1 33 1 1 1 1 1 4 33 1 <				2								
PREFETCHT0/1/2 m 1 1 0.5 LFENCE 2 1 1 1 4 MFENCE 3 1 1 1 1 33 SFENCE 2 1 1 1 1 33 ADD SUB r,r/i 1 1 x x x 1 0.5 ADD SUB r,m 1 1 x x x 1 0.5 ADD SUB m,r/i 2 1 x x x 2 1 6 SUB m,r/i 2 1 x x x 2 1 6 1 SUB r,same 1 0 0.25				-				1		_		
LFENCE												
MFENCE 3 1 1 1 1 1 1 33 6 Arithmetic instructions ADD SUB r,r/i 1 1 x x x 1 0.5 ADD SUB r,m 1 1 x x x 1 0.5 ADD SUB m,r/i 2 1 x x x 2 1 6 1 SUB m,r/i 2 1 x x x 2 1 6 1 SUB r,same 1 0 0.25 0 0.25 0 0.25 ADC SBB r,r/i 2 2 x x x 2 1 1 ADC SBB m,r/i 4 3 x x 2 1 7 1,5 CMP m,r/i 1 1 x x x 1 1 0.5 INC DE		'''							1			
Arithmetic instructions r,r/i 1 2 2 1 1 1 2 2 1 2 2 1 3 2 2 1 3 3 3 3 3 3 3 3 3 3 3 4 3 3 4 3 3 4 3 3 4 3 4 4 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4<				1								
ADD SUB r,r/i 1 1 x x x 1 0.5 ADD SUB r,m 1 1 x x x 1 0.5 ADD SUB m,r/i 2 1 x x 2 1 6 1 SUB n,r/i 2 2 x x 2 1 6 1 ADC SBB r,r/i 2 2 x x 1 2 1 ADC SBB m,r/i 4 3 x x 2 1 7 1,5 CMP r,r/i 1 1 x x 1 1 0.5 INC DEC NEG NOT r 1 1 x x 1 1 0.5				'								
ADD SUB r,r/i 1 1 x x 1 0.5 ADD SUB r,m 1 x x 1 0.5 ADD SUB m,r/i 2 1 x x 2 1 6 1 SUB m,r/i 2 1 0 0 0.25 ADC SBB r,r/i 2 2 x x 1 2 1 ADC SBB m,r/i 4 3 x x 2 1 7 1,5 CMP r,r/i 1 1 x x 1 1 0.5 INC DEC NEG NOT r 1 1 x x 1 1 0.5	Autthorage to the state of											
ADD SUB r,m 1 1 x x x 1 0.5 ADD SUB m,r/i 2 1 x x 2 1 6 1 SUB r,same 1 0 0 0.25 ADC SBB r,r/i 2 2 x x 1 2 1 ADC SBB r,m 2 2 x x 1 2 1 ADC SBB m,r/i 4 3 x x 2 1 7 1,5 CMP r,r/i 1 1 x x 1 1 0.5 INC DEC NEG NOT r 1 1 x x x 1 1 0.5		"		4						4		
ADD SUB m,r/i 2 1 x x 2 1 6 1 SUB r,same 1 0 0 0.25 ADC SBB r,r/i 2 2 x x 2 1 ADC SBB r,m 2 2 x x 1 2 1 ADC SBB m,r/i 4 3 x x 2 1 7 1,5 CMP r,r/i 1 1 x x x 1 1 0.5 INC DEC NEG NOT r 1 1 x x x 1 1 0.5								_		1	2.5	
SUB r,same 1 0 0 0 0.25 ADC SBB r,r/i 2 2 x x x 2 1 ADC SBB r,m 2 2 x x x 1 2 1 ADC SBB m,r/i 4 3 x x 2 1 7 1,5 CMP r,r/i 1 1 x x x 1 0.5 INC DEC NEG NOT r 1 1 x x x 1 1			-	-						•		
ADC SBB ADC SBB r,r/i ADC SBB r,m 2 2 2 x x x 1 2 1 ADC SBB ADC SBB R,r/i ADC SBB R,r/				-	X	X	Х	2	1		Ī -	
ADC SBB r,m 2 2 x x x 1 2 1 ADC SBB m,r/i 4 3 x x x 2 1 7 1,5 CMP r,r/i 1 1 x x x x 1 1 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 1												
ADC SBB					Х	Х	Х					
CMP r,r/i 1 1 x x x 1 1 0.5 CMP m,r/i 1 1 x x 1 1 0.5 INC DEC NEG NOT r 1 1 x x 1 1					Х	Χ	Х				Ī -	
CMP		m,r/i	4	3	Х	Х	Х	2	1	7	1,5	
INC DEC NEG NOT r 1 1 x x x x 1	CMP	r,r/i	1	1	Х	Х	Х			1		
INC DEC NEG NOT r 1 1 x x x 1	CMP	m,r/i	1	1	Х	Х	Х	1		1	0.5	
INC DEC NEG NOT m 3 1 x x x 2 1 6 2	INC DEC NEG NOT		1	1	Х	Х	Х			1		
	INC DEC NEG NOT	m	3	1	x	х	х	2	1	6	2	

			- ,	- 3 -							
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	1100 0 1 510
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		O	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		'		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						21-25	11-14	
DIV	r32	10	10						20-28	11-18	
DIV	r64	34-56	х						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-	Х						40-103	25-84	
		138									
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	00540
POPCNT	r,r	1	1		1		_		3	1	SSE4.2
POPCNT	r,m	1	1		1		1		2	1	SSE4.2
CRC32 CRC32	r,r	1	1		1		1		3	1 1	SSE4.2 SSE4.2
CRU32	r,m	ı	1		ı		I			ı	33E4.2
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	х	х	Х			1		
TEST	m,r/i	1	1	Х	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	

			- ,	- 5							
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									5	5	
	r,i	8	8						5		
RCR	m,i	11	7				Х	X	_	6	
RCR	r,cl	8	8						5	5	
RCR	m,cl	11	7				Х	Х		6	
RCL	r,1	3	3						2	2	
RCL	r,i	8	8						6	6	
RCL	m,i	11	7				Х	х		6	
RCL	r,cl	8	8						6	6	
RCL	m,cl	11	7				х	x		6	
SHRD SHLD	r,r,i	1	1					``		0.5	
SHRD SHLD	m,r,i	3	•				2	1		2	
SHRD SHLD		4	4				_	'	2	2	
	r,r,cl						2	4	2	4	
SHRD SHLD	m,r,cl	5	3				2	1	4		
BT	r,r/i	1	1						1	0.5	
ВТ	m,r	10	8				Х			5	
ВТ	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	х		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	
SETcc	r	1	1	X		Х			1	0.5	
SETcc	m m	2	1	X		X	1	1	•	1	
CLC	""	1	0	^		^	'	' '		0.25	
STC CMC					.,	.,			4	0.25	
		1	1	X	Х	Х			1	_	
CLD STD		3	3							4	
Control transfer instruct											
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 not
											jumping
Fused arithmetic and		1	1			1			0	1-2	
branch											
J(E/R)CXZ	short	2	2	x	Х	1				2-4	
LOOP	short	7	7							5	
LOOP(N)E	short	11	11							5	
CALL	near	3	2			1	1	1		2	
CALL		2	1			1	1	1		2	
	r					_					
CALL	m	3	2			1	2	1		2	
RET		2	2			1	1			2	
RET	i	3	2			1	1			2	
BOUND	r,m	15	13							7	not 64 bit
INTO		4	4							6	not 64 bit
String instructions]										
LODS		3	2				1			1	

REP LODS STOS REP STOS		5n+12 3 2n	1		1	1	~2n n	1	worst case
REP STOS		1.5/16E	3				1/16B		best case
MOVS REP MOVS		5 2n					1.5 n	4	worst case
REP MOVS		3/16B					1/16B		best case
SCAS REP SCAS		3 6n+47					2n+45	1	
CMPS REP CMPS		5 8n+80					2n+80	4	
		011.00					211.00		
Other NOP (90)		1	0					0.25	
Long NOP (0F 1F)		1	0					0.25	decode only 1
									per clk
PAUSE	_	7	7		_			11	
ENTER ENTER	a,0	12	10		2	1	04.26	8	
LEAVE	a,b	49+6b 3	3		1		84+3b	7	
CPUID		31-75	J		'		100-250	,	
RDTSC		21						28	
RDTSCP		23						36	
RDPMC		35						42	

Floating point x87 instructions

Instruction	Operands	μops	μops	un	fuse	ed d	loma	in	Latency	Reci-	Com-
		fused do- main	p015	p0	p1	p5	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 FNSTSW FLDCW FNSTCW FINCSTP FDECSTP FFREE(P) FNSAVE FRSTOR	AX m16 m16 m16 r m	2 3 2 1 1 143 90	2 1 2 1 1 1	1 1			1 1 1	1	2 8 5 1	1 1 1 1 1 166 165	
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 m m	1 2 1 1 1 1 1 1 2 3 2 2 2 2 2 2 41-87 17	1 2 1 1 1 1 1 1 1 2 3 2 2 2 2 1 2 28 17	1 1 1 1 1 1 1 1	1 1 1 1 1 1 2 1 1 1	1	1 1 1 1 1 1 1		3 5 10-24 1 1 3 4	1 1 1 10-24 10-24 1 1 1 1 1 2 1 2 21 26-50	
Math FSCALE FXTRACT FSQRT FSIN FCOS FSINCOS F2XM1 FYL2X FYL2XP1 FPTAN FPATAN		27 17 1 64-100 20-110 20-110 53-118	27 17 1 x x x x	1					12 10 10-24 47-100 47-115 43-123 61-69 130 93-146		
Other FNOP WAIT FNCLEX FNINIT		1 2 5 26	1 2 5 26	1						1 1 22 81	

Instruction	Operands	µops	μops	un	fuse	ed d	oma	in	Latency	Reci-	Com-
		fused do- main	p015				p23			procal through- put	ments
Move instructions											
MOVD	r32/64,(x)mm	1	1	Х	Х	Х			1		
MOVD	m32/64,(x)mm	1					1	1	3	1	
MOVD	(x)mm,r32/64	1	1	Х	Х	Х			1		
MOVD	(x)mm,m32/64	1					1		3	0.5	
MOVQ	(x)mm,(x)mm	1	1	Х	Х	Х			1		
MOVQ	(x)mm,m64	1					1		3	0.5	
MOVQ	m64, (x)mm	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	,										
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	(x)mm,(x)mm	1	1		х	Х			1	0.5	
PUNPCKH/LBW/WD/DQ	(x)mm,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		X	X			1	0.5	SSE4.1
PMOVSX/ZXDQ	x,m64	1	1		X	X	1			0.5	SSE4.1
PSHUFB	(x)mm,(x)mm	1	1		X	X			1	0.5	SSSE3
PSHUFB	(x)mm,m	2	1		X	X	1			0.5	SSSE3
PSHUFW	mm,mm,i	1	1		X	X	'		1	0.5	
PSHUFW	mm,m64,i	2	1		X	X	1			0.5	

PSHUFD	x,x,i	1	1		х	х			1	0.5	
PSHUFD	x,m128,i	2	1		X	X	1		ı	0.5	
PSHUFL/HW	X,111126,1 X,X,İ	1	1		X	X	'		1	0.5	
PSHUFL/HW	x, m128,i	2	1		X	X	1		ı	0.5	
PALIGNR		1	1				ı		1	0.5	SSSE3
	(x)mm,(x)mm,i	2	1		X	X	1		I		SSSE3
PALIGNR	(x)mm,m,i	2	_		X	X	ı		2	0.5	
PBLENDVB	x,x,xmm0		2		1	1	4		2	1	SSE4.1
PBLENDVB	x,m,xmm0	3	2		1	1	1		_	1	SSE4.1
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,(x)mm	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,(x)mm,i	2	2	1	Х	Х			2	1	
PEXTRW	m16,(x)mm,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	Х	Х	1	1		1	SSE4.1
PEXTRQ	r64,x,i	2	2	1	Х	Х			2	1	SSE4.1,
PEXTRQ	m64,x,i	3	2	1	Х	Х	1	1		1	64b
PINSRB	x,r32,i	2	2		Х	х			2	1	SSE4.1
PINSRB	x,m8,i	2	1		Х	х	1			0.5	SSE4.1
PINSRW	(x)mm,r32,i	2	2		Х	х			2	1	
PINSRW	(x)mm,m16,i	2	1		Х	х	1			0.5	
PINSRD	x,r32,i	2	2		Х	х			2	1	SSE4.1
PINSRD	x,m32,i	2	1		Х	х	1			0.5	SSE4.1
PINSRQ	x,r64,i	2	2		Х	х			2	1	SSE4.1,
PINSRQ	x,m64,i	2	1		Х	х	1			0.5	64 b
Arithmetic instructions											
PADD/SUB(U,S)B/W/D/Q	(x)mm, (x)mm	1	1		Χ	Х			1	0.5	
PADD/SUB(U,S)B/W/D/Q	(x)mm,m	1	1		Х	Х	1			0.5	
PHADD/SUB(S)W/D	(x)mm, (x)mm	3	3		Х	Х			2	1,5	SSSE3
PHADD/SUB(S)W/D	(x)mm,m64	4	3		Х	Х	1			1,5	SSSE3
PCMPEQ/GTB/W/D	(x)mm,(x)mm	1	1		Х	Х			1	0.5	
PCMPEQ/GTB/W/D	(x)mm,m	1	1		Х	Х	1			0.5	
PCMPEQQ	X,X	1	1		Х	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	(x)mm,(x)mm	1	1	1					5	1	
PMULL/HW PMULHUW	(x)mm,m	1	1	1			1			1	
PMULHRSW	(x)mm,(x)mm	1	1	1					5	1	SSSE3
PMULHRSW	(x)mm,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	(x)mm,(x)mm	1	1	1			-		5	1	
PMULUDQ	(x)mm,m	l	1	1	I .	1	1	1		1	1

DAMADDIMO					1	ı		_	,	
PMADDWD	(x)mm,(x)mm	1	1	1				5	1	
PMADDWD	(x)mm,m	1	1	1			1	_	1	00050
PMADDUBSW	(x)mm,(x)mm	1	1	1				5	1	SSSE3
PMADDUBSW	(x)mm,m	1	1	1			1	4	1	SSSE3
PAVGB/W	(x)mm,(x)mm	1	1		Х	Х		1	0.5	
PAVGB/W	(x)mm,m	1	1		Х	Х	1	4	0.5	00544
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	(x)mm,(x)mm	1	1		Х	Х		1	0.5	
PMIN/MAXUB	(x)mm,m	1	1		Х	Х	1		0.5	
PMIN/MAXSW	(x)mm,(x)mm	1	1		Х	Х		1	0.5	
PMIN/MAXSW	(x)mm,m	1	1		Х	Х	1		0.5	
PMIN/MAXUW	X,X	1	1		Х	Х	_	1	0.5	SSE4.1
PMIN/MAXUW	x,m	1	1		Х	Х	1		0.5	SSE4.1
PMIN/MAXU/SD	X,X	1	1		Х	Х		1	0.5	SSE4.1
PMIN/MAXU/SD	x,m128	1	1		Х	Х	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	(x)mm,(x)mm	1	1		Х	Х		1	0.5	SSSE3
PABSB/W/D	(x)mm,m	1	1		Х	Х	1		0.5	SSSE3
PSIGNB/W/D	(x)mm,(x)mm	1	1		Х	Х		1	0.5	SSSE3
PSIGNB/W/D	(x)mm,m	1	1		Х	Х	1		0.5	SSSE3
PSADBW	(x)mm,(x)mm	1	1	1				5	1	
PSADBW	(x)mm,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
Logic instructions										
PAND(N) POR PXOR	(x)mm,(x)mm	1	1	X	х	Х		1		
PAND(N) POR PXOR	(x)mm,m	1	1	X	Х	Х	1		0.5	
PXOR	x,same	1	0	^	**		-	0	0.25	
PTEST	x,x	1	2	1	х	х		1	1	SSE4.1
PTEST	x,m128	1	2	1	X	X	1		1	SSE4.1
PSLL/RL/RAW/D/Q	mm,mm/i	1	1	1	**		-	1	1	
PSLL/RL/RAW/D/Q	mm,m64	1	1	1			1		2	
PSLL/RL/RAW/D/Q	x,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,i	1	1	•	X	X	•	1	1	
String instructions										
PCMPESTRI	x,x,i	8	8					4	4	SSE4.2
PCMPESTRI	x,m128,i	8	7				1	·	4	SSE4.2
PCMPESTRM	X,111120,1 X,X,İ	8	8				'	11-12	4	SSE4.2
PCMPESTRM	x,m128,i	8	7				1	. 1 12	4	SSE4.2
PCMPISTRI	X,111120,1 X,X,İ	3	3				'	3	3	SSE4.2
PCMPISTRI	x,m128,i	4	3				1	3	3	SSE4.2
PCMPISTRM	X,111120,1 X,X,İ	3	3				'	11	3	SSE4.2
PCMPISTRM	x,x,i x,m128,i	4	3				1	11	3	SSE4.2
OWI TO FIXIVI	۸,۱۱۱۱۷۵,۱	7	3				'		J	00L4.2
Encryption instructions										
PCLMULQDQ	x,x,i	18	18					14	8	CLMUL

AESDEC, AESDECLAST, AESENC, AESENCLAST									l
	X,X	2	2			8	4	AES	l
AESIMC	X,X	2	2				2	AES	l
AESKEYGENASSIST	x,x,i	11	11			8	8	AES	l
									l
Other									l
EMMS		31	31				18		l

Floating point XMM and YMM instructions

Instruction	Operands	μops	μορε	s un	fus	ed d	loma	in	Latency	Reci-	Com-
		fused do- main	p015	p0	p1	р5	p23	p4		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	y,y,y,i	1	1			1			2	1	AVX
VPERM2F128	y,y,m,i	2	1			1	1+			1	AVX
BLENDPS/PD	x,x,i	1	1	Х		Х			1	0.5	SSE4
BLENDPS/PD	x,m128,i	2	1	Х		Х	1			0.5	SSE4
VBLENDPS/PD	y,y,i	1	1	Х		Х			1	1	AVX
VBLENDPS/PD	y,m256,i	2	1	Х		Х	1+			1	AVX
BLENDVPS/PD	x,x,xmm0	2	2	х		Х			2	1	SSE4
BLENDVPS/PD	x,m,xmm0	3	2	х		Х	1			1	SSE4
VBLENDVPS/PD	y,y,y,y	2	2	Х		Х			2	1	AVX

			- ,	- 5							
VBLENDVPS/PD	y,y,m,y	3	2	Х		Х	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
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	ľ		1	1	AVX
VUNPCKH/LPS/D	y,y,m256	1	1			1	1+			1	AVX
EXTRACTPS	r32,x,i	2	2	1		1	١.		2	1	SSE4.1
EXTRACTPS	m32,x,i	3	2	'		1	1	1	_	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,İ	1	1			1	'	'	1	1	SSE4.1
INSERTPS	x,m32,i	2	1			1	1		'	1	SSE4.1
VINSERTF128		1	1			1	ı		2	1	AVX
VINSERTF128	y,y,x,i	2	1			1	1		2	1	AVX
VMASKMOVPS/D	y,y,m128,i	3	2			1	1			1	AVX
VMASKMOVPS/D	x,x,m128	3	2				1+			1	AVX
VMASKMOVPS/D	y,y,m256	4	2				1	1		1	AVX
	m128,x,x	-					1 .	-		-	
VMASKMOVPS/D	m256,y,y	4	2				1	1+		2	AVX
Conversion											
CVTPD2PS		2	2		1	1			4	1	
CVTPD2PS	x,x x,m128	2	2		1	1	1		4	1	
VCVTPD2PS		2	2		1	1	ı		4	1	AVX
VCVTPD2PS	x,y x,m256				1		1+		4		AVX
CVTSD2SS	·	2 2	2		1	1	ĮΤ		3	1	AVA
CVTSD2SS CVTSD2SS	X,X	2	2		1	1	1		3	· -	
CVTSD2SS CVTPS2PD	x,m64	2	2	1		1	'		3	1 1	
CVTPS2PD	X,X	2		1 -		ı	1		3		
I	x,m64	2	1	1		4	ı		4	1	A\ /\/
VCVTPS2PD VCVTPS2PD	y,x		2	1		1	1		4	1	AVX AVX
1VCV1P5ZPD	1.00	2									Δ \ / X
	y,m128	3	2				'		2		AVA
CVTSS2SD	X,X	2	2	1		1			3	1	AVA
CVTSS2SD CVTSS2SD	x,x x,m32	2 2	2				1			1 1	AVA
CVTSS2SD CVTSS2SD CVTDQ2PS	x,x x,m32 x,x	2 2 1	2 1 1	1	1		1		3	1 1 1	AVA
CVTSS2SD CVTSS2SD CVTDQ2PS CVTDQ2PS	x,x x,m32 x,x x,m128	2 2 1 1	2 1 1 1	1	1				3	1 1 1 1	
CVTSS2SD CVTSS2SD CVTDQ2PS CVTDQ2PS VCVTDQ2PS	x,x x,m32 x,x x,m128 y,y	2 2 1 1 1	2 1 1 1 1	1	1		1			1 1 1 1 1	AVX
CVTSS2SD CVTSS2SD CVTDQ2PS CVTDQ2PS VCVTDQ2PS VCVTDQ2PS	x,x x,m32 x,x x,m128 y,y y,m256	2 2 1 1 1 1	2 1 1 1 1 1	1	1 1 1		1		3	1 1 1 1 1	
CVTSS2SD CVTSS2SD CVTDQ2PS CVTDQ2PS VCVTDQ2PS VCVTDQ2PS CVT(T) PS2DQ	x,x x,m32 x,x x,m128 y,y y,m256 x,x	2 2 1 1 1 1 1	2 1 1 1 1 1 1	1	1 1 1 1		1 1 1+		3	1 1 1 1 1 1	AVX
CVTSS2SD CVTSS2SD CVTDQ2PS CVTDQ2PS VCVTDQ2PS VCVTDQ2PS CVT(T) PS2DQ CVT(T) PS2DQ	x,x x,m32 x,x x,m128 y,y y,m256 x,x x,m128	2 2 1 1 1 1 1	2 1 1 1 1 1 1 1 1 1	1	1 1 1 1 1		1		3 3 3	1 1 1 1 1 1 1	AVX AVX
CVTSS2SD CVTSS2SD CVTDQ2PS CVTDQ2PS VCVTDQ2PS VCVTDQ2PS VCVTDQ2PS CVT(T) PS2DQ CVT(T) PS2DQ VCVT(T) PS2DQ	x,x x,m32 x,x x,m128 y,y y,m256 x,x x,m128 y,y	2 2 1 1 1 1 1 1	2 1 1 1 1 1 1 1	1	1 1 1 1 1 1		1 1+ 1		3	1 1 1 1 1 1 1 1	AVX AVX
CVTSS2SD CVTSS2SD CVTDQ2PS CVTDQ2PS VCVTDQ2PS VCVTDQ2PS VCVTDQ2PS CVT(T) PS2DQ CVT(T) PS2DQ VCVT(T) PS2DQ VCVT(T) PS2DQ	x,x x,m32 x,x x,m128 y,y y,m256 x,x x,m128 y,y y,m256	2 2 1 1 1 1 1 1 1	2 1 1 1 1 1 1 1 1	1	1 1 1 1 1 1 1	1	1 1 1+		3 3 3 3	1 1 1 1 1 1 1 1 1	AVX AVX
CVTSS2SD CVTSS2SD CVTDQ2PS CVTDQ2PS VCVTDQ2PS VCVTDQ2PS VCVTDQ2PS CVT(T) PS2DQ CVT(T) PS2DQ VCVT(T) PS2DQ	x,x x,m32 x,x x,m128 y,y y,m256 x,x x,m128 y,y	2 2 1 1 1 1 1 1	2 1 1 1 1 1 1 1	1	1 1 1 1 1 1		1 1+ 1		3 3 3	1 1 1 1 1 1 1 1	AVX AVX

\ (0) (TD 00DD	I			1				_	1 4	A) () (
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					4	1	
CVT(T) PD2PI	mm,m128	2	2				1		1	
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 CVTSI2SD	x,m32	1	1	'	1		1	7	1,5	
CVTSI2SD CVT(T)SD2SI	r32,x	2	2	1	1		'	4	1,5	
CVT(T)SD2SI	r32,m64	2	2		1		1	4	1	
CV1(1)SD2SI	132,11104	2		'	ı		'		I	
Arithmetic										
ADDSS/D SUBSS/D	- V V	1	1		1			3	1	
ADDSS/D SUBSS/D	x,x x,m32/64	1	1		1		1	J	1	
ADDSS/D SUBSS/D ADDPS/D SUBPS/D		1	1		1		' '	3	1	
	X,X	1			-		4	3		
ADDPS/D SUBPS/D	x,m128		1		1		1	•	1	A) /\/
VADDPS/D VSUBPS/D	y,y,y	1	1		1		١,,	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	x,m128	4	3		1	2	1		2	SSE3
VHADDPS/D										
VHSUBPS/D	y,y,y	3	3		1	2		5	2	AVX
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	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	y,y,y	3	3	2		1		21-29	20-28	AVX
VDIVPS	y,y,m256	4	3	2		1	1+	· _ ·	20-28	AVX
DIVSD DIVPD	x,x	1	1	1		ĺ .		10-22	10-22	
	1 797	•	1 .	1 .	1	1	1 1			

			- ,	- 5						
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	Ū	1	
VRCPPS		3	3	2		1	'	7	2	AVX
	y,y	4	3	-		'	١,, ١	,	2	
VRCPPS	y,m256	4	3				1+		2	AVX
CMPccSS/D CMPccPS/D								_		
	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			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	Ū	1	
MAXPS/D MINPS/D	,	1	1		1		'	3	1	
MAXPS/D MINPS/D	X,X		1		-		1	3	1	
	x,m128	1			1		'	•	1	A) () (
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+	12	4	AVX
DPPD	-	3	3	1	1	1	' '	9	2	SSE4.1
	x,x,i			'	'	'		9		
DPPD	x,m128,i	4	3				1		2	SSE4.1
Math		_	4	,				40.44	10 11	
SQRTSS/PS	X,X	1	1	1				10-14	10-14	
SQRTSS/PS	x,m128	1	1	1			1		10-14	
VSQRTPS	y,y	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	y,y	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	Ū	1	
VRSQRTPS		3	3	'			'	7	2	AVX
	y,y 256	4	3				1+	,	2	
VRSQRTPS	y,m256	4	3						2	AVX
Logio										
Logic AND/ANDN/OR/XORPS/PD	V V	1	1			1		1	4	
	X,X	1	1			1	4	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

(V)XORPS/PD	x/y,x/y,same	1	0					0	0.25	
Other										
VZEROUPPER		4						2	1	AVX
VZEROALL		12							11	AVX, 32 bit
										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 reg-

ister, (x)mm = 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.

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 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	un	fuse	ed d	oma	in	Latency	_	Com- ments
		do- main	p015	p0	p1	р5	p23	p4		through- put	

Move instructions]				I	1					
MOV	r,i	1	1	Х	х	Х			1	0.33	
MOV	r8/16,r8/16	1	1	X	X	X			1	0.33	
MOV	r32/64,r32/64	1	1	X	X	X			0-1	0.25	may be
IVIOV	132/04,132/04	'	'	^	^	^			0-1	0.25	elimin.
MOV	r8/16,m8/16	1	1	х	х	х	1		2	0.5	
MOV	r32/64,m32/64	1					1		2	0.5	
MOV	r,m	1					1		2	1	64 b abs
140)		_									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	Х	Х	Х			1	0.33	
MOVZX	r16,r8	1	1	Х	Х	Х			1	0.33	
MOVZX	r32/64,r8	1	1	X	Х	Х			0-1	0.25	may be elimin.
MOVZX	r32/64,r16	1	1	x	Х	x			1	0.33	ellitiliti.
MOVSX MOVZX	r16,m8	2	1	X	X	X	1		3	0.5	
MOVSX MOVZX	r32/64,m	1		^	^`		1		2	0.5	
MOVSXD	102/04,111	•					Į.			0.5	
CMOVcc	r,r	2	2	Х	Х	х			2	0.67	
CMOVcc	r,m	2	2	х	Х	х	1			~0.8	
XCHG	r,r	3	3	х	х	х			2	1	
XCHG	r,m	7	х				2	3	25		implicit
VIAT		0					4		_		lock
XLAT	_	3	2				1	,	7 3	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	X	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	х	х				1	0.5	1-2 com-
											ponents
LEA	r32/64,m	1	1		1				3	1	3 com- 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	
1	1	_	1	I .	1	1	1 -	1 -	I		1 1

1	I			l			1	1			
Arithmetic instructions											
ADD SUB	r,r/i	1	1	X	Х	Х			1	0.33	
ADD SUB	r,m	1	1	X	Х	X	1			0.5	
ADD SUB	m,r/i	2	1	X	X	X	2	1	6	1	
ADC SBB	r,r/i	2	2	x	X	X	_	'	2	1	
ADC SBB	r,m	2	2	X	X	X	1		2	1	
ADC SBB	m,r/i	4	3	X	X	X	2	1	7-8	2	
CMP	r,r/i	1	1	x	X	X	_	'	1	0.33	
CMP	m,r/i	1	1	x	X	X	1		1	0.5	
INC DEC NEG NOT	r	1	1	x	X	X	'		1	0.33	
INC DEC NEG NOT	m '	3	1	x	X	X	2	1	6	1	
AAA AAS		2	2	x	1	X	_	'	4	•	not 64 bit
DAA DAS		3	3	^	'	^			4		not 64 bit
AAD		3	3						2		not 64 bit
AAM		8	8						20	8	not 64 bit
MUL IMUL	r8	1	1		1				3	1	I TOL OT DIL
MUL IMUL	r16	4	4		'				4	2	
MUL IMUL	r32	3	3						4	2	
MUL IMUL	r64	2	3 2						3	1	
		1			4				3		
IMUL	r,r	2	1 2		1					1	
IMUL	r16,r16,i	1			4				4	1	
IMUL	r32,r32,i		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		_		1			1	
IMUL	r32,m32,i	1	1		1		1			1	
IMUL	r64,m64,i	1	1		1		1		40.00	1	
DIV	r8	11	11						19-22	9	
DIV	r16	11	11						20-24	10	
DIV	r32	10	10						19-27	11	
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						19-26	8-11	
IDIV	r64	59- 134	Х						28-103	26-88	
CBW		1	1	х	Х	х			1	0.33	
CWDE		1	1	х	Х	х			1		
CDQE		1	1	х	Х	х			1		
CWD		2	2	х	Х	х			1		
CDQ		1	1	х		Х			1		
CQO		1	1	х		х			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				SSE4.2
Logic instructions											

			,	J -							
AND OR XOR	r,r/i	1	1	Х	х	Х			1	0.33	
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	
TEST	r,r/i	1	1	x	х	Х			1	0.33	
TEST	m,r/i	1	1	x	х	х	1			0.5	
SHR SHL SAR	r,i	1	1	X		Х	-		1	0.5	
SHR SHL SAR	m,i	3	1	^			2	1		2	
SHR SHL SAR	r,cl	2	2	1		1	_	'	1	1	
SHR SHL SAR	m,cl	5	3	'			2	1	'	4	
ROR ROL	r,1	2	2	x		х	_	'	1	1	short form
ROR ROL	r,i	1	1	x		X			1	0.5	
ROR ROL	m,i	4	3	^		^	2	1	'	2	
ROR ROL	r,cl	2	2			v		'	1	1	
		5	3	X		Х	2	4	'		
ROR ROL	m,cl						2	1	_	4	
RCL RCR	r,1	3	3	X	Х	Х			2	2	
RCL RCR	r,i	8	8	Х	Х	Х		,	5	5	
RCL RCR	m,i	11	8	X	Х	Х	2	1	_	6	
RCL RCR	r,cl	8	8	X	Х	Х	_		5	5	
RCL RCR	m,cl	11	8	X	Х	Х	2	1		6	
SHRD SHLD	r,r,i	1	1	X		Х			1	0.5	
SHRD SHLD	m,r,i	3	3	X	Х	Х	2	1		2	
SHRD SHLD	r,r,cl	4	4	X	1	Х			2	2	
SHRD SHLD	m,r,cl	5	4	X	1	Х	2	1		4	
BT	r,r/i	1	1	Х		Х			1	0.5	
BT	m,r	10	9	x	Х	Х	1			5	
BT	m,i	2	1	x		Х	1			0.5	
BTR BTS BTC	r,r/i	1	1	х		Х			1	0.5	
BTR BTS BTC	m,r	11	8	x	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	x		Х			1	0.5	
SETcc	m	2	1	x		Х	1	1		1	
CLC		1	0							0.25	
STC CMC		1	1	x	Х	х			1	0.33	
CLD STD		3	3	X	Х	Х			-	4	
Control transfer instructi	ons										
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
Conditional jump	Shortmean	'				'				1 2	jump
Fused arithmetic and		1	1			1			0	1-2	fast if no
branch						•					jump
J(E/R)CXZ	short	2	2	X	х	1				1-2	,
LOOP	short	7	7	^		'				4-5	
LOOP(N)E	short	11	11							6	
CALL	near	2	1			1	1	1		2	
CALL	r	2	1			1	1	1		2	
CALL		3	1			1	2	1		2	
RET	m	2				1		'		2	
			1	.,		-	1			2	
RET	į	3	2	X	Х	1	1			2	

BOUND INTO	r,m	15 4	13 4	x	x	x	2			7 6	not 64 bit
			•								
String instructions											
LODS		3	2	Х	Х	Х	1			1	
REP LODS		~5n							~2n		
STOS		3	1	Х	Х	Х	1	1		1	
REP STOS		many							n		worst case
REP STOS		many							1/16B		best
MOVS		5	2	х	х	х	2	1		4	Case
REP MOVS		2n							n		worst
REP MOVS		4/16B							1/16B		case best case
SCAS		3	2	х	х	х	1			1	Case
REP SCAS		~6n							~2n		
CMPS		5	3	х	Х	Х	2			4	
REP CMPS		~8n							~2n		
Synchronization instructi	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	Х	Х	1	1	22		
CMPXCHG	m,r	5	3	Х	Х	х	2	1	7		
LOCK CMPXCHG	m,r	9	6	Х	Х	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	х	Х	2	1		8	
ENTER	a,b	45+7b							84+3b		
LEAVE		3	2	х	х	Х	1			6	
XGETBV		8								9	XGETBV
CPUID		37-82							100-340		
RDTSC		21								27	
RDPMC		35								39	
RDRAND	r	13	12	х	х	х	1			104-117	RDRAND

Floating point x87 instructions

Instruction	Operands	μοps fused					sed domain Latence	_	•	Com- ments	
	do- main	p015	p0	p1	р5	p23	p4		through- put		
Move instructions											

			, Dila	3-							
FLD	r	1	1			1			1	1	
FLD	m32/64	1					1		3	1	
FLD	m80	4	2		1	1	2		5	2	
FBLD		43			'	'	3			21	
	m80		40				3		45		
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 m	3	1		1		1	1	7	1	
FISTTP		3	1		1		1	1	7	2	SSE3
	m				'		ı	'	/		SSES
FLDZ		1	1		_	1					
FLD1		2	2		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	11110	1	1			1		'	1	1	
	_					1			'		
FFREE(P)	r	1	1			1				1	
FNSAVE	m	143								167	
FRSTOR	m	90								162	
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	
FMUL(P)	m	2	1	1			1			1	
FDIV(R)(P)	r	1	1	1					10-24	8-18	
FDIV(R)(P)	m .	2	1	1			1		1021	8-18	
FABS	'''	1	1	'		1			1	1	
						1					
FCHS		1	1			1			1	1	
FCOM(P) FUCOM	r	1	1		1				3	1	
FCOM(P) FUCOM	m 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	
FXAM		2	2		2					2	
FPREM		28	28		_				21-26	12	
FPREM1		41	20						27-50	19	
FRNDINT		17	17						27-50	11	
Math		0.5	0.5						40	40	
FSCALE		25	25	Х	Х	Х			49	49	
FXTRACT		17	17	Х	Х	Х			10	10	
FSQRT		1	1	1					10-23	8-17	
FSIN		21-78		X	Х	X			47-106	47-106	

The state of the s	1 1			1	1	1 1	1		1	
FCOS	23-100		Х	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	Х		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	Х	Х	Х			22		
FNINIT	26	26	Х	Х	Х			80		

Integer MMX and XMM instructions

Instruction	Operands	μοps fused	µops	un	fus	ed d	loma	in	Latency	Reci- procal through- put	Com- ments
		do- main	p015	p0	p1	p5	p23	p4			
Move instructions											
MOVD	r32/64,(x)mm	1	1	1					1	1	
MOVD	m32/64,(x)mm	1					1	1	3	1	
MOVD	(x)mm,r32/64	1	1			1			1	1	
MOVD	(x)mm,m32/64	1					1		3	0.5	
MOVQ	(x)mm,(x)mm	1	1	Х	Х	Х			1	0.33	
MOVQ	(x)mm,m64	1					1		3	0.5	
MOVQ	m64, (x)mm	1					1	1	3	1	
MOVDQA MOVDQU	X,X	1	1	Х	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	Х	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											
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	(x)mm,(x)mm	1	1		Х	Х			1	0.5	
PUNPCKH/LBW/WD/DQ	(x)mm,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	x			1	0.5	SSE4.1

			,	3-							
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	(x)mm,(x)mm	1	1		х	Х			1	0.5	SSSE3
PSHUFB	(x)mm,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	xmm,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		X	X			1	0.5	
PSHUFL/HW	x, m128,i	2	1		X	X	1		•	0.5	
PALIGNR	(x)mm,(x)mm,i	1	1		X	X			1	0.5	SSSE3
PALIGNR	(x)mm,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
PBLENDW	x,x,i	1	1		X	X	'		1	0.5	SSE4.1
PBLENDW	x,m,i	2	1		X	X	1		'	0.5	SSE4.1
MASKMOVQ	mm,mm	4	1	1	^	^	2	1		1	JOL4.1
MASKMOVDQU	X,X	10	4	X	1	х	4	2		6	
PMOVMSKB	r32,(x)mm	10	1	1	'	^	_	_	2	1	
PEXTRB	r32,x,i	2	2	1	х	х			2	1	SSE4.1
PEXTRB	m8,x,i	2	1	'	X	X	1	1		1	SSE4.1
PEXTRW	r32,(x)mm,i	2	1	1	X	X	'	ļ '	2	1	00L4.1
PEXTRW	m16,(x)mm,i	2	1	'	X	X	1	1		1	SSE4.1
PEXTRD	r32,x,i	2	2	1	X	X	'	ļ '	2	1	SSE4.1
PEXTRD	m32,x,i	2	1	'	X	X	1	1		1	SSE4.1
PEXTRQ	r64,x,i	2	2	1	X	X	'	'	2	1	SSE4.1
PEXTRQ	m64,x,i	2	1	'	X	X	1	1		1	0024.1
PINSRB	x,r32,i	2	2		X	X	'	ļ '	2	1	SSE4.1
PINSRB	x,m8,i	2	1		X	X	1			0.5	SSE4.1
PINSRW	(x)mm,r32,i	2	2		X	X	'		2	1	0024.1
PINSRW	(x)mm,m16,i	2	1		X	X	1		_	0.5	
PINSRD	x,r32,i	2	1		X	X	'		2	1	SSE4.1
PINSRD	x,m32,i	2	1		X	X	1			0.5	SSE4.1
PINSRQ	x,r64,i	2	1		X	X	'		2	1	SSE4.1
PINSRQ	x,m64,i	2	1		X	X	1			0.5	SSE4.1
Arithmetic instructions											
PADD/SUB(U,S)B/W/D/Q	(x)mm, (x)mm	1	1		Х	х			1	0.5	
PADD/SUB(U,S)B/W/D/Q	(x)mm,m	1	1		Х	Х	1			0.5	
PHADD/SUB(S)W/D	(x)mm, (x)mm	3	3		Х	Х			3	1,5	SSSE3
PHADD/SUB(S)W/D	(x)mm,m64	4	3		Х	Х	1			1,5	SSSE3
PCMPEQ/GTB/W/D	(x)mm,(x)mm	1	1		X	X			1	0.5	
PCMPEQ/GTB/W/D	(x)mm,m	1	1		Х	Х	1			0.5	
PCMPEQQ	X,X	1	1		X	Х			1	0.5	SSE4.1
T. Control of the Con		1	1	1	1	1	1	1	l .	1	

DOMDE 0.0	1 400			ī	ı	I	l a l	1	1		
PCMPEQQ	x,m128	1	1	١.	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
PMULL/HW PMULHUW	(x)mm,(x)mm	1	1	1					5	1	
PMULL/HW PMULHUW	(x)mm,m	1	1	1			1			1	
PMULHRSW	(x)mm,(x)mm	1	1	1					5	1	SSSE3
PMULHRSW	(x)mm,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	(x)mm,(x)mm	1	1	1					5	1	
PMULUDQ	(x)mm,m	1	1	1			1			1	
PMADDWD	(x)mm,(x)mm	1	1	1					5	1	
PMADDWD	(x)mm,m	1	1	1			1			1	
PMADDUBSW	(x)mm,(x)mm	1	1	1					5	1	SSSE3
PMADDUBSW	(x)mm,m	1	1	1			1			1	SSSE3
PAVGB/W	(x)mm,(x)mm	1	1	Ι.	х	х	·		1	0.5	00020
PAVGB/W	(x)mm,m	1	1		x	X	1		'	0.5	
PMIN/MAXSB	` ,	1	1		x		'		1	0.5	SSE4.1
PMIN/MAXSB	x,x x,m128	1	1			X	1		1	0.5	SSE4.1
	·	•			X	X			4		33E4.1
PMIN/MAXUB	(x)mm,(x)mm	1	1		X	X			1	0.5	
PMIN/MAXUB	(x)mm,m	1	1		Х	Х	1			0.5	
PMIN/MAXSW	(x)mm,(x)mm	1	1		Х	Х			1	0.5	
PMIN/MAXSW	(x)mm,m	1	1		Х	Х	1			0.5	
PMIN/MAXUW	X,X	1	1		Х	Х			1	0.5	SSE4.1
PMIN/MAXUW	x,m	1	1		Х	Х	1			0.5	SSE4.1
PMIN/MAXU/SD	X,X	1	1		Х	Х			1	0.5	SSE4.1
PMIN/MAXU/SD	x,m128	1	1		Х	Х	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	(x)mm,(x)mm	1	1		Х	х			1	0.5	SSSE3
PABSB/W/D	(x)mm,m	1	1		Х	х	1			0.5	SSSE3
PSIGNB/W/D	(x)mm,(x)mm	1	1		Х	х			1	0.5	SSSE3
PSIGNB/W/D	(x)mm,m	1	1		Х	х	1			0.5	SSSE3
PSADBW	(x)mm,(x)mm	1	1	1					5	1	
PSADBW	(x)mm,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
	7.5,.11,11	•		.						·	
Logic instructions											
PAND(N) POR PXOR	(x)mm,(x)mm	1	1	X	х	х			1	0.33	
PAND(N) POR PXOR	(x)mm,m	1	1	X	X	X	1		•	0.5	
PTEST	X,X	2	2	1		X	'		1	1	SSE4.1
PTEST	x,m128	3	2	1	X		1		1	1	SSE4.1
	· ·				Х	Х	1		1		33E4.1
PSLL/RL/RAW/D/Q	mm,mm/i	1	1	1					1	1	
PSLL/RL/RAW/D/Q	mm,m64	1	1	1			1		4	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	Х	Х	1			1	
PSLL/RLDQ	x,i	1	1		Х	Х			1	0.5	
String instructions											

l= 0==0==.				۱ ـ		١	1	1			
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
PCMPISTRM	x,m128,i	4	3	3			1			3	SSE4.2
Encryption instructions											
PCLMULQDQ	x,x,i	18	18	x	х	х			14	8	CLMUL
	·	_					4		14	_	
PCLMULQDQ	x,m,i	18	17	Х	Х	Х	1			8	CLMUL
AESDEC, AESDECLAST,											
AESENC, AESENCLAST		_	_						_	_	
	X,X	2	2	X	Х	1			4	1	AES
AESDEC, AESDECLAST,											
AESENC, AESENCLAST											
	x,m	3	2	Х	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	μοps fused	μops	un	fus	ed d	loma	in	Latency	Reci- procal through- put	Com- ments
		do- main	p015	p0	p1	p5	p23	p4			
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/D	m256,y	1					1	1+	4	2	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	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 VSHUFPS/D VSHUFPS/D VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERMILPS/PD VPERM2F128 VPERM2F128 BLENDPS/PD BLENDPS/PD VBLENDPS/PD VBLENDPS/PD VBLENDVPS/PD	x,m128,i y,y,y,i y, y,m256,i x,x,x/i y,y,y/i x,x,m y,y,m x,m,i y,y,i y,y,m,i x,x,i x,m128,i y,y,i y,m256,i x,x,xmm0 x,m,xmm0 y,y,y,y y,y,m,y	2 1 2 1 1 2 2 2 2 1 2 1 2 1 2 2 3 1 1 1 2 3 1 1 1 1	1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 1	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 X X X X X X X X X X	1 1+ 1 1+ 1+ 1+ 1 1+ 1		1 1 1 2 1 2 2 1 3 1	1 1 1 1 1 1 1 1 0.5 0.5 0.5 1 1 1 1 1 1 1 1	AVX AVX AVX AVX AVX AVX AVX AVX SSE4.1 SSE4.1 AVX AVX SSE4.1 SSE4.1 AVX AVX SSE4.1 SSE4.1 AVX AVX SSE4.1 AVX AVX SSE4.1 AVX AVX
VUNPCKH/LPS/D VUNPCKH/LPS/D EXTRACTPS	y,y,y y,y,m256 r32,x,i	1 1 2	1 1 2	x		1 1 x	1+		1 2	1 1 1	AVX AVX SSE4.1
EXTRACTES VEXTRACTES VEXTRACTF128 VEXTRACTF128 INSERTPS INSERTPS VINSERTF128 VINSERTF128 VMASKMOVPS/D VMASKMOVPS/D VMASKMOVPS/D VMASKMOVPS/D VMASKMOVPS/D VMASKMOVPS/D	m32,x,i x,y,i x,y,i m128,y,i x,x,i x,m32,i y,y,x,i y,y,m128,i x,x,m128 y,y,m256 m128,x,x m256,y,y	3 1 2 1 2 1 2 3 3 4 4	2 1 0 1 1 1 2 2 2	x x x	xx	1 1 1 1 1 x x	1 1 1 1 1 1+ 1	1 1 1+	2 4 1 2 4 4 5	1 1 1 1 1 1 1 1 1 1 2	SSE4.1 AVX AVX SSE4.1 SSE4.1 AVX AVX AVX AVX AVX AVX
CVTPD2PS CVTPD2PS VCVTPD2PS VCVTPD2PS	x,x x,m128 x,y x,m256	2 2 2 2	2 2 2 2		1 1 1 1	1 1 1 1	1 1+		4	1 1 1 1	AVX AVX

			,	3 -							
CVTSD2SS	x,x	2	2		1	1			4	1	
CVTSD2SS	x,m64	2	2		1	1	1			1	
CVTPS2PD	x,x	2	2	1		1			1	1	
CVTPS2PD	x,m64	2	1	1			1			1	
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			2	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	у,у	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+		Ū	1	AVX
CVTDQ2PD	x,x	2	2		1	1	' '		4	1	/\\
CVTDQ2PD	x,m64	2	2		1	1	1		7	1	
VCVTDQ2PD	y,x	2	2		1	1	'		5	1	AVX
VCVTDQ2PD	y,m128	2	2		1	1	1		0	1	AVX
CVT(T)PD2DQ	X,X	2	2		1	1	'		4	1	
CVT(T)PD2DQ	x,m128	2	2		1	1	1		7	1	
VCVT(T)PD2DQ		2	2		1	1	'		5	1	AVX
VCVT(T)PD2DQ VCVT(T)PD2DQ	x,y x,m256	2	2		1	1	1+		5	1	AVX
CVTPI2PS	x,mm	1	1		1		١.,		4	!	
CVTPI2PS	x,m64	1	1		1		1		7	3	
CVT(T)PS2PI	mm,x	2	2		1	1	' '		4	1	
CVT(T)PS2PI	mm,m128	2	1		1		1		7	1	
CVTPI2PD	x,mm	2	2		1	1	'		4	1	
CVTPI2PD	x,m64	2	2		1	1	1		7	1	
CVT(T) PD2PI	mm,x	2	2		1	1	'		4	1	
CVT(T) PD2PI	mm,m128	2	2		1	1	1		7	1	
CVTSI2SS	x,r32	2	2		1	1	'		4	3	
CVTSI2SS	x,m32	1	1		1	'	1		7	3	
CVT(T)SS2SI	r32,x	2	2	1	1		'		4	1	
CVT(T)SS2SI	r32,m32	2	2	1	1		1		7	1	
CVTSI2SD	x,r32	2	2	'	1	1	'		4	3	
CVTSI2SD	x,m32	2	1		1	'	1		7	3	
CVT(T)SD2SI	r32,x	2	2	1	1		'		4	1	
CVT(T)SD2SI	r32,m64	2	2	1	1		1		7	1	
VCVTPS2PH	x,v,i	3	3	1	1	1	'		10	1	F16C
VCVTPS2PH	m,v,i	3	2	1	1	'	1	1	10	1	F16C
VCVTPH2PS	V,X	2	2	1	'	1	'	'	6	1	F16C
VCVTPH2PS	v,m	2	1	'	1	'	1		O	1	F16C
V	V,111		'		'		'				1 100
Arithmetic											
ADDSS/D SUBSS/D	X,X	1	1		1				3	1	
ADDSS/D SUBSS/D	x,m32/64	1	1		1		1		9	1	
ADDPS/D SUBPS/D	X,11132704 X,X	1	1		1		'		3	1	
ADDPS/D SUBPS/D	x,m128	1	1		1		1		J	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+		Ū	1	AVX
ADDSUBPS/D	x,x	1	1		1				3	1	SSE3
r ==	1 797		1 .	1	1 1	1	1	1 1	•		, 5526

			,							
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	x,m128	4	3		1	2	1		2	SSE3
VHADDPS/D	,									
VHSUBPS/D	y,y,y	3	3		1	2		5	2	AVX
VHADDPS/D	3.3.3									
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	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	•	1	1	1			'	5	1 1	AVX
VMULPD	y,y,y y,y,m256	1	1	1			1+	3	1	AVX
DIVSS DIVPS	y,y,111230 X,X	1	1	1			' '	10-13	7	
DIVSS DIVPS		1	1	1			1	10-13	7	
VDIVPS	x,m	3	3	2		1	'	19-21	14	AVX
VDIVPS	y,y,y	4	3	2		1	1+	19-21	14	AVX
DIVSD DIVPD	y,y,m256	1	1	1		'		10.20	8-14	AVA
	X,X	-	-				4	10-20		
DIVSD DIVPD	x,m	1	1	1		1	1	20.25	8-14	A) ()/
VDIVPD	y,y,y	3	3	2		1	١,, ١	20-35	16-28	AVX
VDIVPD	y,y,m256	4	3	2		1	1+		16-28	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	2		1	1+		2	AVX
CMPccSS/D CMPccPS/D								_		
	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	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	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
1		1	0	1	1	1	1 1	1	1	1

Math										
SQRTSS/PS	x,x	1	1	1				11	7	
SQRTSS/PS	x,m128	1	1	1		1			7	
VSQRTPS	y,y	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	
SQRTSD/PD	x,m128	1	1	1		1			8-14	
VSQRTPD	y,y	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,

(x)mm = 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.

μορs fused domain:

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

uops count as one.

μops unfused domain:

The total number of μ ops for all execution port. Fused μ ops count as two. Fused macroops count as one. The instruction has μ op 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 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 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.

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	µops fused domain	µops 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	r,m	1	1	p23		0.5	all other
MOVSXD	,			F -			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		21	-	implicit lock
XLAT	,,,,,	3	3		7	2	
PUSH	r	1	2	p237 p4	3	1	
PUSH	i i	1	2	p237 p4	Ū	1	
PUSH	m .	2	3	p4 2p237		1	
PUSH	stack pointer	2	3	p0156 p237 p4		1	
PUSHF(D/Q)	Ctack pointer	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	HOT OF BIT
POP	stack pointer	3	3	p23 2p0156	_	4	
POP	m	2	3	2p237 p4		1	
POPF(D/Q)	""	9	9	Ζρ237 ρ4		18	
POPA(D)		18	18			9	not 64 bit
LAHF SAHF		10	1	p06	1	1	HOL O4 DIL
SALC		3	3	3p0156	1	1	not 64 bit
LEA	r16,m	2	2	p1 p0156	4	1	16 or 32 bit
	1 10,111	2		ρι ρυ 130	7	'	address size
LEA	r32/64,m	1	1	p15	1	0.5	1 or 2 compo-
	132/04,111	,		p 15	!	0.5	nents in
							address
LEA	r32/64,m	1	1	p1	3	1	3 components
	102/01,111	•	•	ρ.	Ū	•	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/	m	1	1	p23		0.5	
0/1/2							
LFENCE		2		none counted		4	
MFENCE		3	2	p23 p4		33	
SFENCE		2	2	p23 p4		5	
Arithmetic in-							
structions							
ADD SUB	r,r/i	1	1	p0156	1	0.25	
ADD SUB	r,m	1	2	p0156 p23		0.5	

				O. O.			
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	
	,					_	
CMP	r,r/i	1	1	p0156	1	0.25	
CMP	m,r/i	1	2	p0156 p23	1	0.5	
INC DEC NEG	r	1	1	p0156	1	0.25	
NOT							
INC DEC NOT	m	3	4	p0156 2p237 p4	6	1	
NEG	m	2	4	p0156 2p237 p4	6	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	
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	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		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	AVX2
MULX	r32,r32,m32	3	4	p1 2p056 p23		1	AVX2
MULX	r64,r64,r64	2	2	p1 p6	4	1	AVX2
MULX	r64,r64,m64	2	3	p1 p6 p23		1	AVX2
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	9-11	
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		00545
POPCNT	r,r	1	1	p1	3	1	SSE4.2
POPCNT	r,m	1	2	p1 p23		1	SSE4.2

				iowen			
CRC32	r,r	1	1	p1	3	1	SSE4.2
CRC32	r,m	1	2	p1 p23		1	SSE4.2
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			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
BT	r,r/i	1	1	p06	1	0.5	
BT	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	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 BLSI BLSMSK BLSI BLSMSK BLSI BLSMSK BLSI BLSMSK BLSI BLSMSK BLSR BLSTR F, r, r							1	
BLSI BLSMSK BLSR BLSR BLSR BLSR BLSR BLSR BLSR BLSR	ANDN	r,r,r	1 1		p15	1	0.5	BMI1
BLSR BLSI BLSMSK BLSI R BEXTR BEXTR F.R., r.R., r. 2 2 2 2p0156 2 2 0.5 BMI1 BEXTR F.M., r. 3 3 2p0156 p23 1 BMI2 BZHI F.M., r. 1 1 2 p15 p23 0.5 BMI2 BMI2 BZHI F.M., r. 1 1 2 p15 p23 0.5 BMI2 BMI2 BZHI F.M., r. 1 1 2 p15 p23 0.5 BMI2 BMI2 BZHI F.M., r. 1 1 2 p15 p23 0.5 BMI2 BMI2 BZHI F.M., r. 1 1 2 p15 p23 0.5 BMI2 BMI2 PDEP F.R., r. 1 1 1 p1 3 1 BMI2 PEXT F.R., r. 1 1 1 p1 3 1 BMI2 PEXT F.R., r. 1 1 1 p1 3 1 BMI2 PEXT F.R., r. 1 1 1 p1 3 1 BMI2 PEXT F.R., r. 1 1 1 p6 2 2 D15 p23 BMI2 BMI2 BMI2 PEXT F. R. 1 1 PEXT F. R. 1 1 PEXT F. R. 1 1 PEXT F. 1 PEXT F. 1 PEX	ANDN	r,r,m	1 1	2	p15 p23	1	0.5	BMI1
BLSR BEXTR		r,r	1	1	p15	1	0.5	BMI1
BEXTR r,m,r 3 3 2p0156 p23 1 BMI1 BMI1 BMI1 BMI1 BMI1 BMI2		r,m	1	2	p15 p23		0.5	BMI1
BEXTR r,m,r 3 3 2p0156 p23 1 BMI1 BMI1 BMI1 BMI1 BMI1 BMI2	BEXTR	r.r.r	2	2	2p0156	2	0.5	BMI1
BZHI								
BZHI	BZHI		1 1	1		1	0.5	BMI2
PDEP	BZHI		1 1	2			0.5	BMI2
PDEP	PDEP		1 1	1		3	1	BMI2
PEXT PEXT r,r,r 1 1 p1 p1 p23 3 1 BMI2 BMI2 Control transfer instructions JMP short/near 1 1 p6 1-2 JMP JMP m 1 2 p23 p6 2 2 JCOnditional jump Short/near 1 1 p6 1-2 predicted taken 1 1 p6 1-2 predicted taken 1 1 p6 1-2 predicted taken 1 predicted taken 1 predicted not taken 1 1 p6 0.5-1 predicted not taken 1 1 p06 0.5-1 predicted not taken 1 1 p06 0.5-1 predicted not taken 1 1 p06 0.5-1 predicted not taken 1 1 p06 0.5-1 predicted not taken 1 1 p06 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 <t< td=""><td></td><td></td><td>1 1</td><td>2</td><td>1</td><td></td><td>1</td><td>BMI2</td></t<>			1 1	2	1		1	BMI2
PEXT	PEXT		1 1	1		3	1	BMI2
JMP	PEXT		1	2			1	BMI2
JMP JMP JMP r m 1 m <th< td=""><td>Control transfer i</td><td>instructions</td><td></td><td></td><td></td><td></td><td></td><td></td></th<>	Control transfer i	instructions						
JMP Conditional jump m 1 2 p23 p6 2 1-2 predicted taken Conditional jump short/near 1 1 p06 0.5-1 predicted taken Fused arithmetic and branch 1 1 1 p6 1-2 predicted not taken Fused arithmetic and branch 1 1 1 p06 0.5-1 predicted not taken Fused arithmetic and branch 1 1 p06 0.5-1 predicted not taken Fused arithmetic and branch 1 1 p06 0.5-1 predicted not taken Fused arithmetic and branch 1 1 p06 0.5-1 predicted not taken Fused arithmetic and branch 1 1 p06 0.5-2 predicted not taken LOP(R)E short 2 2 p0156 p6 0.5-2 0.5-2 LOOP(N)E short 1 1 p237 p4 p6 2 2 CALL m 3 4 p2237 p4 p6 3 3	JMP	short/near	1 1	1	p6		1-2	
Conditional jump short/near 1 1 p6 1-2 predicted taken taken Conditional jump short/near 1 1 p06 0.5-1 predicted not taken Fused arithmetic and branch 1 1 p6 1-2 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 0.5-1 predicted not taken J(E/R)CXZ short 2 2 p0156 p6 0.5-2 predicted not taken J(E/R)CXZ short 2 2 p0156 p6 0.5-2 predicted not taken J(E/R)CXZ short 2 2 p0156 p6 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-1 predicted not taken J(E/R)CXZ short 2 2 p0156 p6 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 0.5-2 </td <td>JMP</td> <td>r</td> <td>1 1</td> <td>1</td> <td>p6</td> <td></td> <td>2</td> <td></td>	JMP	r	1 1	1	p6		2	
Taken Conditional jump Short/near 1	JMP	m	1 1	2	p23 p6		2	
Fused arithmetic and branch Fused arithmetic and branch Fused arithmetic and branch Fused arithmetic and branch J(E/R)CXZ LOOP Short LOOP(N)E LOOP(N)E Short Shor	Conditional jump	short/near	1	1			1-2	
and branch 1 1 p06 0.5-1 predicted not taken Fused arithmetic and branch J(E/R)CXZ short 2 2 p0156 p6 0.5-2	Conditional jump	short/near	1	1	p06		0.5-1	·
and branch J(E/R)CXZ short 2 2 p0156 p6 0.5-2 taken J(E/R)CXZ short 7 7 5 5 6 LOOP(N)E short 11 11 11 6 6 CALL near 2 3 p237 p4 p6 2 2 CALL r 2 3 4 2p237 p4 p6 3 3 RET 1 2 p237 p6 1 1 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 1 3 3 2 p0156 p23 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2			1	1	p6		1-2	
LOOP LOOP(N)E short short 7 7 5 6 7 7 7 7 7 7 7 6 6 6 6 6 6 2			1	1	p06		0.5-1	
LOOP(N)E short 11 11 11 6 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2	J(E/R)CXZ	short			p0156 p6		0.5-2	
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 not 64 bit LODSB/W 2 2 p0156 p23 1 22n STOS 5n+12 72n 72n 72n 72n STOS 3 3 p23 p0156 p4 1 70.5n worst case REP STOS 2.6/32B 5 5 2p23 p4 2p0156 4 71.5 n worst case		short	7	7			5	
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 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	` '	short						
CALL m 3 4 2p237 p4 p6 3 RET i 3 4 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 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 2.6/32B 1/32B best case aligned by 32 MOVS 5 5 2p23 p4 2p0156 4 REP MOVS 5 5 2p23 p4 2p0156 4 ~1.5 n worst case		near						
RET i 1 2 p237 p6 1 2 B23 2p6 p015 2 BDUND 15 16 15 15 16 15 15 15 16 15 15 15 15 16 </td <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>								
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 ~2n ~2n STOS 3 3 p23 p0156 p4 1 REP STOS <2n		m						
BOUND INTO r,m 15 15 4 4 8 not 64 bit 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 2.6/32B 1/32B best case aligned by 32 MOVS 5 5 2p23 p4 2p0156 4 REP MOVS ~2n ~1.5 n worst case	l		_					
String instructions		-			p23 2p6 p015			
String instructions 3 3 2p0156 p23 1 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		r,m						
tions 3 3 2p0156 p23 1 LODSD/Q 2 2 p0156 p23 1 REP LODS 5n+12 ~2n 7 STOS 3 3 p23 p0156 p4 1 REP STOS <2n	INTO		4	4			5	not 64 bit
LODSD/Q 2 2 p0156 p23 1 REP LODS 5n+12 -2n -2n STOS 3 3 p23 p0156 p4 1 REP STOS -2n -0.5n worst case REP STOS 2.6/32B 1/32B best case MOVS 5 5 2p23 p4 2p0156 4 REP MOVS -2n -2n worst case								
REP LODS 5n+12 72n STOS 3 3 p23 p0156 p4 1 REP STOS <2n	LODSB/W		3	3	2p0156 p23		1	
STOS 3 3 p23 p0156 p4 1 ~0.5n worst case REP STOS 2.6/32B 2.6/32B best case aligned by 32 MOVS 5 5 2p23 p4 2p0156 4 ~1.5 n worst case	LODSD/Q		2	2	p0156 p23		1	
REP STOS <2n	REP LODS		5n+12				~2n	
REP STOS 2.6/32B 1/32B best case aligned by 32 MOVS 5 5 2p23 p4 2p0156 4 REP MOVS ~2n ~1.5 n worst case	STOS		3	3	p23 p0156 p4		1	
MOVS 5 5 2p23 p4 2p0156 4 ~1.5 n worst case	REP STOS		<2n				~0.5n	worst case
REP MOVS ~2n ~1.5 n worst case	REP STOS		2.6/32B				1/32B	
REP MOVS ~2n ~1.5 n worst case	MOVS		5	5	2p23 p4 2p0156		4	
DEP MOVS 4/32R heet case					' '		~1.5 n	worst case
aligned by 32	REP MOVS		4/32B				1/32B	best case aligned by 32
SCAS 3 p23 2p0156 1	SCAS		3	3	p23 2p0156		1	
REP SCAS ≥6n ≥2n	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	
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 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		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	µops fused domain	µops 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	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 e	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	n227 n4 n6	l	1 1	
FINCSTP FDECS		1	1	p237 p4 p6 p01	0	0.5	
					0	0.5	
FFREE(P)	r	1		p01			
FNSAVE	m	147	147			150	
FRSTOR	m	90	90			164	
Arithmetic in- structions							
FADD(P)							
FSUB(R)(P)	r	1	1	p1	3	1	
FADD(P)							
FSUB(R)(P)	m	1	2	p1 p23		1	
FMUL(P)	r	1	1	p0	5	1	
FMUL(P)	m	1	2	p0 p23		1	
FDIV(R)(P)	r	1	1	p0	10-24	8-18	
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) FUCON		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					40.40=		
FSCALE		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		
Other							
FNOP		1	1	p01		0.5	
WAIT		2	2	p01		1	
FNCLEX		5	5	p0156		22	
FNINIT		26	26	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		83	

Integer vector instructions

		µорѕ	µops			Recipro-	
		fused	unfused	_		through	
Instruction	Operands	domain	domain	µops each port	Latency	put	Comments
Move instruc-							
tions	m20/64 ()/\mama			-0	_	4	
MOVD	r32/64,(x)mm	1	1	p0	1	1	
MOVD	m32/64,(x)mm		2	p237 p4	3	1	
MOVD	(x)mm,r32/64	1	1	p5	1	1	
MOVD	(x)mm,m32/64	1	1	p23	3	0.5	
MOVQ	r64,(x)mm	1	1	p0	1	1	
MOVQ	(x)mm,r64	1	1	p5	1	1	
MOVQ	(x)mm,(x)mm	1		p015	1	0.33	
MOVQ	(x)mm,m64	1	1	p23	3	0.5	
MOVQ	m64, (x)mm	1	2	p237 p4	3	1	
MOVDQA/U	X,X	1	1	p015	0-1	0.33	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	0.33	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	0020
MOVQ2DQ	x,mm	1	1	p015	1	0.33	
MOVNTQ	m64,mm	1	2	p237 p4	~400	1	
MOVNTDQ		1	2		~400		
	m128,x		2	p237 p4		1	A) /// O
VMOVNTDQ	m256,y	1		p237 p4	~400	1	AVX2
MOVNTDQA	x, m128	1	1	p23	3	0.5	SSE4.1
VMOVNTDQA PACKSSWB/DW	y,m256	1	1	p23	3	0.5	AVX2
PACKUSWB	mm,mm	3	3	р5	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	, .,.,		_	1 2 1 2			
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

T.	I		I	l I		I	l I
VPMOVSX/ZX BW		2	2	nE n22		4	AVX2
BD BQ DW DQ	y,m	1	2	p5 p23	1	1 1	SSSE3
PSHUFB	V,V / V,V,V	-		p5	ı	-	
PSHUFB	v,m / v,v,m	2	2	p23 p5	4	1	SSSE3
PSHUFW	mm,mm,i	1	1	p5	1	1	
PSHUFW	mm,m64,i	2	2	p23 p5		1	
PSHUFD	V,V,İ	1	1	p5	1	1	
PSHUFD	v,m,i	2	2	p23_p5		1	
PSHUFL/HW	V,V,İ	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	
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	(x)mm,r32,i	2	2	p5	2	2	
PINSRW	(x)mm,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							
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	00/04	_	_	- 00	4	0.5	A) ()(O
D/Q	x,m32/64	1	1	p23	4	0.5	AVX2
VPBROADCAST B/W/D/Q	\ \v	1	1	p5	3	1	AVX2
VPBROADCAST	y,x	'	'	μο	J	I	7774
B/W	y,m8/16	3	3	p01 p23 p5	7	1	AVX2

	I	I	ı	I	1	T.	1
VPBROADCAST	00/04				_		1) 0/0
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
VPGATHERQQ	y,[r+s*y],y	22	22			9	AVX2
Arithmetic in-							
structions							
PADD/SUB(S,US)							
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							
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							
PCMPGTB/W/D	v,m / v,v,m	1	2	p15 p23		0.5	
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	р0	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	р0	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	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	·,, v, v,!!!		_	p 10 p20		0.5	
SB/SW/SD							
UB/UW/UD	x,x / y,y,y	1	1	p15	1	0.5	SSE4.1
PMIN/PMAX	, ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,				'	0.5	
SB/SW/SD							
UB/UW/UD	x,m / y,y,m	1	2	p15 p23		0.5	SSE4.1
152.555	, ,,,,,, J,J,,,,,		_	F.0 P=0	I	1 0.0	

PHMINPOSUW	x,x	1	1	p0	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,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	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- tions								
PAND PANDN	1							
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	111111,111111	ı	'	ρο	'	'		
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		
	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	2		ρυ ρυ		'		
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	V, V, V	3		Ζρο ρο		_	AVAZ	
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	11	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	10	5	SSE4.2	
PCMPESTRM	x,m128,i	9	9	6p05 2p16 p23		5	SSE4.2	
PCMPISTRI	x,x,i	3	3	3p0	11	3	SSE4.2	
PCMPISTRI	x,m128,i	4	4	3p0 p23		3	SSE4.2	
PCMPISTRM	x,x,i	3	3	3p0	10	3	SSE4.2	
PCMPISTRM	x,m128,i	4	4	3p0 p23		3	SSE4.2	

Encryption instru	ıctions						
PCLMULQDQ	x,x,i	3	3	2p0 p5	7	2	CLMUL
PCLMULQDQ	x,m,i	4	4	2p0 p5 p23		2	CLMUL
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		3	3	2p5 p23	17	2	AES
AESKEYGENAS SIST	x,m x,x,i	10	10	2p0 8p5	10	9	AES
AESKEYGENAS SIST	x,m,i	10	10	2p0 p23 7p5		8	AES
Other							
EMMS		31	31			13	

Floating point XMM and YMM instructions

Instruction	Operands	µops fused domain	µops unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instruc- 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 MOVUPS/D VMOVAPS/D VMOVUPS/D	x,m128 y,m256	1	1	p23	3	0.5	AVX
MOVAPS/D	y,111230	'	'	μ23	3	0.5	AVA
MOVAPS/D MOVUPS/D VMOVAPS/D	m128,x	1	2	p237 p4	3	1	
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	p23	5	0.5	AVX
VBROADCASTSD	y,1110-1 y,x	1	1	p5	3	1	AVX2
VBROADCASTF128	y,n y,m128		1	p23	3	0.5	AVX
MOVSH/LDUP	y,111120 V,V		1	p5	1	1	SSE3
MOVSH/LDUP	v,v v,m			p23	3	0.5	SSE3
UNPCKH/LPS/D	x,x / v,v,v		1	p5	1	1	SSE3
UNPCKH/LPS/D	x,m / v,v,m		2	p5 p23	ı	1	SSE3
EXTRACTPS	r32,x,i	2	2	p0 p5			SSE4.1
EXTRACTPS	m32,x,i	3	3	p0 p5 p23	4	1	SSE4.1
VEXTRACTF128	x,y,i	1	1	p5 p5 p25	3	1	AVX
VEXTRACTF 128	m128,y,i	2	2	p23 p4	4	1	AVX
INSERTPS	•	1	1	p23 p4	1	1	SSE4.1
INSERTPS	x,x,i x,m32,i	2	2	p23 p5	4	1	SSE4.1
VINSERTF128		1	1		3	1	AVX
VINSERTF128	y,y,x,i	2	2	p5 p015 p23	4	2	AVX
	y,y,m128,i	3	3		4	2	AVX
VMASKMOVPS/D VMASKMOVPS/D	v,v,m	4	4	2p5 p23	13	1	AVX
	m128,x,x		4	p0 p1 p4 p23	14	2	AVX
VMASKMOVPS/D	m256,y,y	4	1	p0 p1 p4 p23	14		
VPGATHERDPS	x,[r+s*x],x	20	20			9 12	AVX2
VPGATHEROPS	y,[r+s*y],y	34	34				AVX2
VPGATHEROPS	x,[r+s*x],x	15	15			8	AVX2
VPGATHERDER	x,[r+s*y],x	22	22			7	AVX2
VPGATHERDPD	x,[r+s*x],x	12	12			7	AVX2
VPGATHEROPD	y,[r+s*x],y	20	20			9	AVX2
VPGATHERQPD	x,[r+s*x],x	14	14			7	AVX2
VPGATHERQPD	y,[r+s*y],y	22	22			9	AVX2
Conversion							
CVTPD2PS	x,x	2	2	p1 p5	4	1	
CVTPD2PS	x,m128	2	3	p1 p5 p23		1	
1	2.,	_	, ,	F . PO P-0	I		ı

1		1	1	I	1	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	у,у	1	1	p1	3	1	AVX
VCVTDQ2PS	y,m256	1	2	p1 p23		1	AVX
CVT(T) PS2DQ	x,x	1	1	p1	3	1	,,,,
CVT(T) PS2DQ	x,m128	1	2	p1 p23		1	
VCVT(T) PS2DQ	у,у	1	1	p1	3	1	AVX
VCVT(T) PS2DQ	y,y y,m256	1	2	p1 p23		1	AVX
CVTDQ2PD	y,111230 X,X	2	2	p1 p25	4	1	AVA
CVTDQ2PD		2	2		4	1	
	x,m64	2	2	p1 p23	6	1	A) //
VCVTDQ2PD	y,x	2	2	p1 p5	0	1	AVX
VCVTDQ2PD	y,m128	1		p1 p23			AVX
CVT(T)PD2DQ	X,X	2	2	p1 p5	4	1	
CVT(T)PD2DQ	x,m128	2	3	p1 p5 p23		1	A) () (
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		1	
CVTPI2PD	x,mm	2	2	p1 p5	4	1	
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,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	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,m	2	2	p1 p23	-	1	F16C
Arithmetic							
ADDSS/D PS/D							
SUBSS/D PS/D	x,x / v,v,v	1	1	p1	3	1	
ADDSS/D PS/D							
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,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	,			40.500			0050
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,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,y	3	3	2p0 p15	19-35	16-28	AVX
VDIVPD	y,y,m256	4	4	2p0 p15 p23		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 CMPccPS/D	x,x / v,v,v	1	1	p1	3	1	
CMPccSS/D							
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	x,x,i	3	3	p0 p1 p5	9	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/PS	x,x	1	1	0q	11	7	
SQRTSS/PS	x,m128	1	2	p0 p23	'''	7	
VSQRTPS		3	3	2p0 p15	19	14	AVX
VSQRTPS	y,y y,m256	4	4	2p0 p15 2p0 p15 p23	10	14	AVX
SQRTSD/PD	y,111230 X,X	1	1	p0	16	8-14	AVA
SQRTSD/PD	x,m128	1 1	2	p0 p23	10	8-14	
VSQRTPD	y,y	3	3	2p0 p15	28-29	16-28	AVX
VSQRTPD	y,y y,m256	4	4	2p0 p15 2p0 p15 p23	20-23	16-28	AVX
RSQRTSS/PS	y,111230 X,X	1	1	p0	5	10-20	AVA
RSQRTSS/PS	x,m128	1 1	2	p0 p23	5	1	
1.13611136/10	۸,111120	'	ı -	PO P20		ı .	

VRSQRTPS	y,y	3	3	2p0 p15	7	2	AVX
VRSQRTPS	y,m256	4	4	2p0 p15 p23		2	AVX
Logio							
Logic AND/ANDN/OR/XO							
RPS/PD	x,x / v,v,v	1	1	p5	1	1	
AND/ANDN/OR/XO				·			
RPS/PD	x,m / v,v,m	1	2	p5 p23		1	
0.11							
Other							
VZEROUPPER		4	4	none		1	AVX
V.7550444		4.0	4.0			4.0	AVX,
VZEROALL		12	12	none		10	32 bit
V/7EDOALI		-00	00				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	
VSTMXCSR	m32	3				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,

(x)mm = 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 domain:

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

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 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 in-

structions of the same kind in the same thread.

Integer instructions

Instruction	Operands	µops fused domain	µops 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	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)		3	4	p1 p4 p237 p06		1	
PUSHA(D)		11	19			8	not 64 bit
POP	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			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 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	
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		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	
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	

			2.0	aavvon			
ADD SUB ADD SUB	r,m m,r/i	1 2	2 4	p0156 p23 2p0156 2p237 p4	6	0.5 1	
ADC SBB	r,r/i	1	1	p06	1	1	
ADC SBB	r,m	2	2	p06 p23		1	
ADC SBB	m,r/i	4	6	3p0156 2p237 p4	7	2	
СМР	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	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 IMUL	r64,r64,i	1	1	p1	3	1	
IMUL	r16,m16,i r32,m32,i	2 1	3 2	p1 p0156 p23 p1 p23		1	
IMUL	r64,m64,i	1	2	p1 p23		1	
MULX	r32,r32,r32	3	3	p1 2p056	4	1	AVX2
MULX	r32,r32,m32	3	4	p1 2p056 p23	т	1	AVX2
MULX	r64,r64,r64	2	2	p1 p5	4	1	AVX2
MULX	r64,r64,m64	2	3	p1 p6 p23		1	AVX2
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- 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	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	F		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	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	1	p1	3	1	BMI1
TZCNT	r,m	1	2	p1 p23		1	BMI1
ANDN	r,r,r	1 1	1	p15	1	0.5	BMI1
ANDN	r,r,m	1 1	2	p15 p23	1	0.5	BMI1
BLSI BLSMSK BLSR	r,r	1	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	r,m,r	3	3	2p0156 p23		1	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	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 1	2	p1 p23	U	1	BMI2
LXI	1,1,111	'	_	p 1 p23		'	DIVITZ
Control transfer	instructions						
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
Sorialisma jamp	onoronoa.		•	Po			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 branch		1	1	p06		0.5-1	predicted not 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		2	
CALL	r	2	3	p237 p4 p6		2	
CALL	m	3	4	2p237 p4 p6		3	
RET		1	2	p237 p6		1	
RET	İ	3	4	p23 2p6 p015		2	
BOUND	r,m	15	15			8	not 64 bit
INTO		4	4			5	not 64 bit
String instruc-							
LODSB/W		3	3	2p0156 p23		1	
LODSD/Q		2	2	p0156 p23		1	
REP LODS		5n+12	_	po 100 p20		~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
ILLI 0100		2.0/020				17020	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			6	
LOCK XADD	m,r	9	9			21	
LOCK ADD	m,r	8	8			21	
CMPXCHG	m,r	5	6			7	
LOCK CMPXCHG	m,r	10	10			21	
CMPXCHG8B	m,r	15	15			8	
LOCK CMPXCHG8B	m,r	19	19			21	
CMPXCHG16B	m,r	22	22			15	
LOCK CMPXCHG16B	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	

			D.O.	advvon		
FLDPI FLDL2E et FCMOVcc FNSTSW FNSTSW FLDCW FNSTCW FINCSTP FDECS FFREE(P) FNSAVE FRSTOR	r AX m16 m16 m16	2 3 2 2 3 2 1 1 152 95	2 3 2 3 3 1 1 152 95	2p01 2p0 p5 p0 p0156 p0 p4 p237 p01 p23 p6 p237 p4 p6 p01 p01	2 6 6 7 6 0 173 175	2 2 1 1 2 1 0.5 0.5 173 175
Arithmetic in- structions						
FADD(P) FSUB(R)(P) FADD(P)	r	1	1	p1	3	1
FSUB(R)(P) FMUL(P)	m r	1 1	2	p1 p23 p0	5	1
FMUL(P) FDIV(R)(P) FDIV(R)(P)	m r m	1 1 1	2 1 2	p0 p23 p0 p0 p23	10-15	1 4-5 4-5
FABS FCHS FCOM(P) FUCOM	r	1 1 1	1 1 1	p0 p0 p1	1 1 3	1 1 1
FCOM(P) FUCOM FCOMPP FUCOM	m IPP	1 2	2 2	p1 p23 2p01		1 1
FCOMI(P) FUCOMI(P) FIADD FISUB(R) FIMUL FIDIV(R)	r m m m	3 2 2 2	3 3 3 3	3p01 2p1 p23 p0 p1 p23 p0 p1 p23	7	1.5 2 2
FICOM(P) FTST FXAM	m	2 1 2	3 1 2	2p1 p23 p1 2p1	3	2 1 2
FPREM FPREM1 FRNDINT		28 28 17	28 28 17		20-24 23-48 11	13 13 23
Math FSCALE		27	27		125	130
FXTRACT FSQRT FSIN		17 1 75-100	17 1	р0	12 10-23 48-106	11 4-9
FCOS FSINCOS F2XM1		70-100 70-110 70-110 16-86			49-112 52-124 63-68	
FYL2X FYL2XP1 FPTAN		55-96 56 71-102			92 74 132	
FPATAN		27-71			97-147	
Other FNOP		1	1	p01		0.5

١	WAIT	2	2	p01	1	
F	FNCLEX	5	5	p0156	22	
ı	FNINIT	26	26		84	

Integer vector instructions

Integer vector	Instructions	5	T		T		
Instruction	Operands	μορs fused domain	µops unfused domain	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc-	o por unuo	u o i i i u		popo odon port		Par	
tions							
MOVD	r32/64,(x)mm	1	1	p0	1	1	
MOVD	m32/64,(x)mm	1	2	p237 p4	3	1	
MOVD	(x)mm,r32/64	1 1	1	p5	1	1	
MOVD	(x)mm,m32/64	1 1	1 1	p23	3	0.5	
MOVQ	r64,(x)mm	1	1 1	p0	1	1	
MOVQ	(x)mm,r64	1 1	1 1	p5	1		
MOVQ			I	· ·		1 -	
	(x)mm,(x)mm	1		p015	1	0.33	
MOVQ	(x)mm,m64	1	1	p23	3	0.5	
MOVQ	m64, (x)mm	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	7,47,42
PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW	''''''			ρυ			
PACKUSWB	mm,m64	3	3	p23 2p5		2	
	111111,11104	3	3	p23 2p3			
PACKSSWB/DW PACKUSWB	V V / V V V	1	1	n5	1	1	
	x,x / y,y,y	'	I	p5	'	'	
PACKSSWB/DW PACKUSWB				-00 -5			
	x,m / y,y,m	1	2	p23 p5	_	1	00544
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	

I	I	1	l	1		1	ı ı
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	000_0
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,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	18-500	1	
MASKMOVDQU	x,x	10	10	4p04 2p56 4p23	18-500	6	
VPMASKMOVD/Q	v,v,m	3	3	p23 2p5	4	2	AVX2
VPMASKMOVD/Q	m,v,v	4	4	p0 p1 p4 p23	15	1	AVX2
PMOVMSKB	r,v	1	1	p0	3	1	
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	(x)mm,r32,i	2	2	p5	2	2	
PINSRW	(x)mm,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 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	v m 22/64	_	_	~00	4	0.5	A) (VO
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),,,,						7.17.2
B/W	y,m8/16	3	3	p01 p23 p5	7	1	AVX2
VPBROADCAST					_		
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
VPGATHERDO	x,[r+s*y],x	10	10			6	AVX2
VPGATHERDO	x,[r+s*x],x	7	7			5	AVX2
VPGATHERDQ	y,[r+s*x],y	9 7	9 7			6 5	AVX2
VPGATHERQQ VPGATHERQQ	x,[r+s*x],x	9	9			6	AVX2 AVX2
VEGATHERQQ	y,[r+s*y],y	9	9			0	AVAZ
Arithmetic in-							
structions							
PADD/SUB(S,US)							
B/W/D/Q	v,v / v,v,v	1	1	p15	1	0.5	
PADD/SUB(S,US) B/W/D/Q				-45 -00		0.5	
	v,m / v,v,m	1	2	p15 p23		0.5	
PHADD(S)W/D		3	3	n1 2n5	3	2	CCCE2
PHSUB(S)W/D	v,v / v,v,v	3	3	p1 2p5	3		SSSE3
PHADD(S)W/D	ym /yym	4	4	n1 2n5 n22		2	SSSE3
PHSUB(S)W/D PCMPEQB/W/D	v,m / v,v,m	4	4	p1 2p5 p23			333E3
PCMPGTB/W/D	v,v / v,v,v	1	1	p15	1	0.5	
PCMPEQB/W/D	, v, v, v, v, v	•	'	Pio		0.0	
PCMPGTB/W/D	v,m / v,v,m	1	2	p15 p23		0.5	
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	р0	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	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				I			
SB/SW/SD							
UB/UW/UD	x,x / y,y,y	1	1	p15	1	0.5	SSE4.1
PMIN/PMAX	,,,,,,	-		P	-		
SB/SW/SD							
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	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
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	COOLO
PSADBW	v,m / v,v,m	1	2	p0 p23		1	
MPSADBW	x,x,i / v,v,v,i	3	3	p0 p25	6	2	SSE4.1
MPSADBW	x,m,i / v,v,m,i	4	4	p0 2p5 p23		2	SSE4.1
INIFOADBVV	X,111,1 / V,V,111,1	4	4	ρυ 2μ3 μ23			33E4.1
Logic instruc-							
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		_		po po p=0			002
PSRLW/D/Q							
PSRAW/D/Q	mm,mm	1	1	р0	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	·	0		005.040	_		00540
PCMPESTRI	X,X,İ	8	8	6p05 2p16	4	4	SSE4.2
PCMPESTRI	x,m128,i	8	8	3p0 2p16 2p5 p23	44	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		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	11	11	SSE4.2
PCMPISTRM	x,m128,i	4	4	3p0 p23		3	SSE4.2
Enomination instru	uotiono						
Encryption instru			_	0	_	_	01.841.11
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	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	17	2	AES
AESKEYGENAS SIST	x,x,i	10	10	2p0 8p5	10	9	AES
AESKEYGENAS							
SIST	x,m,i	10	10	2p0 p23 7p5		8	AES
Other							
EMMS		31	31			12	

Floating point XMM and YMM instructions

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

1	i	1		1	ı	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
VPGATHERDPS	x,[r+s*x],x	10	10	F- F- F- P-0		6	AVX2
VPGATHERDPS	y,[r+s*y],y	14	14			7	AVX2
VPGATHERQPS	x,[r+s*x],x	9	9			6	AVX2
VPGATHERQPS	x,[r+s*y],x	10	10			6	AVX2
VPGATHERQPS	x,[r+s*x],x	7	7			5	AVX2 AVX2
VPGATHERDPD	y,[r+s*x],y	9	9			6	AVX2 AVX2
VPGATHERQPD	x,[r+s*x],x	7	7			5	AVX2
IN OVALLE OF D	, ,,,, · · · · · ,,,	· •	ı •	I	I	1	, , , , , , , ,

			Di O	aawen			
VPGATHERQPD	y,[r+s*y],y	9	9			6	AVX2
Conversion							
CVTPD2PS		2	2	p1 p5	4	1	
CVTPD2PS	x,x x,m128	2	3	p1 p5 p23	4		
VCVTPD2PS	· ·	2	2	p1 p3 p23	5		AVX
VCVTPD2PS	x,y x,m256	2	3	p1 p5 p23		1	AVX
CVTSD2SS	· ·	2	2		4	1	AVA
CVTSD2SS	x,x x,m64	2	3	p1 p5	4	1	
CVTSD2SS CVTPS2PD		2	2	p1 p5 p23	2	1	
CVTPS2PD	X,X	2	2	p0 p5		1	
	x,m64			p0 p23	_		A) ()/
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
CVT(T) PS2DQ	X,X	1	1	p1	3	1	
CVT(T) PS2DQ	x,m128	1	2	p1 p23		1	
VCVT(T) PS2DQ	y,y	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		1	
CVTPI2PD	x,mm	2	2	p1 p5	4	1	
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,İ	2	2	p1 p5	4-6	1	F16C
VCVTPS2PH	m,v,i	3	3	p1 p4 p23	- 3	1	F16C
VCVTPH2PS	V,X	2	2	p1 p4 p23	4-6	1	F16C
VCVTPH2PS	v,x v,m	2	2	p1 p23	7-0	1	F16C
1,000 11 1121 0	, v, i i i	_	_	P1 P20	1	ı .	, ,,,,,

				1			
Arithmetic							
ADDSS/D PS/D							
SUBSS/D PS/D	x,x / v,v,v	1	1	p1	3	1	
ADDSS/D PS/D							
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,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,x / v,v,v	1	1	p01	3	0.5	
MULSS/D PS/D	x,m / v,v,m	1	2	p01 p23		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	x,m	1	2	p0 p23		4-5	
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							
CMPccPS/D	x,x / v,v,v	1	1	p1	3	1	
CMPccSS/D			_				
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	
DOLUMBOO(D DO(D	:	0		04	0	_	00544
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	2p1 p23 2p0 p1 p5	12	2	SSE4.1
DPPS	x,m,i / v,v,m,i	6	6	2p0 p1 p5 p23 p6	12	4	SSE4.1
DPPD	x,x,i	3	3	p0 p1 p5	7	1	SSE4.1
DPPD	x,x,i x,m128,i	4	4	p0 p1 p5 p23	1	1	SSE4.1
VFMADD	۸,۱۱۱۱۷۵,۱	7	•	P0 P1 P3 P23		'	JUL4.1
(all FMA instr.)	VVV	1	1	p01	5	0.5	FMA
VFMADD	V,V,V	ı	'	ροι	J	0.5	1 1717
(all FMA instr.)	v,v,m	1	2	p01 p23		0.5	FMA
(all I WA IIIsti.)	v, v,111	ı		ρυτ μ23		0.5	1 1717
Math							
SQRTSS	x,x	1	1	p0	11	4	
-455	Λ,Λ	•			• • •	'	

SQRTPS	x,x	1	1	0q	11	7	
SQRTSS/PS	x,m128	1	2	p0 p23		4-7	
VSQRTPS	y,y	3	3	2p0 p15	19	14	AVX
VSQRTPS	y,m256	4	4	2p0 p15 p23		14	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	y,y	3	3	2p0 p15	7	2	AVX
VRSQRTPS	y,m256	4	4	2p0 p15 p23		2	AVX
Logic							
AND/ANDN/OR/XO	,			_	4	_	
RPS/PD	x,x / v,v,v	1	1	p5	1	1	
AND/ANDN/OR/XO RPS/PD	v m / v v m	1	2	nF n22		1	
TKI O/I D	x,m / v,v,m	'		p5 p23		'	
Other							
VZEROUPPER		4	4	none		1	AVX
VZEROOFFER				110110			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,

(x)mm = 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.

μορs fused domain:

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

nain: 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 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 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.

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	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 addressing modes

1			1	- !			1
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 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	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	;	1	2	p237 p4	3	1	
PUSH	m	2	3	p4 2p237		1	
PUSH	stack pointer	2	3	p0156 p237 p4		1	
PUSHF(D/Q)	Stack politici	3	4	p1 p4 p237 p4		1	
PUSHA(D)		11	19	p1 p4 p237 p00		8	not 64 bit
POP	r	1	19	22	2	0.5	1101 04 011
	· ·			p23	2		
POP	stack pointer	3	3	p23 2p0156		3	
POP	m	2	3	2p237 p4		1	
POPF(D/Q)		9	9			20	
POPA(D)		18	18			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	
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
MOVBE	11104,104	3		poo p 13 p237 p4		'	WOVBE
PREFETCHNTA/ 0/1/2	m	1	1	p23		0.5	
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 OUD	<i>t</i>	۱ ،				0.05	I I
ADD SUB	r,r/i	1	1	p0156	1	0.25	
ADD SUB	r,m	1 2	2	p0156 p23	_	0.5	
ADD SUB	m,r/i		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	
ADC SBB	m,r/i	4	6	3p0156 2p237 p4	5	2	
7.2002	,					_	
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	1 1	p0 p1 p3 p0	3	1	HOL O4 DIL
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	3	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	1	
IMUL	r,r	1	2	p1 p1 p23	3	1	
IMUL	r,m r16,r16,i	2	2	p1 p0156	4	1	
IMUL	r32,r32,i	1	1	p1 p0130	3	1	
IMUL	r64,r64,i		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	AVX2
MULX	r32,r32,m32	3	4	p1 2p056 p23	-	1	AVX2
MULX	r64,r64,r64	2	2	p1 p5	4	1	AVX2
MULX	r64,r64,m64	2	3	p1 p6 p23		1	AVX2
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	104	1	1	p0 p1 p3 p0 p0156	1	<u>∠</u> -7-30	
CWDE			1	p0156	1		
CDQE		1	1	p0156	1		
CWD		2	2	p0156	1		
CDQ		1	1	p0156	1		
		ļ !	'		ı		

000	I	1	1 4	206	1 4		
CQO		1	1	p06	1	4	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- 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	5	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	01101111011111
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	2p00 p0130		3	
RCR RCL	r,i	8	8	p0156	6	6	
RCR RCL	m,i	11	11	p0130		6	
RCR RCL	r,cl	8	8	p0156	6	6	
RCR RCL	m,cl	11	11	p0130		6	
SHRD SHLD	r,r,i	1	1	p1	3	1	
SHRD SHLD	m,r,i	3	5	Pi		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	p0130	, ,	4	
SHLX SHRX SARX		1	1	p06	1	0.5	BMI2
SHLX SHRX SARX	r,r,r r,m,r	2	2	p06 p23	'	0.5	BMI2
RORX	r,r,i	1	1	p06 p25	1	0.5	BMI2
RORX	r,m,i	2	2	p06 p23	'	0.5	BMI2
BT	r,r/i	1	1	p06 p25	1	0.5	DIVIIZ
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 p23	1	0.5	
BTR BTS BTC	m,r	10	11	ροσ	'	5	
BTR BTS BTC	m,i	3	4	n06 n4 n23		1	
BSF BSR		1		p06 p4 p23	3		
	r,r		1	p1)	1	
BSF BSR	r,m	1	2	p1 p23	4	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

l. =0.1=	1			1 4 00 1		1	
LZCNT	r,m	1	2	p1 p23	_	1	LZCNT
TZCNT	r,r	1 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 BLSR	r,r	1	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	r,m,r	3	3	2p0156 p23		1	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	3	1	BMI2
PDEP	r,r,m		2	p1 p23	3	1	BMI2
PEXT			1	p1	3	1	BMI2
PEXT	r,r,r		2	1 '	3	1	BMI2
PEXI	r,r,m	'	2	p1 p23		ı	DIVIIZ
Control transfer	instructions						
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	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 branch		1	1	p06		0.5-1	predicted not taken
J(E/R)CXZ	short	2	2	p0156 p6		0.5-2	taitori
LOOP	short	7	7	p0130 p0		5	
LOOP(N)E	short	11	, 11			6	
CALL	near	2	3	p237 p4 p6		3	
CALL		1		1			
	r	2	3	p237 p4 p6		2	
CALL RET	m	3	4	2p237 p4 p6		3	
	:	'	2	p237 p6		1	
RET	i	4.5	2			2	
BOUND	r,m	15	15			8	not 64 bit
INTO		5	5			6	not 64 bit
String instruc-							
LODSB/W		3	3	2p0156 p23		1	
LODSD/Q		2	2	p0156 p23		1	
REP LODS		5n+12	2	μυ 130 μ23		•	
		1	2	n00 n04E0 4		~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 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	
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 1F)		1	0	none		0.25	
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
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	μops 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 FLD1 FLDPI FLDL2E et FCMOVcc FNSTSW FNSTSW FLDCW FNSTCW FINCSTP FDECS FFREE(P) FNSAVE FRSTOR	r AX m16 m16 m16	1 2 2 4 2 2 3 2 1 1 133 89	1 2 2 4 2 3 3 3 1 1 133 89	p05 2p05 2p05 p0 p1 p56 p0 p0156 p0 p4 p237 p01 p23 p6 p237 p4 p6 p05 p05	3 6 7 6 0 176 175	1 2 2 2 1 2 1 0.5 0.5 176 175
Arithmetic instructions 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 FCHS		1	1	p0	1	1
FCHS FCOM(P) FUCOM	r	1	1	р0 p5	1 3	1 1
FCOM(P) FUCOM	m '	1	2	p5 p23	3	1
FCOMPP FUCOM		2	2	p0 p5		1
FCOMI(P)				F - F -		
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 2	3	p0 p5 p23		2
FICOM(P) FTST	m m	1	3 1	2p5 p23 p5	3	2 1
FXAM		2	2	2p5	6	2
FPREM		31	31	200	26-30	17
FPREM1		31	31		30-57	17
FRNDINT		17	17		21	11
84 - 41-						
Math FSCALE		27	27		130	130
FXTRACT		17	17		11	11
FSQRT		1	1	р0	14-21	4-7
FSIN		53-105	-	F -	50-120	
FCOS		53-105			50-130	
FSINCOS		55-120			55-150	
F2XM1		16-90			65-80	
FYL2X		40-100			103	
FYL2XP1 FPTAN		56 40-112			77 140-160	
FPATAN		30-160			100-160	
					.55 100	

Other				
FNOP	1	1	p05	0.5
WAIT	2	2	p05	2
FNCLEX	5	5	p156	22
FNINIT	18	18		78

Integer vector instructions

Instruction	Operands	μορs fused domain	µops unfused domain	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc-				popo concer poss			
tions							
MOVD	r32/64,(x)mm	1	1	р0	2	1	
MOVD	m32/64,(x)mm	1	2	p237 p4	3	1	
MOVD	(x)mm,r32/64	1	1	p5	2	1	
MOVD	(x)mm,m32/64	1	1	p23	2	0.5	
MOVQ	r64,(x)mm	1	1	p0	2	1	
MOVQ	(x)mm,r64	1	1	p5	1	1	
MOVQ	mm,mm	1		p05	1	0.5	
MOVQ	x,x	1		p015	1	0.33	
MOVQ	(x)mm,m64	1	1	p23	2	0.5	
MOVQ	m64, (x)mm	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	j,00	_	_	p=0 p0 .0			7.07.
PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW	,				_	_	
PACKUSWB	mm,m64	3	3	p23 2p5		2	
PACKSSWB/DW	,			P=0 = 00			
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	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		_	1-365			
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			_	1-365			
QDQ	x,x / y,y,y	1	1	p5	1	1	

1	ı	ı	ı	1		ı	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	р5	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	
PSHUFD	v,v,i	1	1	p5	1	1	
PSHUFD	v,m,i	1-2	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	1	1	p015	1	1	SSE4.1
PBLENDVB	x,m,xmm0	2	2	p015 p23		2	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	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	~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	
PEXTRB/W/D/Q	r32,x,i	2	2	p0 p5	3	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	2p5	3	2	SSE4.1
PINSRB	x,m8,i	2	2	p23 p5		1	SSE4.1
PINSRW	(x)mm,r32,i	2	2	p5	3	2	
PINSRW	(x)mm,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
אינטועויטו		ı '	· '		1	ı '	/\V/\ <u>L</u>

1	I	ı	ı	- 	I	I	I
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	,			F - F -			
D/Q	y,m32/64	1	1	p23	3	0.5	AVX2
VBROADCASTI128	y,m128	1	1	p23	3	0.5	AVX2
VPGATHERDD	x,[r+s*x],x	4	4	p0 p1 p23 p5		4	AVX2
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)							
B/W/D/Q PADD/SUB(S,US)	V,V / V,V,V	1	1	p015	1	0.33	
B/W/D/Q	v,m / v,v,m	1	2	p015 p23		0.5	
PHADD(S)W/D PHSUB(S)W/D	v,v / v,v,v	3	3	p01 2p5	3	2	SSSE3
PHADD(S)W/D PHSUB(S)W/D	v,m / v,v,m	4	4	p01 2p5 p23		2	SSSE3
PCMPEQB/W/D	, , ,						
PCMPGTB/W/D	mm,mm	1	1	p0	1	1	
PCMPEQB/W/D	,			'			
PCMPGTB/W/D	x,x / y,y,y	1	1	p01	1	0.5	
PCMPEQB/W/D	, 3,3,3			'			
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	1		1			1	SSSE3
	mm,mm	1		p0	5		
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	
PMIN/PMAX							
SB/SW/SD							
UB/UW/UD	mm,mm	1	1	p0	1	1	SSE4.1
		•		ρ σ			002
PMIN/PMAX SB/SW/SD							
UB/UW/UD		4	1	n01	1	0.5	SSE4.1
	x,x / y,y,y	1	1	p01	ı	0.5	SSE4.1
PMIN/PMAX							
SB/SW/SD							
UB/UW/UD	x,m / y,y,m	1	2	p01 p23		0.5	SSE4.1
PHMINPOSUW	X,X	1	1	p0	4	1	SSE4.1
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	p0 p0	1	1	SSSE3
PSIGNB/W/D	1	1	1	· ·	1	0.5	SSSE3
	x,x / y,y,y			p01	1		
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		•		Poo		0.0	
POR PXOR	x,x / y,y,y	1	1	p015	1	0.33	
	^,^ / y,y,y	'	l	ρυ15	'	0.55	
PAND PANDN		4		-045 00		0.5	
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							
PSRLW/D/Q							
PSRAW/D/Q	mm,mm	1	1	p0	1	1	
PSLLW/D/Q				-			
PSRLW/D/Q							
PSRAW/D/Q	mm,m64	2	2	p0 p23		1	
PSLLW/D/Q	,	-	_	F - P=0			
PSRLW/D/Q							
PSRAW/D/Q	x,x / v,v,x	2	2	p01 p5	1	1	
I SIVANNU/Q	_ ^,^ / V,V,X	4	4	pui po	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	20	1	1	
PSLLW/D/Q	mm,i	ı	'	р0	I	'	
PSRLW/D/Q PSRAW/D/Q	vi / vvi	1	1	n01	1	0.5	
VPSLLVD/Q	x,i / y,y,i	1	ļ .	p01	I	0.5	
VPSRLVD/Q	V,V,V	1	1	p01	1	0.5	AVX2
VPSLLVD/Q VPSRAVD							
VPSRLVD/Q	v,v,m	1	2	p01 p23		0.5	AVX2
PSLLDQ PSRLDQ	x,i / v,v,i	1	1	p5	1	1	
String instruc-							
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	12	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	ıctions						
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,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	μορs each port	Latency	Recipro- cal through put	Comments
Move instruc- tions	•					•	
MOVAPS/D VMOVAPS/D	x,x y,y	1	1 1	p015 p015	0-1 0-1	0.25 0.25	may eliminate may eliminate
MOVAPS/D MOVUPS/D	x,m128	1	1	p23	2	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 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	p23 p237 p4	3	1	
MOVHPS/D	x,m64	1	2		4		
MOVHPS/D	m64,x		2	p23 p5	3		
	1			p4 p237		· ·	
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,İ	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	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	р5	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

VDDOADCACTCD	v m64	1 4	1 4		ا ء	0.5	A)//
VBROADCASTSD	y,m64	1 1	1 1	p23	3	0.5 1	AVX AVX2
VBROADCASTSD	y,x	1 1		p5	3	0.5	AVX
VBROADCASTF128 MOVSH/LDUP	y,m128	1 1	1	p23		1	SSE3
MOVSH/LDUP	V,V			p5	1	· ·	SSE3
	v,m	1	1	p23	3	0.5	
UNPCKH/LPS/D	x,x / v,v,v	1	1 2	p5	I	1	SSE3
UNPCKH/LPS/D EXTRACTPS	x,m / v,v,m	1	2	p5 p23		1	SSE3
	r32,x,i	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 2	1 2	p5	1	1	SSE4.1
INSERTPS	x,m32,i			p23 p5	4	1	SSE4.1
VINSERTF128	y,y,x,i	1	1	p5	3	1	AVX
VINSERTF128	y,y,m128,i	2 2	2	p015 p23	5	0.5	AVX
VMASKMOVPS/D	v,v,m		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	13	1	AVX
VPGATHERDPS	x,[r+s*x],x	4	4	p0 p1 p23 p5	12	4	AVX2
VPGATHERDPS	y,[r+s*y],y	4	4	p0 p1 p23 p5	13	5	AVX2
VPGATHERQPS	x,[r+s*x],x	5	5	p0 p1 p23 p5		2 4	AVX2
VPGATHERQPS	x,[r+s*y],x	4	4	p0 p1 p23 p5		2	AVX2
VPGATHERDPD	x,[r+s*x],x	5	5	p0 p1 p23 p5			AVX2
VPGATHERDPD	y,[r+s*x],y	4	4	p0 p1 p23 p5		4	AVX2
VPGATHERQPD	x,[r+s*x],x	5 4	5 4	p0 p1 p23 p5		2 4	AVX2 AVX2
VPGATHERQPD	y,[r+s*y],y	4	4	p0 p1 p23 p5		7	AVAZ
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 p5 p23	_	0.5	A) /)/
VCVTPS2PD	y,x	2	2	p01 p5	7	1	AVX
VCVTPS2PD	y,m128	1	2	p01 p5 p23	_	0.5	AVX
CVTSS2SD	X,X	2	2	p01 p5	5	2 2	
CVTSS2SD	x,m32		2	p01 p5 p23	4		
CVTDQ2PS	X,X	1	1 2	p01	4	0.5 0.5	
CVTDQ2PS	x,m128	1	1	p01 p23	4		A) /V
VCVTDQ2PS	y,y	1	2	p01	4	0.5	AVX AVX
VCVTDQ2PS	y,m256	1 1		p01 p23		0.5	AVA
CVT(T) PS2DQ	X,X v m129	1	1 2	p01	4	0.5 0.5	
CVT(T) PS2DQ	x,m128	1	1	p01 p23	4	0.5	AVX
VCVT(T) PS2DQ	y,y y,m256		2	p01 p01 p23	4	0.5	AVX
VCVT(T) PS2DQ CVTDQ2PD	=	2	2	p01 p23	5	1	_ ^v^
CVTDQ2PD CVTDQ2PD	x,x x,m64	2	2	p01 p3	3	0.5	
VCVTDQ2PD	х,пю 4 у,х	2	2	p01 p23	7	1	AVX
VCVTDQ2PD	y,x y,m128	1	2	p01 p3	'	0.5	AVX
VOVIDQZED	y,111120	ļ ,	4	ροι μεσ		0.5	7/7

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		'	1	
CVTPI2PD		2	2	p0 p23	5		
	x,mm	1	2	p01 p5	5		
CVTPI2PD	x,m64			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 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		1	F16C
VCVTPH2PS	V,X	2	2	p01 p5	5-7	1	F16C
VCVTPH2PS	v,m	1	2	p01 p23		1	F16C
VCVIFIIZES	V,111	'		ρ01 μ23		l I	1 100
A with we atio							
Arithmetic	_						
ADDSS/D PS/D		4	4	m01	4	0.5	
SUBSS/D PS/D	x,x / v,v,v	1	1	p01	4	0.5	
ADDSS/D PS/D		_		m04 m00		0.5	
SUBSS/D PS/D	x,m / v,v,m	1	2	p01 p23		0.5	0050
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, <u></u>	1	1	p0	13-14	8	AVX
VDIVPD	y,y,m256	4	4	p0 p23		8	AVX
RCPSS/PS	y, y,111230 V,V	1	1	p0 p20	4	1	, , , , ,
RCPSS/PS		1 1	2	p0 p23	7	1	
INOF JOIF J	v,m	1		ρυ μεσ		ļ !	

CMPECPS/ID	OMD 00/D	ı		I	I	l	l	1
CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMMCSC/D CMPccPS/D CMCCMCC/D CMCCMCCMSC/D CMCCMCMSC/D CMCCMCCMSC/D CM	CMPccSS/D	,	4	_	0.4	_	0.5	
CMPecPSID		X,X / V,V,V	1	1	p01	4	0.5	
U)COMISS/D								
IU)COMISS/ID		x,m / v,v,m	2	2			0.5	
MAXSS/D PS/D MINSS/D MINSS/D PS/D MINS	(U)COMISS/D	X,X	1	1	p0		1	
MINSS/D PS/D	(U)COMISS/D	x,m32/64	2	2	p0 p23		1	
MINSS/D PS/D	MAXSS/D PS/D							
MAXSS/ID PS/ID x,m / v,v,m 1 2 p01 p23 0.5 ROUNDSS/ID PS/ID v,v,i 2 2 2p01 8 1 SSE4.1 ROUNDSS/ID PS/ID v,m,i 3 3 2p01 p23 1 SSE4.1 DPPS x,X,I / v,v,v,i 4 4 3p01 p25 13 1.5 SSE4.1 DPPS x,x,I / v,v,w,i 6 6 3p01 p23 p5 p6 1.5 SSE4.1 DPPD x,x,i 3 3 2p01 p5 9 1 SSE4.1 DPPD x,x,i 3 3 2p01 p2 9 1 SSE4.1 DPPD x,x,i 4 4 2p01 p23 p5 1 SSE4.1 DPPD x,x,i 1 1 p01 4 0.5 FMA VFMADD (all FMA instr.) v,v,w 1 1 p01 4 0.5 FMA Math Xx 1 1 p0 1 2	1	x.x / v.v.v	1	1	p01	4	0.5	
MINSS/D PS/D		, , ,			F -			
ROUNDSS/D PS/D		v m / v v m	1	2	n01 n23		0.5	
ROUNDSS/D PS/D DPPS	WIII VOO/D T O/D	A,111 / V, V,111	ı	_	po 1 p25		0.5	
ROUNDSS/D PS/D DPPS	POLINDSS/D PS/D	vvi	2	2	2n01	, a	1	SSF4 1
DPPS	IKOONDSS/D I S/D	v, v,ı	_	_	Ζρο 1		'	OOL4.1
DPPS	DOLINDSS/D DS/D	v m i	3	3	2n01 n23		1	SSE/ 1
DPPS						12		
DPPD						13		
DPPD								
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 VSQRTSS/PS VSQRTPS x,x 1 1 p0 p23 3 VSQRTPS VSQRTPS VSQRTPS y,y 1 1 p0 p23 3 4 AVX VSQRTPS VSQRTPD y,m256 4 4 p0 p23 6 AVX SQRTSD/PD VSQRTPD x,x 1 1 p0 15-16 4-6 AVX VSQRTPD VSQRTPD x,m128 1 2 p0 p23 4-6 AVX VSQRTPD VSQRTPD y,m256 4 4 p0 p23 9-12 AVX VSQRTPD VSQRTSS/PS v,v 1 1 p0 4 1 AVX RSQRTSS/PS v,v 1 1 p0 p23 9-12 AVX VSQRTPD VSQRTPD x,x/v,v,v 1 1 p0 p23 1			3	3		9	1	
(all FMA instr.) v,v,v 1 1 p01 4 0.5 FMA VFMADD v,v,m 1 2 p01 p23 0.5 FMA Math SQRTSS/PS x,x 1 1 p0 12 3 SQRTSS/PS 3 VSQRTPS y,m256 4 4 p0 p23 6 AVX AVX SQRTSD x,x 1 1 p0 15-16 4-6 AVX SQRTSD x,x 1 1 p0 15-16 4-6 AVX SQRTSD/PD x,m128 1 2 p0 p23 6 AVX AVX SQRTSD/PD x,m128 1 2 p0 p23 4-6 AVX AVX AVX AVX SQRTSD/PD x,m128 1 2 p0 p23 4-6 AVX <td>DPPD</td> <td>x,m128,i</td> <td>4</td> <td>4</td> <td>2p01 p23 p5</td> <td></td> <td>1</td> <td>SSE4.1</td>	DPPD	x,m128,i	4	4	2p01 p23 p5		1	SSE4.1
(all FMA instr.) v,v,v 1 1 p01 4 0.5 FMA VFMADD v,v,m 1 2 p01 p23 0.5 FMA Math SQRTSS/PS x,x 1 1 p0 12 3 SQRTSS/PS 3 VSQRTPS y,m256 4 4 p0 p23 6 AVX AVX SQRTSD x,x 1 1 p0 15-16 4-6 AVX SQRTSD x,x 1 1 p0 15-16 4-6 AVX SQRTSD/PD x,m128 1 2 p0 p23 6 AVX AVX SQRTSD/PD x,m128 1 2 p0 p23 4-6 AVX AVX AVX AVX SQRTSD/PD x,m128 1 2 p0 p23 4-6 AVX <td>VFMADD</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>	VFMADD							
VFMADD (all FMA instr.) v,v,m 1 2 p01 p23 0.5 FMA Math SQRTSS/PS SQRTSS/PS VSQRTPS VSQRTPS VSQRTPS VSQRTPS VSQRTPS VSQRTPS VSQRTPS VSQRTPD VSQRTPD VSQRTPD VSQRTPD VSQRTPD VSQRTPD VSQRTPD VSQRTPD V,m256 V,v 1 1 p0 12 3 AVX SQRTSD/PD VSQRTPD VSQRTPD VSQRTPD VSQRTPD VSQRTPD VX,m 1 1 p0 15-16 4-6 AVX VSQRTPD VSQRTPD VSQRTSS/PS RSQRTSS/PS V,v 1 1 p0 15-16 4-6 AVX VSQRTPD VSQRTSS/PS RSQRTSS/PS V,v 1 1 p0 15-16 9-12 AVX VSQRTSD/PS RSQRTSS/PS V,v 1 1 p0 4 1 2 AVX RSQRTSS/PS RSQRTSS/PS V,vm 1 1 p0 4 1 2 AVX RPS/PD AND/ANDN/OR/XO RPS/PD x,x / v,v,v 1 1 p015 1 0.33 AVX VZEROALL LDMXCSR 4 4 none 1 AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A		v.v.v	1	1	p01	4	0.5	FMA
Math SQRTSS/PS X,X 1 1 2 p0 p23	1, ,	, ,			F -			
Math SQRTSS/PS X,X		vvm	1	2	n01 n23		0.5	EMA
SQRTSS/PS x,x 1 1 p0 p23 3 VSQRTPS y,y 1 1 p0 p23 3 VSQRTPS y,y 1 1 p0 p23 6 AVX VSQRTPS y,m256 4 4 p0 p23 6 AVX SQRTSD x,x 1 1 p0 p20 15-16 4-6 SQRTPD x,x 1 1 p0 p23 4-6 AVX SQRTSD/PD x,m128 1 2 p0 p23 4-6 AVX VSQRTPD y,m256 4 4 p0 p23 9-12 AVX VSQRTPD y,m256 4 4 p0 p23 1 AVX RSQRTSS/PS v,m 1 2 p0 p23 1 1 Logic AND/ANDION/OR/XO x,x / v,v,v 1 1 p0 p5 p23 0.5 Other VZEROALL 25 25 p0 p1 p5 p6 12 32 bit mode	(all FIVIA IIISII.)	V, V, 111	ı		ρυτ μ23		0.5	FIVIA
SQRTSS/PS x,x 1 1 p0 12 3 SQRTSS/PS x,m128 1 2 p0 p23 3 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 VX VSQRTPD y,m256 4 4 p0 p23 9-12 AVX VSQRTSS/PS y,v 1 1 p0 4 1 RSQRTSS/PS y,w 1 1 p0 p23 1 AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX AVX </td <td>Moth</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>	Moth							
SQRTSS/PS x,m128 1 2 p0 p23 3 AVX VSQRTPS y,y 1 1 p0 12 6 AVX VSQRTSD y,m256 4 4 p0 p23 6 AVX SQRTSD x,x 1 1 p0 15-16 4-6 SQRTSD/PD x,m128 1 2 p0 p23 4-6 VSQRTPD VSQRTPD y,y 1 1 p0 15-16 9-12 AVX VSQRTPD y,y 1 1 p0 p23 9-12 AVX VSQRTPD y,m256 4 4 p0 p23 9-12 AVX RSQRTSS/PS v,v 1 1 p0 p23 1 1 Logic AND/AND/OR/XOR RPS/PD x,m / v,v,m 1 2 p0 p1 p23 0.5 Other VZEROALL 25 25 p0 p1 p5 p6 12 32 bit VZEROALL 34<		V V	1	4	20	12	2	
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 4-6 VSQRTPD y,m256 4 4 p0 p23 9-12 AVX VSQRTSS/PS v,v 1 1 p0 p23 9-12 AVX RSQRTSS/PS v,v 1 1 p0 p23 9-12 AVX RSQRTSS/PS v,v 1 1 p0 p23 1 1 Logic AND/ANDN/OR/XOR RPS/PD x,x / v,v,w 1 1 p0 p15 p23 0.5 Other VZEROALL 2 2 p0 p1 p5 p6 12 32 bit VZEROALL						12		
VSQRTPS y,m256 4 4 p0 p23 6 AVX SQRTSD x,x 1 1 p0 15-16 4-6 AVX SQRTPD x,x 1 1 p0 15-16 4-6 AVX VSQRTPD y,m128 1 2 p0 p23 4-6 AVX VSQRTPD y,m256 4 4 p0 p23 9-12 AVX RSQRTSS/PS v,v 1 1 p0 p23 9-12 AVX RSQRTSS/PS v,m 1 2 p0 p23 1 1 Logic AND/ANDN/OR/XO RPS/PD x,x / v,v,v 1 1 p015 1 0.33 AND/ANDN/OR/XO RPS/PD x,m / v,v,m 1 2 p015 p23 0.5 0.5 Other VZEROUPPER 4 4 none 1 AVX VZEROALL 25 25 p0 p1 p5 p6 12 32 bit VZEROALL 34 4								
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 p23 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/XO x,x / v,v,v 1 1 p0 p23 0.5 Other VZEROUPPER 4 4 none 1 AVX VZEROALL 25 25 p0 p1 p5 p6 12 32 bit VZEROALL 34 34 p0 p5 p6 p23 5 3 STMXCSR m32 3 4 p0 p5 p6 p23 5 2 FXSAVE m4096 <						12		
SQRTPD x,x 1 1 p0 15-16 4-6 SQRTSD/PD x,m128 1 2 p0 p23 4-6 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 1 Logic AND/ANDN/OR/XO x,x / v,v,v 1 1 p015 1 0.33 AND/ANDN/OR/XO x,m / v,v,m 1 2 p015 p23 0.5 0.5 Other VZEROUPPER 4 4 none 1 AVX VZEROALL 25 25 p0 p1 p5 p6 12 32 bit VZEROALL 34 34 4 p0 p1 p5 p6 p23 5 3 VZEROALL 32 3 4 p0 p4 p6 p23	VSQRTPS	y,m256	4	4	p0 p23		6	AVX
SQRTSD/PD x,m128 1 2 p0 p23 4-6 4-7 4-6 VX	SQRTSD	X,X	1	1	p0	15-16	4-6	
SQRTSD/PD x,m128 1 2 p0 p23 4-6 4-8 4-8 4-8 4-8 4-6 9-12 AVX AVX 4-6 9-12 AVX 4-8 4-8 9-12 AVX 4-8 4-8 9-12 AVX 4-8 4-8 9-12 AVX 4-8 4-8 9-12 AVX 4-8 4-8 9-12 AVX 4-8 4-8 9-12 AVX 4-8 4-8 9-12 AVX AVX 4-8 1-90 4 1	SQRTPD	X,X	1	1	0q	15-16	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 1 Logic AND/ANDN/OR/XO RPS/PD x,x / v,v,v 1 1 p015 1 0.33 AND/ANDN/OR/XO RPS/PD x,m / v,v,m 1 2 p015 p23 0.5 Other VZEROUPPER 4 4 none 1 AVX VZEROALL 25 25 p0 p1 p5 p6 12 32 bit VZEROALL 34 34 p0 p1 p5 p6 p23 5 3 VZEROALL 34 34 p0 p5 p6 p23 5 3 STMXCSR m32 4 4 p0 p5 p6 p23 5 3 STMXCSR m32 3 4 p0 p4 p6 p237 5 2 <td>SQRTSD/PD</td> <td></td> <td>1</td> <td>2</td> <td></td> <td></td> <td>4-6</td> <td></td>	SQRTSD/PD		1	2			4-6	
VSQRTPD y,m256 4 4 p0 p23 9-12 AVX RSQRTSS/PS v,v 1 1 p0 p23 4 1 RSQRTSS/PS v,m 1 2 p0 p23 1 1 Logic AND/ANDN/OR/XO RPS/PD x,x / v,v,v 1 1 p015 1 0.33 AND/ANDN/OR/XO RPS/PD x,m / v,v,m 1 2 p015 p23 0.5 Other VZEROUPPER 4 4 none 1 AVX VZEROALL 25 25 p0 p1 p5 p6 12 32 bit VZEROALL 34 34 p0 p1 p5 p6 12 64 bit VZEROALL 34 34 p0 p5 p6 p23 5 3 STMXCSR m32 3 4 p0 p4 p6 p237 5 2 FXSAVE m4096 106 78 78 32 bit mode FXRSTOR m4096 105 76 76 76 32 bit mode			1			15-16		Δ\/Υ
RSQRTSS/PS			=			13-10		
RSQRTSS/PS v,m		-						AVA
Logic AND/ANDN/OR/XO RPS/PD x,x / v,v,v 1 1 p015 1 0.33 AND/ANDN/OR/XO RPS/PD x,m / v,v,m 1 2 p015 p23 0.5 Other VZEROUPPER 4 4 none 1 AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A				I -	-	4		
AND/ANDN/OR/XO RPS/PD	RSQRTSS/PS	v,m	1	2	p0 p23		1	
AND/ANDN/OR/XO RPS/PD								
RPS/PD AND/ANDN/OR/XO RPS/PD x,x / v,v,v 1 1 p015 1 0.33 Other VZEROUPPER 4 4 none 1 AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A								
AND/ANDN/OR/XO RPS/PD		,			0.45			
Other VZEROUPPER 4 4 none 1 AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A		x,x / v,v,v	1	1	p015	1	0.33	
Other 4 4 none 1 AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A		,			0.45			
VZEROUPPER 4 4 none 1 AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A	RPS/PD	x,m / v,v,m	1	2	p015 p23		0.5	
VZEROUPPER 4 4 none 1 AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A								
VZEROALL 25 25 p0 p1 p5 p6 12 AVX, 32 bit AVX, AVX, AVX, AVX, AVX, AVX, AVX, AVX,			_				_	
VZEROALL 25 25 p0 p1 p5 p6 12 32 bit AVX, AVX, AVX, AVX, AVX, AVX, AVX, AVX,	VZEROUPPER		4	4	none		1	
VZEROALL 34 34 p0 p1 p5 p6 12 64 bit LDMXCSR m32 4 4 p0 p5 p6 p23 5 3 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								
VZEROALL 34 34 p0 p1 p5 p6 12 64 bit LDMXCSR m32 4 4 p0 p5 p6 p23 5 3 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	VZEROALL		25	25	p0 p1 p5 p6		12	32 bit
LDMXCSR m32 4 4 p0 p5 p6 p23 5 3 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								AVX,
LDMXCSR m32 4 4 p0 p5 p6 p23 5 3 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	VZEROALL		34	34	p0 p1 p5 p6		12	64 bit
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		m32	4	4		5	3	
FXSAVE m4096 106 78 78 32 bit mode FXSAVE m4096 136 64 64 64 bit mode FXRSTOR m4096 105 76 76 32 bit mode								
FXSAVE m4096 136 64 64 64 bit mode FXRSTOR m4096 105 76 76 32 bit mode			_	, ,	ρυ ρ τ ρυ ρ237			32 hit mada
FXRSTOR m4096 105 76 76 32 bit mode								
FXRSTOR								
	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 Pentium 4

List of instruction timings and µop 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.

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 op-

erand, etc.

μορs: Number of μops 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

different execution unit. There is no additional latency between ALU0 and

ALU1.

Reciprocal This is also called issue latency. This value indicates the number of clock cythroughput:Cles from the execution of an instruction begins to a subsequent independent

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.

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 differ-

ent ports.

Execution unit: Use this information to determine additional latency. When an instruction with

more than one uop 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 sub-

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	pops	Microcode	Latency	Additional latency	Reciprocal through-	Port	Execution unit	Subunit	Instruction set	Notes
Move instructions											
MOV	r,r	1	0	0,5	0.5-1	0,25	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	0,5	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	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	

1	ı	ı	ı	ı	1	ı	I	I	I		1 1
SFENCE		4	2			40				sse	
LFENCE		4	2			38				sse2	
MFENCE		4	2			100				sse2	
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	0, .	alao, i		86	c
ADD, SUB	m,r	3	0	≥ 8	0.0 .	≥ 4				86	c
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	'	ii it,aia		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		0/1	aluo/ i		86	c
INC, DEC	r	2	0	0,5	0.5-1		0/1	alu0/1		86	
INC, DEC	m	4	0	4	0.5-1	0,5	0/1	aluo/ i		86	
NEG	r	1	0	0,5	0.5-1		0	alu0		86	
NEG		3	0	0,5	0.5-1	0,5	U	aiuu		86	
AAA, AAS	m	4	27	90		2 3				86	
DAA, DAS		4	57	100						86	
AAD			10	22			4	int	formul	86	
		4					1	int	fpmul		
AAM	0/20	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	0,5	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		_			86	c
AND, OR, XOR	m,r	3	0	≥ 8		≥ 4				86	C
TEST	r,r	1	0	0,5	0.5-1		0	alu0		86	C
TEST	r,m	2	0	≥ 1	0.5-1			2.30		86	C
NOT	r	1	0		0.5-1		0	alu0		86	
1101	ı •	'	J	0,5	0.5-1	0,5	ı	aiuu	I	00	1 1

NOT	m	4	0	1		≥ 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	int	mmxsh	86	d
ROL, ROR		1	0	4	1		1	int		186	d
ROL, ROR	r,i	2	0	6	0	1 1	1		mmxsh	86	
	r,CL		_	4	1	1	-	int	mmxsh		d
RCL, RCR	r,1	1	0		0	· .	1	int	mmxsh	86	d
RCL, RCR	r,i	4	15	16	•	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	7-6	10	0	10	1	int	mmxsh	86	d
RCL, RCR	m,i/CL	4	18	18-28		14	1	int	mmxsh	86	d
		-	14	14	ı	14	1				u
SHLD, SHRD	r,r,i/CL	4			0			int	mmxsh	386	
SHLD, SHRD	m,r,i/CL	4	18	14	0	14	1	int	mmxsh	386	اء
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
BT	m,i	4	0	4	0	2	1	int	mmxsh	386	d
BT DTD DTG	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										
JMP	short/near	1	0	0	0	1	0	alu0	branch	86	
JMP	far	4	28	118		118	0			86	
JMP	r	3	0	4		4	0	alu0	branch	86	
JMP	m(near)	3	0	4		4	0	alu0	branch	86	
JMP	m(far)	4	31	11		11	0			86	
Jcc	short/near	1	0	0		2-4	0	alu0	branch	86	
J(E)CXZ	short	4	4	0		2-4	0	alu0	branch	86	
LOOP	short	4	4	0		2-4	0	alu0	branch	86	
CALL	near	3	0	2		2	0	alu0	branch	86	
CALL	far	4	34				0			86	
CALL	r	4	4	8			0	alu0	branch	86	
CALL	m(near)	4	4	9			0	alu0	branch	86	
CALL	m(far)	4	38				0			86	
RETN		4	0	2			0	alu0	branch	86	
RETN	i	4	0	2			0	alu0	branch	86	
RETF		4	33	11			0			86	

RETF	i	4	33	11			0			86	
IRET		4	48	24			0			86	
ENTER	i,0	4	12	26		26				186	
ENTER	i,n	4	45+2	4n		128+	16n		186		
LEAVE		4	0	3		3				186	
BOUND	m	4	14	14		14				186	
INTO		4	5	18		18				86	
INT	i	4	84	644						86	
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	ļ	200-5	00		p5			
RDTSC		4	7			80				p5	

Notes:

a) Add 1 µop if source is a memory operand.

Uses an extra µop (port 3) if SIB byte used. A SIB byte is needed if the memb)

ory operand has more than one pointer register, or a scaled index, or ESP is

used as base pointer.

Add 1 µop if source or destination, but not both, is a high 8-bit register (AH, c)

BH, CH, DH).

d) Has (false) dependence on the flags in most cases.

Not available on PMMX e)

Latency is 12 in 16-bit real or virtual mode, 24 in 32-bit protected mode. q)

Floating point x87 instructions

Instruction	Operands	nops	Microcode	Latency	Additional latency	Reciprocal through- put	Port	Execution unit	Subunit	Instruction set	Notes
Move instructions											
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	
					0		_				
FXCH	r	1	0	0	U	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	
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		_	_	'	U	6	· -				
	m16	4	4			_	0			87	
FNSTCW	m16	4	4			6	0			87	_
FLDCW	m16	4	7	(3)		(8)	0,2			87	f
Arithmetic instructions											
	_	1	_	_	4	1	1	fn	add	07	
FADD(P),FSUB(R)(P)	r		0	5	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	
FMUL	m	2	0	7	1	2	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	9,
FCHS		1	0	2	1	1	1	fp	misc	87	
FCOM(P), FUCOM(P)	r	1	0	2	0		1	fp	misc	87	
, , , , , , , , , , , , , , , , , , , ,		2	0	2	0	1		-			
FCOMPD FLICOMPD	m		_			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	≈175	≈207		≈207	1	fp		387	
FSINCOS	7	≈178	≈216		≈211	1	fp		387	
FPTAN	6	≈160	≈230		≈200	1	fp		87	
FPATAN	3	92	≈187		≈153	1	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	134	≈242		≈220	1	fp		87	
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

f) 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	sdon	Microcode	Latency	Additional latency	Reciprocal through-		Execution unit	Subunit	Instruction set	Notes
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	SHIIL	sse2	
MOVD	m32, r	2	0	≈ 8	U	2	0,1	load		mmx	
MOVQ	· /	1	0	6	0	1	0, 1	mov			
MOVQ	mm,mm	-	0	2	1	2		mov	ob:ft	mmx	
	xmm,xmm	1	-		ı		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	
PACKSSWB/DW											
PACKUSWB	mm,r/m	1	0	2	1	1	1	mmx	shift	mmx	а
PACKSSWB/DW						_					
PACKUSWB	xmm,r/m	1	0	4	1	2	1	mmx	shift	mmx	а
PUNPCKH/LBW/WD/	,		_								
DQ	mm,r/m	1	0	2	1	1	1	mmx	shift	mmx	а
PUNPCKHBW/WD/DQ/			0						- I- :£4	0	_
QDQ	xmm,r/m	1	0	4	1	2	1	mmx	shift	sse2	а
PUNPCKLBW/WD/DQ/QDQ	vmm r/m	4	0	2	4	2	4	mmy	ob:ft	222	
	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	0	mov		sse	
MASKMOVDQU	xmm,xmm	4	6	_		10	0	mov		sse2	
PMOVMSKB	r32,r	2	0	7	1	3	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
1 . ==	.,	- 1	-			, .,_		1		1	,

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	а
Other											
EMMS		4	11	12		12	0			mmx	

Notes:

a) Add 1 µop if source is a memory operand.

j) Reciprocal throughput is 1 for 64 bit operands, and 2 for 128 bit operands.

k) It may be advantageous to replace this instruction by two 64-bit moves

Floating point XMM instructions

Instruction	Operands	sdorl	Micro	Latency	Addi	Reci put	Port	Exec	Subunit	Instr	Notes
			Microcode	ncy	Additional latency	Reciprocal through-		Execution unit	unit	Instruction set	Š
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											
	r,m	3	0			4	2			sse	
MOVHPS/D, MOVLPS/D							_				
MOV/NTDC/D	m,r	2	0			2	0			sse	
MOVNTPS/D	m,r	2	0			4	0	f.a		sse/2	
MOVMSKPS/D	r32,r	2	0	6	1	3	1	fp	ob:ft	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	

I		I									
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	a
CVTDQ2PS	r,r/m	1	0	4	1	2	1	fp	011111	sse2	a
CVTDQ2PD	r,r/m	3	0	9	1	4	1	mmx-fp	sse2	a	
CVT(T)PS2DQ	r,r/m	1	0	4	1	2	1	fp	0002	sse2	а
CVT(T)PD2DQ	r,r/m	2	0	9	1	2	1	fp-mmx	sse2	a	u
CVTPI2PS	xmm,mm	4	0	10	1	4	1	mmx	3302	sse	а
CVTPI2PD	xmm,mm	4	0	11	1	5	1	fp-mmx	sse2	а	u
CVT(T)PS2PI	mm,xmm	3	0	7	0	2	0,1	fp-mmx	sse	a	
CVT(T)PD2PI	mm,xmm	3	0	11	1	3	0,1	fp-mmx	sse2	a	
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		3362	sse2	
1 ' '		2	0	8	1		1	fp			а
CVT(T)SS2SI	r32,xmm	2	0	0	ı	2,5	1	fp		sse	а
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	а
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	a
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	w w/ma	4		2	_	2	4	ma ma v	al		
ORF3/D XORF3/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	a
RSQRTPS	r,r/m	2	0	4	1	4	1	mmx		sse	а
Other											
Other			_	00		400					
LDMXCSR	m	4	8	98		100	1			sse	
STMXCSR Notes:	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 μops 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 uop uses more than one execution unit, only the first and the

last execution unit is listed.

Execution subunit:

Instruction set

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	hobs	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	1	alu1		x64	1 1
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	·	1	0		U	2	0	store		86	h a
MOV	m,r	2	0			2	0,3			86	b,c
	m,i	2				2		store			
MOV	m64,i32		0 2				0,3	store		x64	
MOV	r,sr	1				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	C
MOVZX	r16,r8	2	0	2	0	1	0/1	alu0/1		386	С
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		386	a,c,o
MOVSX	r,m	2	0	3	0	1	2	load		386	
MOVSXD	r64,r32	1	0	1	0	0,5	0	alu0		x64	a
CMOVcc	r,r/m	3	0	9,5	0	3				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		Ŭ	28	•			86	m
LODS	.,	1	3	8		8				86	'''
REP LODS		1	5n	≈ 4n+	50			86		00	
STOS		1	2	8	55	8		00		86	
REP STOS		1	2.5n	≈ 3n				86		50	
MOVS		1	4	8		8		00		86	
REP MOVSB		9		≈.3n				86		00	
REP MOVSW			~.5.1 ≈.5-1.1r	1	4n			86			
INCE INICARA		1	۱۱ .۱۱ -ن.ح	U- I	.411	1 1		00			1

				,				1				
REP MOVSD		1	≈1.1n	≈ 1.4	n			86				
REP MOVSQ		1	≈1.1n	≈ 1.4	n			x64				
BSWAP	r	1	0	1	0	1		alu		486		
IN, OUT	r,r/i	1	52			>100	0		86			
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	'	iiit,aia		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	0/1	aluo/ i		86	C	
INC, DEC	r	2	0	1	0	0,5	0/1	alu0/1		86		
INC, DEC		4	0	5	U	3	0/1	aluu/ l		86		
NEG	m r	1	0	1	0	0,5	0	alu0		86		
NEG		3	0	5	U	3	0	aluu		86		
	m	1	10	26		3				86	_ m	
AAA, AAS		1								86	m	
DAA, DAS AAD		2	16	29 13			4	int			m	
		2	5				1	int	mul	86	m	
AAM	0		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	٥.	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 IDIV IDIV CBW CWD CDQ CQO CWDE CDQE SCAS REP SCAS CMPS REP CMPS	r8/m8 r16/m16 r32/m32 r64/m64	1 1 1 1 2 2 1 1 1 1 1 1 1	21 19 19 58 0 0 0 0 0 3 ≈ 54+6 5 ≈ 81+8		0 0 0 0 0 0 0 0 0 0 ≈ 4n	34 34 91 1 1 1 1 1 8	1 1 1 0 0/1 0/1 0/1 0/1 0/1	int int int alu0 alu0/1 alu0/1 alu0/1 alu0/1	fpdiv fpdiv fpdiv fpdiv	86 86 386 x64 86 386 x64 386 x64 86	a a a
AND, OR, XOR		1	0	1	0	0,5	0	alu0		86	
AND, OR, XOR	r,r	2	0	1	0	1	U	aiuu		86	С
1	r,m	3	0	5	U	2				86	С
AND, OR, XOR TEST	m,r	ა 1	0	1	0	0,5	0	alu0		86	С
TEST	r,r	2	0	1	0	1	U	aluu		86	C
NOT	r,m r	1	0	1	0	0,5	0	alu0		86	
NOT		3	0	5	U	2	U	aluu		86	
SHL	m r,i	ა 1	0	1	0	0,5	1	alu1		186	
SHR, SAR	r8/16/32,i	1	0	1	0	0,5	1	alu i alu1		186	
SHR, SAR	r64,i	1	0	7	0	2	1	alu1 alu1		x64	
SHL	r,CL	2	0	2	0	2	1	alu1 alu1		86	
SHR, SAR	r8/16/32,CL	2	0	2	0	2	1	alu1		86	
SHR, SAR	r64,CL	2	0	8	0	_	1	alu1		x64	
ROL, ROR	r8/16/32,i	1	0	1	0	1	1	alu1		186	d
ROL, ROR	r64,i	1	0	7	0	7	1	alu1		x64	d
ROL, ROR	r8/16/32,CL	2	0	2	0	2	1	alu1		86	d
ROL, ROR	r64,CL	2	0	8	0	8	1	alu1		x64	d
RCL, RCR	r,1	1	0	7	0	7	1	alu1		86	d
RCL	r,i	2	11	31	0	31	1	alu1		186	d
RCR	r,i	2	11	25	0	25	1	alu1		186	d
RCL	r,CL	1	11	31	0	31	1	alu1		86	d
RCR	r,CL	1	11	25	0	25	1	alu1		86	d
SHL, SHR, SAR	m8/16/32,i	3	6	10	0		1	alu1		86	
ROL. ROR	m8/16/32,i	3	6	10	0		1	alu1		86	d
SHL, SHR, SAR	m8/16/32,cl	2	6	10	0		1	alu1		86	
ROL. ROR	m8/16/32,cl	2	6	10	0		1	alu1		86	d
RCL, RCR	m8/16/32,1	2	5	27	0	27	1	alu1		86	d
RCL, RCR	m8/16/32,i	3	13	38	0	38	1	alu1		86	d
RCL, RCR	m8/16/32,cl	2	13	37	0	37	1	alu1		86	d
SHLD, SHRD	r8/16/32,r,i	3	0	8	0	7	1	alu1		386	
SHLD	r64,r64,i	4	5	10	0		1	alu1		x64	
SHRD	r64,r64,i	3	7	10	0		1	alu1		x64	
SHLD, SHRD	r8/16/32,r,cl	4	0	9	0	8	1	alu1		386	
SHLD	r64,r64,cl	4	5	14	0		1	alu1		x64	

SHRD SHLD, SHRD SHLD, SHRD BT BT BT BT BT BTR, BTS, BTC BTR, BTS, BTC BTR, BTS, BTC BTR, BTS, BTC BTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, BTC CBTR, BTS, CBTC CBTR, CBTC CMC	r64,r64,cl m,r,i m,r,CL r,r m,i m,r r,i r,r m,i m,r r,r/m r	3 3 2 1 2 3 2 1 2 3 2 2 2 2 3 2 2 3 2 3	8 8 0 0 7 0 0 6 10 0 0	12 20 20 8 9 8 10 8 9 28 14 16 9	0 0 0 0 0 0 0 0 0 0	10 10 8 9 8 10 8 9 10 14 4 1 2 8	1 1 1 1 1 1 1 1 1 1 1	alu1 alu1 alu1 alu1 alu1 alu1 alu1 alu1		x64 386 386 386 386 386 386 386 386 386 386	d d d
CLD, STD		1	8	-	0	53				86	
Control transfer in atoms	liono										
Control transfer instruct	short/near	1	0	0	0	1	0	alu0	branch	86	
JMP	far	2	25	U	U	154	0	aluo	Diancii	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			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			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			86	
RETN		4	0			7	0	alu0	branch	86	
RETN	i	4	0			7	0	alu0	branch	86	
RETF		1	30			160	0			86	
RETF	i	2	30			160	0			86	
IRET		1	49			325	0			86 406	
BOUND INT	m i	2	11 67			12 470				186 86	m
INTO	I	1	4			26				86	m
		'	7			20				00	'''
Other											
NOP (90)		1	0	0		0,25		alu0/1		86	
Long NOP (0F 1F)		1	0	0		0,25	0/1	alu0/1		ppro	
PAUSE		1	2			50				sse2	
LEAVE		4	0	5		5				186	
CLI		1	5			52				86	
STI		1	11			64				86	
CPUID		1	49-90	ı	300-5	1		p5			
RDTSC		1	12			100				p5	

RDPMC (bit 31 = 1)	1	37	100	p5	
RDPMC (bit 31 = 0)	4	154	240	p5	
MONITOR				(s	se3)
MWAIT				(s	se3)

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

I) Move accumulator to/from memory with 64 bit absolute address (opcode A0 -

A3).

m) Not available in 64 bit mode.

n) Not available in 64 bit mode on some processors.

ο) MOVSX uses an extra μop 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.

p) 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 sign-extended 32-bit direct memory operand in 64-bit mode without RIP-relative address takes 2 μ ops because of the SIB byte. The throughput is 1 in this case.

You may use a MOV instead.

q) These values are measured in 32-bit mode. In 16-bit real mode there is 1 mi-

crocode µop and a reciprocal throughput of 17.

Floating point x87 instructions

Instruction	Operands	sdorl	Mic	Lat	Ado	Rec	Port	Exe	Sub	Inst	Notes
		38	Microcode	Latency	Additional latency	Reciprocal through- put	7	Execution unit	Subunit	Instruction set	les
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	

			•	10000	,,,,							
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)	r	1	0	8	1	2	1	fp	mul	87		
FMUL	m	2	0	8	1	2	1	fp	mul	87		
FIMUL	m16	3	3	8	1	8	1	fp	mul	87		
FIMUL	m32	3	0	8	1	3	1	fp	mul	87		
FDIV(R)(P)	r	1	0	45	1	45	1	fp	div	87	g,h	
FDIV(R)	m	2	0	45	1	45	1	fp	div	87	g,h	
FIDIV(R)	m16	3	3	45	1	45	1	fp	div	87	g,h	
FIDIV(R)	m32	3	3	45	1	45	1	fp	div	87	g,h	
FABS		1	0	3	1	1	1	fp	misc	87	9,	
FCHS		1	0	3	1	1	1	fp	misc	87		
FCOM(P), FUCOM(P)	r	1	0	3	0	1	1	fp	misc	87		
FCOM(P)	m	2	0	3	0	1	1	fp	misc	87		
FCOMPP, FUCOMPP		2	0	3	0	1	1	fp	misc	87		
FCOMI(P)	r	3	0		Ŭ	3	0,1	fp	misc	PPro		
FICOM(P)	m16	3	3			8	1	fp	misc	87		
FICOM(P)	m32	3	0			2	1,2	fp	misc	87		
FTST		1	0			1	1	fp	misc	87		
FXAM		1	0			1	1	fp	misc	87		
FRNDINT		3	14	28	1	16	0,1			87		
FPREM		8	86	220	1		1	fp		87		
FPREM1		9	92	220	1		1	fp		387		
								r				
Math												
FSQRT		1	0	45	1	45	1	fp	div	87	g,h	
FLDPI, etc.		2	0			2	1	fp		87		
FSIN, FCOS		3	≈100	≈200		≈200	1	fp		387		
FSINCOS		5	≈150	≈200		≈200	1	fp		387		
FPTAN		8	≈170			≈270	1	fp		87		
FPATAN		4	97	≈250		≈250	1	fp		87		
FSCALE		3	25	96			1	fp		87		
FXTRACT		4	16	27			1	fp		87		
F2XM1		3	190	≈270			1	fp		87		
FYL2X		3	63	≈170			1	fp		87		
FYL2XP1		3	58	≈170			1	fp		87		
								·				
			•			*		÷	•	•	•	

Other										
FNOP	1	0	1	0	1	0	mov	87		
(F)WAIT	2	0	0	0	1	0	mov	87		
FNCLEX	1	4			120	1		87		
FNINIT	1	30			200			87		
FNSAVE	2	181	500			0,1		87		
FRSTOR	2	96	570					87		
FXSAVE	2	121			160			sse	i	
FXRSTOR	2	118			244			sse	i	

Notes:

e) Not available on PMMX

f) 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 be-

tween 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.

i) Takes fewer microcode µops when XMM registers are disabled, but the

throughput is the same.

Integer MMX and XMM instructions

Instruction	Operands	sdori	Microcode	Latency	Additional latency	Reciprocal through- put	Port	Execution unit	Subunit	Instruction set	Notes
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		mmx	
MOVD	r32, xmm	1	0	7	1	1	0	fp		sse2	
MOVD	xmm, r32	2	0	4	1	2	1	mmx	shift	sse2	
MOVD	xmm,m32	1	0			1	2	load		sse2	
MOVD	m32, r	2	0			2	0,1			mmx	
MOVQ	mm,mm	1	0	7	0	1	0	mov		mmx	
MOVQ	xmm,xmm	1	0	2	1	2	1	mmx	shift	sse2	
MOVQ	r,m64	1	0			1	2	load		mmx	
MOVQ	m64,r	2	0			2	0	mov		mmx	
MOVDQA	xmm,xmm	1	0	7	0	1	0	mov		sse2	
MOVDQA	xmm,m	1	0			1	2	load		sse2	
MOVDQA	m,xmm	2	0			2	0	mov		sse2	
MOVDQU	xmm,m	4	0			23	2	load		sse2	k
MOVDQU	m,xmm	4	2			8	0	mov		sse2	k
LDDQU	xmm,m	4	0			2,5	2	load		sse3	
MOVDQ2Q	mm,xmm	3	0	10	1	2	0,1	mov-mmx	sse2		

Prescott

						1					
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											
MOVSLDUP	xmm,xmm	1	0	4	1	2	1	mmx	shift	sse3	
PACKSSWB/DW											
PACKUSWB	mm,r/m	1	0	2	1	2	1	mmx	shift	mmx	а
PACKSSWB/DW											
PACKUSWB	xmm,r/m	1	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/	,			_							
QDQ	xmm,r/m	1	0	4	1	4	1	mmx	shift	sse2	а
PUNPCKLBW/WD/DQ/Q			_						1 :60		
DQ	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		sse	
MASKMOVDQU	xmm,xmm	1	6			12	0	mov		sse2	
PMOVMSKB	r32,r	2	0	7		3	0,1	mmx-alu0	sse		
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		
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
	,					ĺ					
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	,
,	,								-		
1					i e	1				•	

Prescott

Other								
EMMS	10	10		12	0		mmx	

Notes:

a) Add 1 µop if source is a memory operand.

j) Reciprocal throughput is 1 for 64 bit operands, and 2 for 128 bit operands.

k) It may be advantageous to replace this instruction by two 64-bit moves or LD-

DQU.

Floating point XMM instructions

Instruction	Operands	sdorl	Mic	Lat	Ado	Rec	Port	Exe	Sub	Inst	Notes
		38	Microcode	Latency	Additional latency	Reciprocal through-	4	Execution unit	Subunit	Instruction set	tes
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	1	mmx	shift	sse	
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

Prescott

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	a
CVT(T)SS2SI	r32,xmm	2	0	17	1	4	1	fp		sse	a
								-			
Arithmetic											
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	а
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	a
CMPccPS/D	,			_							
CMPccSS/D	r,r/m	1	0	5	1	2	1	fp	add	sse	a
COMISS/D UCOMISS/D	r,r/m	2	0	6	1	3	1	fp	add	sse	a
Logic											
ANDPS/D ANDNPS/D											
ORPS/D XORPS/D	r,r/m	1	0	2	1	2	1	mmx	alu	sse	a
	.,			_	•	_			_ u.u		
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, (x)mm = mmx or xmm register, sr = segment register, m =

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

μops: 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

			7 ((011)			
XLAT		3		6	6	
PUSH	r	1	np	1	1	
PUSH	i i	1	np		1	
PUSH	m '	2	116		5	
PUSH		3			6	
	sr					
PUSHF(D/Q)		14			12	Nation (C4 manda
PUSHA(D)		9			11	Not in x64 mode
POP	r	1	np	1	1	
POP	(E/R)SP	1	np	1	1	
POP	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	ALOU/ I	7	5	Not in x64 mode
SALC				'	5	
		_	A 01.14		_	4 clock latency
LEA	r,m	1	AGU1	1-4	1	on input register
BSWAP	r	1	ALU0	1	1	
LDS LES LFS LGS LSS	m	10		30	30	
PREFETCHNTA	m	1	Mem		1	
PREFETCHT0/1/2	m	1	Mem		1	
LFENCE		1			1/2	
MFENCE		1			1	
SFENCE						
SPENCE		1			1	
Arithmetic instructions						
ADD SUB	r,r/i	1	ALU0/1	1	1/2	
ADD SUB	r,m	1	ALU0/1, Mer		1	
ADD SUB	· ·	1	ALOU/ I, IVICI	2		
	m,r/i				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	m,r/i	1			1	
INC DEC NEG NOT	r	1	ALU0/1	1	1/2	
INC DEC NEG NOT	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	
			1			
IMUL	r16,r16	2	ALU0, 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		
MUL IMUL	m32	4	ALU0, Mul	7		
MUL IMUL	m64	8	ALU0, Mul	14		
DIV	r/m8	9	ALU0, Div	22	22	
DIV	r/m16	12	ALU0, Div	33	33	
DIV	r/m32	12	ALU0, Div	49	49	
DIV	r/m 64	38	ALU0, Div	183	183	
IDIV	r/m8	26	ALU0, Div	38	38	
IDIV	r/m16	29	ALU0, Div	45	45	
IDIV	r/m32	29	ALU0, Div	61	61	
IDIV	r/m64	60	ALU0, Div	207	207	
CBW	1/11104	2	ALU0	5	207	
CWDE		1	ALU0	1		
CDQE		1	ALU0	1		
CWD				5		
		2	ALU0			
CDQ		1	ALU0	1		
CQO		1	ALU0	1		
Logic instructions						
AND OR XOR	r,r/i	1	ALU0/1	1	1/2	
AND OR XOR	r,m	1	ALU0/1, Mer		1	
AND OR XOR	m,r/i	1	ALU0/1, Me	1	1	
TEST	r,r/i	1	ALU0/1	1	1/2	
TEST	m,r/i	1	ALU0/1, Mer		1	
SHR SHL SAR	r,i/cl	1	ALU0	1	1	
SHR SHL SAR	m,i/cl	1	ALU0	1	1	
ROR ROL	r,i/cl	1	ALU0	1	1	
ROR ROL	m,i/cl	1	ALU0	1	1	
RCR	r,1	5	ALU0	7		
RCL	r,1	2	ALU0	1		
RCR	r/m,i/cl	12-17	ALU0	12-15		
RCL	r/m,i/cl	14-20	ALU0	14-18		
SHLD	r16,r16,i	10	ALU0	14-18		1-2 more if mem
SHLD	r32,r32,i	2	ALU0	5		1-2 more if mem
				11		1-2 more if mem
SHLD	r64,r64,i	10	ALU0			
SHLD	r16,r16,cl	9	ALU0	9		1-2 more if mem
SHLD	r32,r32,cl	2	ALU0	5		1-2 more if mem
SHLD	r64,r64,cl	9	ALU0	10		1-2 more if mem
SHRD	r16,r16,i	8	ALU0	8		1-2 more if mem
SHRD	r32,r32,i	2	ALU0	5		1-2 more if mem
SHRD	r64,r64,i	10	ALU0	9		1-2 more if mem
SHRD	r16,r16,cl	7	ALU0	8		1-2 more if mem
SHRD	r32,r32,cl	2	ALU0	5		1-2 more if mem
SHRD	r64,r64,cl	9	ALU0	9		1-2 more if mem
ВТ	r,r/i	1	ALU1	1	1	
ВТ	m,r	9		10		
ВТ	m,i	2		5		

BTR BTS BTC	r,r/i	1	ALU1	1 1	1	
BTR BTS BTC	m,r	10	ALU1	11		
BTR BTS BTC	m,i	3	ALU1	6		
BSF BSR	r,r/m	10		16		
SETcc	r	1	ALU0+1	2	2	
SETcc	m	2			5	
CLC STC		1	ALU0/1		1/2	
CMC		1	, 1200, 1	2	2	
CLD		5		_	7	
STD		6			25	
Control transfer instruction	ne					
JMP	short/near	1	ALU1		2	
JMP	far	29	ALOT		66	Not in x64 mode
						NOT III X04 IIIOUE
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	
LOOP	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		
		4n+11				factoot for bigh n
REP MOVS		3		2n - 4n		fastest for high n
SCAS				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

Floating point x87 inst						
	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
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		1			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	
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		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 FPREM	1 26		1 ~110	1	
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	
WAIT	2		5	5	
FNCLEX	4			26	
FNINIT	23		74		

Integer MMX and XMM instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVD	r32/64,(x)mm	1		4	2	
MOVD	m32/64,(x)mm	1	Mem	5	1 1	
MOVD	(x)mm,r32/64	1		3	1 1	
MOVD	(x)mm,m32/64	1	Mem	4	1	
MOVQ	(x)mm, (x)mm	1	FP0/1	1	1/2	
MOVQ	(x)mm,m64	1	Mem	4	1 1	
MOVQ	m64, (x)mm	1	Mem	5	1 1	
MOVDQA	xmm, xmm	1	FP0/1	1	1/2	
MOVDQA	xmm, m128	1	Mem	4	1 1	
MOVDQA	m128, xmm	1	Mem	5	1 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 1	
MOVQ2DQ	xmm,mm	1		1	1 1	
MOVNTQ	m64,mm	1	Mem	~400	1	
MOVNTDQ	m128,xmm	1	Mem	~450	3	
PACKSSWB/DW						
PACKUSWB	(x)mm, (x)mm	1	FP0	1	1 1	
PUNPCKH/LBW/WD/DQ	(x)mm, (x)mm	1	FP0	1	1 1	
PUNPCKH/LQDQ	(x)mm, (x)mm	1	FP0	1	1 1	

PSHUFB	mm mm	1	FP0	1	1]
	mm,mm	1	FPU	1 6	1	
PSHUFB	xmm,xmm	4	ED0		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,(x)mm	1		4	2	
PINSRW	(x)mm,r32,i	1		3	1	
PEXTRW	r32,(x)mm,i	2		5	5	
Arithmetic instructions						
PADD/SUB(U)(S)B/W/D	(x)mm, (x)mm	1	FP0/1	1	1/2	
PADDQ PSUBQ	(x)mm, (x)mm	2		5	5	
PHADD(S)W PHSUB(S)W	(x)mm, (x)mm	7		8	8	
PHADDD PHSUBD	(x)mm, (x)mm	3		6		
PCMPEQ/GTB/W/D	(x)mm,(x)mm	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	(x)mm,(x)mm	1	FP0/1	1	1/2	
PMIN/MAXUB	(x)mm,(x)mm	1	FP0/1	1	1/2	
PMIN/MAXSW	(x)mm,(x)mm	1	FP0/1	1	1/2	
PABSB PABSW PABSD	(x)mm,(x)mm	1	FP0/1	1	1/2	
PSIGNB PSIGNW PSIGND	(x)::::::,(x)::::::		11 0/1		.,_	
I GIGIND I GIGINV I GIGIND	(x)mm,(x)mm	1	FP0/1	1	1/2	
Logic instructions						
PAND(N) POR PXOR	(x)mm,(x)mm	1	FP0/1	1	1/2	
PSLL/RL/RAW/D/Q	(x)mm,(x)mm	2	FP0	5	5	
PSLL/RL/RAW/D/Q	(x)xmm,i	1	FP0	1	1	
PSLL/RLDQ	xmm,i	1	FP0	1	1	
. JELINED &	7311111,1	•	110	•	ľ	
Other						
EMMS		9			9	

Floating point XMM instructions

Operands	μops	Unit	Latency	Reciprocal	Remarks
				throughput	

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	·	
	7,7	•		·		
Conversion						
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		3	FP1	6	6	
ADDSUBPS	xmm,xmm	ა 1	FP1	5	1	
ADDSUBPD	xmm,xmm	3	FP1	6	6	
HADDPS HSUBPS	xmm,xmm	5 5	FP0+1	8	7	
HADDPD HSUBPD	xmm,xmm	5 5	FP0+1	8	7	
	xmm,xmm			4		
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	
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, (x)mm = mmx or xmm register, m = memory, m32 = 32-bit memory

operand, etc.

μops: 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			0	VOITION			
PUSH	PUSH	r	1	IP0+1		1	
PUSH							
PUSHA(D)							
PUSHA(D)							
POP				" 0 1			Not in v64 mode
POP		r					NOT III XO4 IIIOGE
POP	I						
POPF(D/Q)	I						
POPA(D)		m					
LAHF SAHF SAHF SAHF SAHF SAHF SAHC 2	, ,						
SAHF SALC	1 , ,		17			14	Not in x64 mode
SALC			1			1	
LEA	SAHF		1	IP0	2	1	
LEA	SALC		2		6	4	Not in x64 mode
LEA	LEA	r,[r+d]	1	IP0/1	1	1	
LEA	LEA	r,[r+r*s]	1	IP1	1	1	
LEA r, [r p+d] 1 IP0/1 0.5 LEA r16, [m] 2 4 4 BSWAP r 1 IP0 1 1 MOVBE r16, m16 1 2 1 MOVBE r32/64, m32/64 1 1 1 MOVBE m,r 1 1 1 PREFETCHNTA m 1 1 1 PREFETCHNTA m 1 1 1 PREFETCHNTW m 1 1 1 PREFETCHNTW m 1 1 1 MFENCE 2 8 8 MFENCE 2 14 4 SFENCE 1 1 IP0/1 1 0.5 ADD SUB r,r/i 1 IP0/1, Mem 1 1 ADD SUB r,r/i 1 IP0/1, Mem 1 1 ADC SBB r,r/i 1 IP0/1 2	LEA		1	IP0+1	2	2	
LEA	I	1 1					
BSWAP	I	1 1			4		
MOVBE r16,m16 1 1 2 1 2 2 2 1 1 1 2 2 2 <th< td=""><td></td><td>r</td><td></td><td>IPO</td><td></td><td></td><td></td></th<>		r		IPO			
MOVBE r32/64,m32/64 1 1 1 MOVBE m,r 1 1 1 PREFETCHNTA m 1 1 1 PREFETCHTO/1/2 m 1 1 1 PREFETCHNTW m 1 1 1 LFENCE 2 8 8 MFENCE 2 8 8 MFENCE 2 14 5 SFENCE 1 1 1 1 ADD SUB r,r/i 1 1PO/1 1 0.5 ADD SUB m,r/i 1 1PO/1 1 0.5 ADC SBB r,r/i 1 1PO/1 2 2 ADC SBB r,m 1 1PO/1 2 2 ADCX r32,r32 1 1PO+1 2 2 ADCX r64,r64 1 1PO+1 2 2 ADOX r64,r64 1 1PO+1 1		r16 m16		"	•		
MOVBE m,r 1 2 2 2 2 2 2 2 2 2 2 </td <td></td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td>		1					
PREFETCHNTA m 1 <th< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td></th<>							
PREFETCHT0/1/2 m 1	I						
PREFETCHNTW m 1 <th< td=""><td></td><td>m</td><td></td><td></td><td></td><td></td><td></td></th<>		m					
LFENCE	I	m	1			1	
MFENCE SFENCE 2 1 14 7 Arithmetic instructions r,r/i 1 IP0/1 1 0.5 ADD SUB ADD SUB ADD SUB ADD SUB M,r/i r,m 1 IP0/1, Mem 1 1 ADC SBB ADC SBB r,r/i 1 IP0/1, Mem 6 1 2 ADC SBB r,m 1 IP0/1 2 2 2 ADC SBB m,r/i 1 IP0+1 2 2 2 ADCX r32,r32 1 IP0+1 2 2 2 ADOX r64,r64 1 IP0+1 2 2 2 ADOX r64,r64 1 IP0+1 1 0.5 6 6 CMP r,r/i 1 IP0/1 1 0.5 1 1 CMP r,r/i 1 IP0/1 1 0.5 1 1 1 1 1 1 1 1 1 1 1 1		m					
Arithmetic instructions r,r/i 1 IPO/1 1 0.5 ADD SUB r,m 1 IPO/1, Mem 1 1 ADD SUB r,m 1 IPO/1, Mem 6 1 ADD SUB m,r/i 1 IPO/1, Mem 6 1 ADC SBB m,r/i 1 IPO/1 2 2 ADC SBB m,r/i 1 6 2 ADCX r32,r32 1 IPO+1 2 2 ADCX r64,r64 1 IPO+1 6 6 2 ADOX r64,r64 1 IPO+1 6 6 6 ADOX r64,r64 1 IPO/1 1 0.5 1 CMP r,r/i 1 IPO/1 1 0.5 1 INC DEC r 1 IPO/1 1 0.5 1 INC DEC NEG NOT m 1 1 0.5 Not in x64 mode D	LFENCE					8	
Arithmetic instructions	MFENCE		2			14	
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 6 1 ADC SBB r,r/i 1 IP0/1 2 2 ADC SBB r,m 1 6 2 ADCX r32,r32 1 IP0+1 2 2 ADCX r32,r32 1 IP0+1 6 6 ADOX r32,r32 1 IP0+1 6 6 ADOX r64,r64 1 IP0+1 1 0.5 CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 INC DEC r 1 IP0/1 1 0.5	SFENCE		1			7	
ADD SUB r,m 1 IP0/1, Mem 1 1 IP0/1, Mem 6 1 ADC SBB r,r/i 1 IP0/1, Mem 6 1 IP0/1, Mem 6 1 IP0/1, Mem 6 1 IP0/1, Mem 6 1 IP0/1, Mem 6 1 IP0/1, Mem 6 1 IP0/1, Mem 6 1 IP0/1 2	Arithmetic instructions						
ADD SUB m,r/i 1 IP0/1, Mem 6 1 ADC SBB r,r/i 1 IP0/1 2 2 ADC SBB r,m 1 6 2 ADCX SBB m,r/i 1 6 2 ADCX r32,r32 1 IP0+1 2 2 ADOX r64,r64 1 IP0+1 6 6 ADOX r32,r32 1 2 2 2 ADOX r64,r64 1 IP0/1 1 0.5 1 CMP r,r/i 1 IP0/1 1 0.5 1 CMP m,r/i 1 IP0/1 1 0.5 1 LINC DEC r 1 IP0/1 1 0.5 1 INC DEC NEG NOT m 1 6 1 1 Not in x64 mode AAS 13 12 Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode	ADD SUB	r,r/i	1	IP0/1	1	0.5	
ADC SBB r,r/i 1 IP0/1 2 2 ADC SBB r,m 1 6 2 ADCX 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 2 ADOX r64,r64 1 6 6 6 6 CMP r,r/i 1 IP0/1 1 0.5 1 1 CMP m,r/i 1 IP0/1 1 0.5 1 </td <td>ADD SUB</td> <td>r,m</td> <td>1</td> <td>IP0/1, Mem</td> <td></td> <td>1</td> <td></td>	ADD SUB	r,m	1	IP0/1, Mem		1	
ADC SBB r,r/i 1 IP0/1 2 2 ADC SBB r,m 1 6 2 ADCX 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 2 ADOX r64,r64 1 6 6 6 6 CMP r,r/i 1 IP0/1 1 0.5 1 1 CMP m,r/i 1 IP0/1 1 0.5 1 </td <td>ADD SUB</td> <td>m,r/i</td> <td>1</td> <td>IP0/1, Mem</td> <td>6</td> <td>1</td> <td></td>	ADD SUB	m,r/i	1	IP0/1, Mem	6	1	
ADC SBB r,m 1 6 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 r64,r64 1 6 6 6 CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 LEG NOT r 1 IP0/1 1 0.5 INC DEC NEG NOT m 1 6 1 AAA 13 12 Not in x64 mode DAA 20 16 Not in x64 mode <	ADC SBB		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 2 ADOX r64,r64 1 6 6 6 CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 IP0/1 1 0.5 latency to flag=2 NEG NOT r 1 IP0/1 1 0.5 latency to flag=2 NEG NOT m 1 6 1 Not in x64 mode Not in x64 mode AAS 13 12 Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode Not	I	1	1			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 IP0/1 1 0.5 latency to flag=2 NEG NOT r 1 IP0/1 1 0.5 latency to flag=2 NEG NOT m 1 IP0/1 1 0.5 latency to flag=2 NEG NOT m 1 6 1 Not in x64 mode AAA 13 12 Not in x64 mode Not in x64 mode AAS 20 16 Not in x64 mode DAS 21 16 Not in x64 mode AAD 4 5 Not in x64 mode MUL IMUL r8 3 IP0 5 5					6		
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 IP0/1 1 0.5 latency to flag=2 NEG NOT r 1 IP0/1 1 0.5 latency to flag=2 NEG NOT m 1 6 1 0.5 latency to flag=2 NEG NOT m 1 1 0.5 latency to flag=2 NEG NOT m 1 6 1 Not in x64 mode AAA 13 12 Not in x64 mode Not in x64 mode DAA 20 16 Not in x64 mode Not in x64 mode AAD 4 5 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 <td< td=""><td></td><td></td><td></td><td>IP0+1</td><td></td><td></td><td></td></td<>				IP0+1			
ADOX ADOX ADOX ADOX CMP CMP CMP CMP CMP T,r/i INC DEC R R INC DEC R INC DEC NEG NOT AAA AAS AAS DAA AAS DAA AAS DAA AAD AAD	I						
ADOX r64,r64 1 6 6 CMP r,r/i 1 IP0/1 1 0.5 CMP m,r/i 1 1 0.5 latency to flag=2 INC DEC r 1 IP0/1 1 0.5 latency to flag=2 INC DEC NEG NOT m 1 6 1 1 Not in x64 mode AAA 13 12 Not in x64 mode	I			" 0 1			
CMP r,r/i 1 IP0/1 1 0.5 1 INC DEC r 1 IP0/1 1 0.5 latency to flag=2 NEG NOT r 1 IP0/1 1 0.5 latency to flag=2 INC DEC NEG NOT m 1 6 1 Not in x64 mode AAA 13 12 Not in x64 mode Not in x64 mode AAS 13 12 Not in x64 mode Not in x64 mode DAA 20 16 Not in x64 mode Not in x64 mode AAD 4 5 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5	I						
CMP m,r/i 1 IP0/1 1 0.5 latency to flag=2 NEG NOT r 1 IP0/1 1 0.5 latency to flag=2 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 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5		1 '		IDO/1			
INC DEC r 1 IP0/1 1 0.5 latency to flag=2 NEG NOT r 1 IP0/1 1 0.5 latency to flag=2 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 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5				150/1	ı		
NEG NOT r 1 IPO/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 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IPO 5 5	I			ID0/4	4		latara av. ta fla av. O
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 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5	I						latency to flag=2
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 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5				IP0/1			
AAS 13 12 Not in x64 mode DAA 20 16 Not in x64 mode DAS 21 16 Not in x64 mode AAD 4 5 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5	I	m				1	
DAA 20 16 Not in x64 mode DAS 21 16 Not in x64 mode AAD 4 5 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5							
DAS 21 16 Not in x64 mode AAD 4 5 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5							
AAD 4 5 Not in x64 mode AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5	I						
AAM 11 24 16 Not in x64 mode MUL IMUL r8 3 IP0 5 5	DAS		21		16		Not in x64 mode
MUL IMUL r8 3 IPO 5 5	AAD		4		5		Not in x64 mode
MUL IMUL r8 3 IPO 5 5	AAM		11		24	16	Not in x64 mode
	MUL IMUL	r8	3	IP0	5		
	MUL IMUL	r16	4	IP0	5	5	

MUL IMUL	r32	3	IP0	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	r,r,m	4	IP0		8-10	
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	
CBW	1/1110-4	2	IP0	4	47-107	
CWDE		1	IP0	1		
CDQE		1	IP0	1		
CWD		2	IP0	4		
CDQ		1	IP0	1		
CQO		1	IP0	1		
	r16 r16		IFU		4	
POPCNT	r16,r16	2		4	4	
POPCNT	r32,r32	1		3	1	
POPCNT	r64,r64	1		3	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	1	
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
SHLD	r32,r32,i	1	IP0	2		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
				1		
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	r64,r64,cl	2	IP0	4		2 more if mem
вт	r,r/i	1	IP0+1	1 1	1	
BT	m,r	7	•	9	•	
BT	m,i	1 1		1 1		
BTR BTS BTC			IP0+1	1 1	1	
1	r,r/i	1	IPU+1	I I		
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 1	IP0/1		1	
CMC		1		1 1	1	
CLD		4	IP0+1		7	
STD		5	IP0+1		35	
0.2			• .		00	
Control transfer instruction	⊓ ns					
JMP	short/near	1 1	IP1		2	
JMP	r	1			2	
JMP	m(near)	1 1			2	
Conditional jump	short/near		IP1		1-2	
- ·		1	IF I		2-15	
J(E/R)CXZ	short	2				
LOOP	short	7			10-20	
LOOP(N)E	short	8				
CALL	near	1			2	
CALL	r	1			9	
CALL	m	3			14	
RET		1 1			3	
RET	i	1 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		
		_				per byte, best
REP STOS		~0.12B		~0.1B		case
MOVS		5		6		Case
WO V S						nor byto boot
REP MOVS		~ 0.2B		~0.15B		per byte, best case
SCAS		3		5		Case
1				1		
REP SCAS		~5n		~3n		
CMPS		6		6		
REP CMPS		~6n		~3n		

Synchronization instruction	ns					
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						
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		1	1	
FCHS		1		1	1	
FCOM(P) FUCOM	r/m	1		5	1	
FCOMPP FUCOMPP		1		5	1	
FCOMI(P) FUCOMI(P)	r	1		5	1	
FIADD FISUB(R)	m	3			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		
Other						
FNOP		1			0.5	
WAIT		2			4	
FNCLEX		4			24	
FNINIT		19			65	

Integer MMX and XMM instructions

	Operands	µops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
MOVD MOVQ	r32/64,(x)mm	1		4	1	
MOVD MOVQ	m,(x)mm	1	Mem	3	1	
MOVD MOVQ	(x)mm,r32/64	1		3	1	
MOVD MOVQ	(x)mm,m	1	Mem	4	1	
MOVQ	(x)mm, (x)mm	1	FP0/1	1	0.5	
MOVDQA	x, x	1	FP0/1	1	0.5	
MOVDQA MOVDQU	x, m128	1	Mem	4	1 1	

MOVDQA MOVDQU							
MOVDQ2DQ mm, x mm 1 mg, mm, mm 1 mg, mm, mm 1 mg, mm, mm<	MOVDQA MOVDQU	m128, x	1	Mem	3	1	
MOVAZDQ	LDDQU	x, m128	1	Mem	4	1	
MOVNTDQ	MOVDQ2Q	mm, x	1		1	1	
MOVNTDQA m128,x 1 Mem ~370 1 PACKSSWB/DW x, m128 1 4 1 PACKUSWB (x)mm, (x)mm 1 FP0 1 1 PACKUSWB (x)mm, (x)mm 1 FP0 1 1 PUNPCKH/LQDQ (x)mm, (x)mm 1 FP0 1 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 FP0 1 1 PMOVSX/ZX BW BD BQ DW DQ x,x 1 FP0 1 1 PSHUFB mm,mm 1 FP0 1 1 PSHUFB x,x 4 FP0 5 5 PSHUFUHW x,x,i 1 FP0 1 1 PSHUFUHW x,x,i 1 FP0 1 1 PSHUFD x,x,i 1 FP0 1 1 PSHUFD x,x,i 1 FP0 1 1 PSHUFD x,x,i 1 FP0 1	MOVQ2DQ	x,mm	1		1	1	
MOVNTDOA	MOVNTQ	m64,mm	1	Mem	~370	1	
MOVNTDQA	MOVNTDQ	m128,x	1	Mem	~370	1	
PACKSSWB/DW PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PUNPCKH/LQDQ PUNPCKH/LQDQ PUNPCKH/LQDQ PUNPCKH/LQDQ PUNPCKH/LQDQ PMOVSXZX BW BD BQ DW DQ XX 1 FP0 1 1 PF0	MOVNTDQA		1		4	1	
PACKUSWB		,					
PACKUSDW		(x)mm. (x)mm	1	FP0	1	1	
PUNPCKH/LBW/WD/DQ	PACKUSDW	1 ' ' '	1		1	1	
PUNPCKH/LQDQ					1	1	
PMOVSX/ZX BW BD BQ DW DQ		1 ' ' ' '			1	1	
DQ		(**)***********************************	•				
PMOVSX/ZX BW BD QD W DQ		X,X	1		1	1	
DQ	PMOVSX/ZX BW BD BO DW	,					
PSHUFB		x,m	1		1	1	
PSHUFW	PSHUFB	mm,mm	1	FP0	1	1	
PSHUFL/HW	PSHUFB	X,X	4	FP0	5	5	
PSHUFD	PSHUFW	mm,mm,i	1	FP0	1	1	
PSHUFD	PSHUFL/HW		1	FP0	1	1	
PALIGNR	PSHUFD		1		1	1	
PBLENDVB x,x,xmm0 2 FP0 4 4 PBLENDVB x,m,xmm0 3 FP0 5 PBLENDW x,x/m,i 1 FP0 1 1 MASKMOVQ mm,mm 1 Mem ~370 1 MASKMOVDQU x,x 3 Mem ~370 1 MASKMOVDQU x,x 3 Mem ~370 1 MASKMOVDQU x,x 3 Mem ~370 1 MASKMOVDQU x,x 3 Mem ~370 1 MASKMOVDQU x,x 3 Mem ~370 1 MASKMOVDQU x,x 3 Mem ~370 1 PMDVMSKB r32,xi 1 3 1 1 PINSRB/DQ x,m8,i 1 1 1 1 PEXTRBW r32,xi 2 5 4 4 PEXTRD m32,xi 4 5 8			1		1	1	
PBLENDVB x,m,xmm0 3 FP0 1 1 PBLENDW x,x/m,i 1 FP0 1 1 MASKMOVQ mm,mm 1 Mem ~370 1 MASKMOVDQU x,x 3 Mem ~370 5 PMOVMSKB f32,(x)mm 1 4 1 PINSRW (x)mm,r32,i 1 3 1 PINSRB/D/Q x,r32,i 1 3 1 PINSRB/D/Q x,r32,i 1 3 1 PEXTRW r32,(x)mm,i 2 5 4 PEXTRBW r64,x,i 2 7 7 PEXTRD m32,x,i 4 5 8 PEXTRD m32,x,i 4 5 8 Arithmetic instructions (x)mm, (x)mm 2 4 4 PADD/SUB(U)(S)B/W/D (x)mm, (x)mm 3 5 PADDQ PSUBQ (x)mm, (x)mm 5 9 9 PH						4	
PBLENDW							
MASKMOVQ mm,mm 1 Mem ~370 1 MASKMOVDQU x,x 3 Mem ~370 5 PMOVMSKB r32,(x)mm 1 4 1 PINSRW (x)mm,r32,i 1 3 1 PINSRB/D/Q x,r32,i 1 3 1 PINSRB/D/Q x,m8,i 1 1 1 PEXTRW r32,(x)mm,i 2 5 4 PEXTRW r32,x,i 2 5 4 PEXTRQ r64,x,i 2 7 7 PEXTRD m8/16,x,i 5 6 PEXTRQ m64,x,i 4 8 Arithmetic instructions (x)mm, (x)mm 1 FP0/1 1 0.5 PADDQ PSUBQ (x)mm, (x)mm 2 4 4 4 PADDQ PSUBQ (x)mm, (x)mm 5 6 6 6 PHADD(s)W PHSUB(s)W x, x/m 7-8 9 9 9					1		
MASKMOVDQU x,x 3 Mem ~370 5 PMOVMSKB r32,(x)mm 1 4 1 PINSRW (x)mm,r32,i 1 3 1 PINSRB/D/Q x,r32,i 1 3 1 PINSRB/D/Q x,m8,i 1 1 1 PEXTRW r32,(x)mm,i 2 5 4 PEXTRBW r32,x,i 2 5 4 PEXTRD m8/16,x,i 5 6 6 PEXTRD m32,x,i 4 8 8 Arithmetic instructions m64,x,i 4 8 8 Arithmetic instructions (x)mm, (x)mm 1 FP0/1 1 0.5 PADDQ PSUBQ (x)mm, (x)mm 2 4 4 4 PADDQ PSUBQ (x)mm, (x)mm 5 6 6 PHADD(S)W PHSUB(S)W x, x/m 7-8 9 9 PCMPEQ/GTB/W/D x, x 2 4 4 <td< td=""><td></td><td></td><td>-</td><td></td><td>· ·</td><td>· •</td><td></td></td<>			-		· ·	· •	
PMOVMSKB						=	
PINSRW		· ·		1010111			
PINSRB/D/Q x,r32,i 1 3 1 PINSRB/D/Q x,m8,i 1 1 1 PEXTRW r32,(x)mm,i 2 5 4 PEXTRB/W r32,x,i 2 5 4 PEXTRQ r64,x,i 2 7 7 PEXTRD m8/16,x,i 5 6 PEXTRQ m8/16,x,i 4 5 PEXTRQ m8/16,x,i 4 5 PEXTRQ m8/16,x,i 4 5 PEXTRQ m8/16,x,i 4 5 PEXTRQ m8/16,x,i 4 5 PEXTRQ m8/16,x,i 4 5 PEXTRQ m8/16,x,i 4 5 PEXTRQ m8/16,x,i 4 5 PEXTRQ m8/16,x,i 4 8 Arithmetic instructions (x)mm, (x)mm (x)mm, (x)mm 2 1 1 0.5 PADD/SUBQ(U)(S)B/W/D (x)mm, (x)mm 2 4 4 4 PAD		1 ' '				-	
PINSRB/D/Q		' '	-			•	
PEXTRW r32,(x)mm,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 6 PEXTRD m32,x,i 4 5 5 PEXTRQ m64,x,i 4 5 6 PEXTRD m32,x,i 4 5 6 PEXTRD m32,x,i 4 5 6 PEXTRD m32,x,i 4 5 6 PEXTRD m32,x,i 4 5 6 PEXTRD m32,x,i 4 4 8 **PADD/SUB(U)(S)B/W/D (x)mm, (x)mm 1 **PADDQ PSUBQ (x)mm, (x)mm 2 (x)mm, (x)mm 3 **TRAIN PROVIDE PROVIDE PROVIDE PROVIDE PROVID PROVIDE						-	
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 (x)mm, (x)mm 1 FP0/1 1 0.5 PADD/SUB(U)(S)B/W/D (x)mm, (x)mm 2 4 4 4 PADDQ PSUBQ (x)mm, (x)mm 2 4 4 4 PADDQ PSUBQ (x)mm, (x)mm 3 5 9 9 9 PHADD(S)W PHSUB(S)W mm, mm 5 6 6 6 6 6 6 6 6 9 <td></td> <td></td> <td></td> <td></td> <td>5</td> <td></td> <td></td>					5		
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 PADDQ PSUBQ PADDQ PSUBQ PADDQ PSUBQ PADDQ PSUBQ PADDQ PSUBQ PHADD(S)W PHSUB(S)W PHADD(S)W PHSUB(S)W PHADD(S)W PHSUB(S)W PHADDD PHSUBD PHADDD PHSUBD PHADDD PHSUBD PCMPEQ/GTB/W/D PCMPEQ/GTB/W/D PCMPEQQ PCMPGTQ PCMPGTQ PCMPGTQ PCMPGTQ PMULL/HW PMULHUW PMULL/HW PMULHUW PMULL/HW PMULHUW PMULL/HW PMULHUW PMULL/HW PMULHUW PMULL/HW PMULHUW PMULL/HW PMULHUW PMULL/HW PMULHUW PMULHRSW PMULHRSW PMULHRSW PMULHRSW PMULHRSW PMULHRSW PMULHRSW T T T T T T T T T T T T T T T T T T T		1 ' '				-	
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 (x)mm, (x)mm 1 FP0/1 1 0.5 PADDQ PSUBQ (x)mm, (x)mm 2 4 4 4 PADDQ PSUBQ (x)mm, (x)mm 3 5 5 5 PHADD(S)W PHSUB(S)W N, x/m 7-8 9 9 9 PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 PCMPEQ/GTB/W/D (x)mm,(x)mm 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 5 2 PMULHRSW mm,mm 1 FP0 5 2 PMULHRSW x, x 1							
PEXTRD					'		
Arithmetic instructions PADD/SUB(U)(S)B/W/D (x)mm, (x)mm 1 FP0/1 1 0.5 PADDQ PSUBQ (x)mm, (x)mm 2 4 4 PADDQ PSUBQ (x)mm, m 3 5 PHADD(S)W PHSUB(S)W mm, mm 5 6 6 PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 PCMPEQ/GTB/W/D (x)mm,(x)mm 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 5 2 PMULL/HW PMULHUW x, x 1 FP0 5 2 PMULHRSW mm,mm 1 FP0 5 2 PMULHRSW x, x 1 FP0 5 2			•				
Arithmetic instructions PADD/SUB(U)(S)B/W/D (x)mm, (x)mm 1 FP0/1 1 0.5 PADDQ PSUBQ (x)mm, (x)mm 2 4 4 PADDQ PSUBQ (x)mm, m 3 5 PHADD(S)W PHSUB(S)W mm, mm 5 6 6 PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 PCMPEQ/GTB/W/D (x)mm,(x)mm 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 5 2 PMULL/HW PMULHUW x, x 1 FP0 5 2 PMULHRSW mm,mm 1 FP0 5 2 PMULHRSW x, x 1 FP0 5 2							
PADD/SUB(U)(S)B/W/D (x)mm, (x)mm 1 FP0/1 1 0.5 PADDQ PSUBQ (x)mm, (x)mm 2 4 4 PADDQ PSUBQ (x)mm, m 3 5 PHADD(S)W PHSUB(S)W mm, mm 5 6 6 PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 PCMPEQ/GTB/W/D (x)mm,(x)mm 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	PEXTRQ	11104,8,1	4			0	
PADD/SUB(U)(S)B/W/D (x)mm, (x)mm 1 FP0/1 1 0.5 PADDQ PSUBQ (x)mm, (x)mm 2 4 4 PADDQ PSUBQ (x)mm, m 3 5 PHADD(S)W PHSUB(S)W mm, mm 5 6 6 PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 PCMPEQ/GTB/W/D (x)mm,(x)mm 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	A						
PADDQ PSUBQ (x)mm, (x)mm 2 4 4 PADDQ PSUBQ (x)mm, m 3 5 PHADD(S)W PHSUB(S)W mm, mm 5 6 6 PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 PCMPEQ/GTB/W/D (x)mm, (x)mm 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 PMULHRSW mm,mm 1 FP0 4 1 PMULHRSW x, x 1 FP0 5 2		()() ==================================	4	ED0/4	4	0.5	
PADDQ PSUBQ (x)mm, m 3 PHADD(S)W PHSUB(S)W mm, mm 5 PHADD(S)W PHSUB(S)W x, x/m 7-8 PHADDD PHSUBD (x)mm, (x)mm 3-4 PCMPEQ/GTB/W/D (x)mm,(x)mm 1 PCMPEQQ x, x 2 PCMPGTQ x, x 1 PMULL/HW PMULHUW mm,mm 1 PMULL/HW PMULHUW x, x 1 PMULL/HW PMULHUW x, x 1 PMULHRSW mm,mm 1 PMULHRSW x, x 1 FPO 5 2 PMULHRSW x, x 1 FPO 5 2	` ,` ,	1 ' ' ' '		FPU/I			
PHADD(S)W PHSUB(S)W mm, mm 5 6 6 9 9 PHADD(S)W PHSUB(S)W x, x/m 7-8 9 9 9 PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 5-6 PCMPEQ/GTB/W/D (x)mm,(x)mm 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 PMULHRSW mm,mm 1 FP0 4 1 PMULHRSW x, x 1 FP0 5 2		1 ' ' ' '			4		
PHADD(S)W PHSUB(S)W x, x/m 7-8 9 9 PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 PCMPEQ/GTB/W/D (x)mm,(x)mm 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 PMULHRSW mm,mm 1 FP0 4 1 PMULHRSW x, x 1 FP0 5 2 PMULHRSW x, x 1 FP0 5 2		` '			_		
PHADDD PHSUBD (x)mm, (x)mm 3-4 5-6 5-6 PCMPEQ/GTB/W/D (x)mm, (x)mm 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	, , , , , , , , , , , , , , , , , , , ,	· ·					
PCMPEQ/GTB/W/D (x)mm,(x)mm 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	, , , , , , , , , , , , , , , , , , , ,						
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 PMULHRSW mm,mm 1 FP0 5 2 PMULHRSW mm,mm 1 FP0 4 1 PMULHRSW x, x 1 FP0 5 2		1 ' ' ' '		ED0/4			
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		' ' '		FP0/1			
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		· .					+1 if mem
PMULL/HW PMULHUW x, x 1 FP0 5 2 PMULHRSW mm,mm 1 FP0 4 1 PMULHRSW x, x 1 FP0 5 2			-				
PMULHRSW mm,mm 1 FP0 4 1 PMULHRSW x, x 1 FP0 5 2			-				
PMULHRSW x, x 1 FP0 5 2							
			-			=	
PMULLD							
	PMULLD	X, X	7	FP0	11	11	+1 it mem

I=- · · · · = =	1				1 -	I
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	
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	(x)mm,(x)mm	1	FP0/1	1	0.5	
PMIN/MAXUB	(x)mm,(x)mm	1	FP0/1	1	0.5	
PMIN/MAXSW	(x)mm,(x)mm	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	(x)mm,(x)mm	1	FP0/1	1	0.5-1	
PSIGNB PSIGNW PSIGND						
	(x)mm,(x)mm	1	FP0/1	1	0.5-1	
Logic instructions						
PAND(N) POR PXOR	(x)mm,(x)mm	1	FP0/1	1	0.5	
PTEST	x,x	1		1	1	
PSLL/RL/RAW/D/Q	(x)mm,(x)mm	2	FP0	2	2	
PSLL/RL/RAW/D/Q	(x)mm,i	1	FP0	1	1	
PSLL/RLDQ	x,i	1	FP0	1	1	
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	1					
PCLMULQDQ	x,x,i	8	FP0	10	10	+1 if mem
Other						
EMMS		9			10	

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 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 1	

MOVSS/D MOVLPS/D MOVLPS/D MOVHPS/D MOVLPS/D MOVHPS/D MOVLPS/D MOVHPS/D MOVLPS/D MOVHPS/D MOVLPS/D MOVHPS/D MOVLPS/D MOVHPS/D MOVLPS/D MOVHPS/D							
MOVHPS/ID MOVLPS/ID m64,x	MOVSS/D	m32/64,x	1	Mem	3	1	
MOVHPS/ID MOVLPS/ID m64,x	MOVHPS/D MOVLPS/D	x.m64	1	Mem	4	1	
MOVLHPS X,x	1		1			1	
BLENDPS/PD	1					-	
BLENDVPS/PD				110	· ·		
SLENDVPS/PD				ED0 : 4			
INSERTPS							
INSERTPS	BLENDVPS/PD	x,m,xmm0	3	FP0+1	5	5	
EXTRACTPS r32,xi 2	INSERTPS	x,x,i	1		1	1	
EXTRACTES m32,xi	INSERTPS	x,m32,i	3		5	5	
EXTRACTES m32,xi	EXTRACTPS	r32.x.i	2			4	
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 MOVSH/LDUP x,x 1 FP0 1 1 UNPCKH/LPS x,x 1 FP0 1 1 UNPCKH/LPD x,x 1 FP0 1 1 UNPCKH/LPD x,x 1 FP0 1 1 UNPCKH/LPD x,x 1 FP0 1 1 UNPCKH/LPD x,x 1 FP0 5 2 CVTS2SS x,x 1 FP0 5 2 CVTS2SS x,x 1 FP0 5 2 <t< td=""><td></td><td></td><td></td><td></td><td>4</td><td></td><td></td></t<>					4		
MOVNTPS/D				EDN			
SHUFPS						-	
SHUFPD							
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 COMMONIAN CONTROLOR COVTED2PS X, X 1 FP0 5 2 CVTSD2SS X, X 1 FP0 4 2 CVTSD2PD X, X 1 FP0 5 2 CVTS2SDD X, X 1 FP0 4 2 CVTDQ2PS X, X 1 FP0 5 2 CVTTQ2PS X, X 1 FP0 5 2 CVTTQ2PS X, X 1 FP0 5 2 CVT(T)PS2DQ X, X 1 FP0 5 2 CVT(T)PD2DQ X, X 1 FP0 5 2 CVT(T)PD2PI mm,x							
MOVSH/LDUP		x,x,i	1		1	1	
UNPCKH/LPS UNPCKH/LPD x, x 1 FP0 1 1 Conversion CVTPD2PS CVTSD2SS x, x 1 FP0 5 2 CVTSS2SD CVTSS2SD x, x 1 FP0 5 2 CVTDQ2PS CVTDQ2PS x, x 1 FP0 5 2 CVTDQ2PS x, x 1 FP0 5 2 CVTDQ2PS x, x 1 FP0 5 2 CVT(T)PS2DQ x, x 1 FP0 5 2 CVT(T)PS2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PS2PI mm, x 1 FP0 4 2 CVT(T)PS2PI mm, x 1 FP0 5 2 CVT(T)PD2PI mm, x 1 FP0 5 2 CVT(T)PD2PI cVTP12PD x,mm 1 FP0 5 2 CVT(T)PD2PI mm, x 1 FP0 5 2 CVT(T)PD2PI cVTS12SS x, x32 1 FP0 5 2 CVT(T)S2SI cVT(T)S2SI cVT(T)S2SI cVT(T)SD2SI r32, x 1 FP0 5 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 4 2 ADDSUBPD x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 3 1 ADDPS SUBPD x, x 1 FP1 4 2 ADDSUBPD x, x 1 FP1 4 2 ADDSUBPD x, x 1 FP1 4 2 ADDSUBPD x, x 1 FP1 4 2 ADDSUBPD x, x 1 FP1 4 1 ADDPS HSUBPS x, x 1 FP1 4 2 ADDSUBPD x, x 1 FP1 4 1 ADDPS HSUBPS x, x 1 FP1 4 2 ADDSUBPD x, x 1 FP1 4 1 ADDSUBPD x, x 1 FP1 4 2 ADDSUBPD x, x 1 FP1 4 1 ADDSUBPD x, x 1 FP1 4 1 ADDPS HSUBPD x, x 1 FP1 4 1 ADDPS HSUBPD x, x 1 FP1 4 1 ADDPS HSUBPD 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 7 4 DIVSS	MOVDDUP	X, X	1	FP0	1	1	
UNPCKH/LPD	MOVSH/LDUP	x, x	1	FP0	1	1	
UNPCKH/LPD	UNPCKH/LPS	x, x	1	FP0	1	1	
Conversion x, x 1 FP0 5 2 CVTPD2PS x, x 1 FP0 4 2 CVTDS2SS x, x 1 FP0 4 2 CVTPS2PD x, x 1 FP0 5 2 CVTDQ2PS x, x 1 FP0 5 2 CVT(T)PS2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2PQ x, x 1 FP0 5 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)PS2SIS r32,x 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 <t< td=""><td>UNPCKH/LPD</td><td></td><td>1</td><td></td><td>1</td><td>1</td><td></td></t<>	UNPCKH/LPD		1		1	1	
CVTPD2PS x, x 1 FP0 5 2 CVTSD2SS x, x 1 FP0 4 2 CVTPS2PD x, x 1 FP0 5 2 CVTS2SDD x, x 1 FP0 5 2 CVTCDQ2PS x, x 1 FP0 5 2 CVTT(T) PS2DQ x, x 1 FP0 5 2 CVT(T) PS2DQ x, x 1 FP0 5 2 CVT(T) PS2DQ x, x 1 FP0 5 2 CVT(T) PD2DQ x, x 1 FP0 5 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)S2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 <		7, 7					
CVTPD2PS x, x 1 FP0 5 2 CVTSD2SS x, x 1 FP0 4 2 CVTPS2PD x, x 1 FP0 5 2 CVTS2SDD x, x 1 FP0 5 2 CVTDQ2PS x, x 1 FP0 5 2 CVTT(T) PS2DQ x, x 1 FP0 5 2 CVTT(T) PS2DQ x, x 1 FP0 5 2 CVTT(T) PS2DQ x, x 1 FP0 5 2 CVT(T) PD2DQ x, x 1 FP0 5 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)S2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5	Conversion						
CVTSD2SS x, x 1 FP0 4 2 CVTPS2PD x, x 1 FP0 5 2 CVTS2SD x, x 1 FP0 4 2 CVTDQ2PS x, x 1 FP0 5 2 CVT(T)PS2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2PQ x, mm 1 FP0 4 2 CVT(T)PS2PI mm,x 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 1 Arithmetic Anithmetic Anithmeti			1	EDO	5	2	
CVTPS2PD x, x 1 FP0 5 2 CVTSS2SD x, x 1 FP0 4 2 CVTDQ2PS x, x 1 FP0 5 2 CVTTQ2PD x, x 1 FP0 5 2 CVTQ2PD x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PD2PI mm, x 1 FP0 4 2 CVT(T)PD2PI mm, x 1 FP0 5 2 CVT(T)PD2PI mm, x 1 FP0 5 2 CVT(T)S2SS x,r32 1 FP0 5 2 CVT(T)S2SI r32,x 3 FP0 5 1 Arithmetic ADDSS SUBSS x, x 1 FP1 3 1 ADDSUBPD x, x 1 <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>							
CVTSS2SD x, x 1 FP0 4 2 CVTQQPS x, x 1 FP0 5 2 CVT(T) PS2DQ x, x 1 FP0 5 2 CVTTQ2PD x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)P2PS x,mm 1 FP0 4 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 1 CVT(T)SD2SI r32,x 3 FP0 5 1 ADDSS SUBSS x, x 1 FP1 3 1 ADDSD SUBPD x, x 1 FP1 3							
CVTDQ2PS x, x 1 FP0 5 2 CVT(T) PS2DQ x, x 1 FP0 5 2 CVTOQ2PD x, x 1 FP0 5 2 CVT(T)PD2DQ x, x 1 FP0 5 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVT(T)PS2PI mm,x 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)S2SS x,r32 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 1 Arithmetic ADDS SUBSD x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDSUBPD x, x 1 FP1 3 1 ADDSUBPS x, x			-				
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 CVT(T)PS2PI x,mm 1 FP0 4 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)S2SS x,r32 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 1 Arithmetic ADDSS SUBSS x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDSUBPS x, x 1 FP1 3 1 ADDSUBPS x, x	1	X, X					
CVTDQ2PD	1	X, X			1		
CVT(T)PD2DQ x, x 1 FP0 5 2 CVTPI2PS x,mm 1 FP0 4 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVTP12PD x,mm 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVT(T)S2SS x,r32 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 1 CVT(T)SD2SI r32,x 3 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 1 Arithmetic ADDSUBSS x, x 1 FP1 3 1 ADDSUBSD x, x 1 FP1 3 1 ADDSUBPD x, x 1 FP1 4 2 ADDSUBPD x, x 1 </td <td>CVT(T) PS2DQ</td> <td>X, X</td> <td>1</td> <td>FP0</td> <td>5</td> <td></td> <td></td>	CVT(T) PS2DQ	X, X	1	FP0	5		
CVTPI2PS x,mm 1 FP0 4 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVTPI2PD x,mm 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVTSI2SS x,r32 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 1 Arithmetic ADDSS SUBSS x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDPD SUBPD x, x 1 FP1 3 1 ADDSUBPS x, x 1 FP1 4 2 HADDPS HSUBPS x, x	CVTDQ2PD	x, x	1	FP0	5	2	
CVTPI2PS x,mm 1 FP0 4 2 CVT(T)PS2PI mm,x 1 FP0 4 2 CVTPI2PD x,mm 1 FP0 5 2 CVT(T)PD2PI mm,x 1 FP0 5 2 CVTSI2SS x,r32 1 FP0 5 2 CVT(T)SS2SI r32,x 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 1 Arithmetic ADDSS SUBSS x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDPD SUBPD x, x 1 FP1 3 1 ADDSUBPS x, x 1 FP1 4 2 HADDPS HSUBPS x, x	CVT(T)PD2DQ	x, x	1	FP0	5	2	
CVT(T)PS2PI mm,x 1 FP0 4 2 CVTPI2PD x,mm 1 FP0 5 2 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 x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDSUBPD x, x 1 FP1 3 1 ADDSUBPS x, x 1 FP1 4 2 HADDPS HSUBPS x, x	1 ' '		1		4		
CVTPI2PD x,mm 1 FP0 5 2 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 x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDSUBPD x, x 1 FP1 4 2 ADDSUBPS x, x 1 FP1 4 2 HADDPS HSUBPS x, x 1 FP1 4 2 HADDPS HSUBPD x, x 4 6 6 +1 if mem HADDPS HSUBPD x, x <td></td> <td></td> <td>-</td> <td></td> <td></td> <td></td> <td></td>			-				
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 x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDSD SUBPS x, x 1 FP1 3 1 ADDPD SUBPD x, x 1 FP1 4 2 ADDSUBPS x, x 1 FP1 3 1 ADDSUBPS x, x 1 FP1 4 2 HADDPS HSUBPS x, x 4 6 6 +1 if mem HADDPD HSUBPD x, x 1 FP0 4 1 MULSS x, x	1 ' '		-				
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 X, X 1 FP1 3 1 ADDSD SUBSD X, X 1 FP1 3 1 ADDPS SUBPS X, X 1 FP1 3 1 ADDPD SUBPD X, X 1 FP1 3 1 ADDSUBPS X, X 1 FP1 4 2 ADDSUBPS 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 5 2 MULPS X, X			-				
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 x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDSD SUBPS 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 1 FP1 4 2 HADDPD HSUBPD x, x 4 6 6 +1 if mem MULSS x, x 1 FP0 4 1 MULPS x, x 1 FP0 5 2 MULPD x, x 1<	1 ' '	· ·					
CVTSI2SD xm,r32 1 FP0 5 2 CVT(T)SD2SI r32,x 3 FP0 5 1 Arithmetic X X 1 FP1 3 1 ADDSD SUBSD X, X 1 FP1 3 1 ADDSD SUBPS 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 1 FP1 4 2 HADDPD HSUBPD X, X 4 6 6 +1 if mem MULSS X, X 1 FP0 4 1 MULPS X, X 1 FP0 5 2 MULPD X, X 1 FP0 7 4 DIVSS X, X 1 FP0 19							
Arithmetic X, X 1 FP0 5 1 ADDSS SUBSS X, X 1 FP1 3 1 ADDSD SUBSD X, X 1 FP1 3 1 ADDPS SUBPS 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 1 FP1 4 2 HADDPD HSUBPD X, X 4 6 6 +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 <td></td> <td></td> <td>1</td> <td></td> <td></td> <td></td> <td></td>			1				
Arithmetic x, x 1 FP1 3 1 ADDSS SUBSS x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS x, x 1 FP1 3 1 ADDSUBPD SUBPD x, x 1 FP1 4 2 ADDSUBPD NSUBPS 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 HI if mem MULSD x, x 1 FP0 5 2 2 MULPS x, x 1 FP0 5 2 MULPD x, x 1 FP0 7 4 DIVSS x, x 1 FP0 19 17	CVTSI2SD	xm,r32		FP0	1	2	
ADDSS SUBSS x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS 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 7 4 MULPD x, x 1 FP0 7 4 DIVSS x, x 1 FP0 19 17	CVT(T)SD2SI	r32,x	3	FP0	5	1	
ADDSS SUBSS x, x 1 FP1 3 1 ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS 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 7 4 MULPD x, x 1 FP0 7 4 DIVSS x, x 1 FP0 19 17							
ADDSD SUBSD							
ADDSD SUBSD x, x 1 FP1 3 1 ADDPS SUBPS 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 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 7 4 DIVSS x, x 1 FP0 7 4 DIVSS x, x 1 FP0 19 17	ADDSS SUBSS	x, x	1	FP1	3	1	
ADDPS SUBPS 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	ADDSD SUBSD		1	FP1		1	
ADDPD SUBPD	1				1	1	
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	1						
ADDSUBPD	1						
HADDPS HSUBPS x, x 4 6 6 +1 if mem HADDPD HSUBPD x, x 4 6 5 +1 if mem MULSS x, x 1 FPO 4 1 MULSD x, x 1 FPO 5 2 MULPS x, x 1 FPO 5 2 MULPD x, x 1 FPO 7 4 DIVSS x, x 1 FPO 19 17							
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	1						. 4 :6
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							
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							+1 if mem
MULPS x, x 1 FP0 5 2 MULPD x, x 1 FP0 7 4 DIVSS x, x 1 FP0 19 17		X, X					
MULPD x, x 1 FP0 7 4 DIVSS x, x 1 FP0 19 17	1	X, X			1		
DIVSS x, x 1 FP0 19 17	MULPS	x, x	1	FP0			
DIVSS x, x 1 FP0 19 17	MULPD	x, x	1	FP0	7	4	
	DIVSS		1	FP0	19	17	
	1						

DIVPS	x, x	6	FP0	39	39	
DIVPD	x, x	6	FP0	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 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, (x)mm = 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.

μops: 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	
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	

		3	3			
MOVSX MOVZX MOVSXD	r16,r8	2	IP0	7	7	
MOVSX MOVZX MOVSXD	r16,m8	3	IP0	7	8	
MOVSX MOVZX MOVSXD	r32/64,r	1	IP0	1	1	
MOVSX MOVZX MOVSXD	r32/64,m	1	IP0	4	1	
CMOVcc	r,r	1	IP0/1	2	1	
CMOVcc	r,m	1	11 0/1	_	1	
XCHG			IP0/1	8	8	
XCHG	r,r	3	10/1			Impuliait la ak
	r,m	3		24	24	Implicit lock
XLAT		4		8		
PUSH	r	1			1	
PUSH	i	1	IP0+1		1	
PUSH	m	3	IP0+1		8	
PUSHF(D/Q)		18	IP0+1		28	
PUSHA(D)		10			10	Not in x64 mode
POP	r	2			1	
POP	(E/R)SP	2			4	
POP	m	6			9	
POPF(D/Q)		21			48	
POPA(D)		17			15	Not in x64 mode
LAHF		1	IP0	1	1	140t III XO4 IIIOGC
SAHF		1	IP0	2	1	
SALC		-			7	Not in vC4 manda
1	[m. 1. ml]	2	ID0/4	9		Not in x64 mode
LEA	r,[r+d]	1	IP0/1	1	0.5	
LEA	r,[r+r*s]	1	IP1	1	1	
LEA	r,[r+r*s+d]	1	IP0+1	2	1	
LEA	r,[rip+d]	1	IP0/1		0.5	
LEA	r16,[m]	2		7		
BSWAP	r	1	IP0	1	1	
MOVBE	r16,m16	1			1	
MOVBE	r/m32/64	1			1	
MOVBE	m,r	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	
OI EIVOE		'			10	
Arithmetic instructions						
ADD SUB	r,r/i	1	IP0/1	1	0.5	
ADD SUB		1	IP0/1, Mem	'		
	r,m			7	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		_	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	
СМР	m,r/i	1			1	
INC DEC	r	1	IP0/1	1-2	0.5	
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
1	1	I .	1	1	1	1

		0	J			
DAA		20		17		Not in x64 mode
DAS		21		17		Not in x64 mode
AAD		4		8		Not in x64 mode
AAM		10		30	14	
	0		IDO		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	
				5	2	
IMUL	r64,r64,i	1	IP0	כ	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	13-23	
DIV	r/m 64	23	IPO, FPO	39-95	22-95	
IDIV	r/m8	26	IP0, FP0	39	20	
IDIV	r/m16	29	IP0, FP0	38-42	22	
IDIV		29				
I	r/m32		IP0, FP0	37-49	22-26	
IDIV	r/m64	44	IP0, FP0	53-108	36-107	
CBW		2	IP0	7		
CWDE		1	IP0	1		
CDQE		1	IP0	1		
CWD		2	IP0	7		
CDQ		1	IP0	1		
CQO		1	IP0	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						
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	4.0	1	
RCR	r,1	7	IP0	10	10	
RCL	r,1	1	IP0	2	2	
RCR	r,i/cl	11	IP0	13	13	

		3	3		
RCR	m,i/cl	14	IP0	13	
RCL	r,i/cl	13	IP0	13	13
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
	' '				I I
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 1
ВТ	r,r/i	1	IP0+1	1	1 1
ВТ	m,r	7		10	10
ВТ	m,i	1		1	1 1
BTR BTS BTC	r,r/i	1	IP0+1	1	1 1
BTR BTS BTC	m,r	8		11	11
BTR BTS BTC	m,i	1	IP0+1	1	1
BSF BSR	r,r/m	10	IP0/1	11	11
SETcc	r/m	1	IP0+1	2	1
CLC STC	1/111	1	IP0	_	1 1
CMC		1	IP0	1	1 1
CLD		4	IP0/1	'	8
STD		5	IP0/1		36
LZCNT	r,r/m	1	11 0/ 1	3	1
TZCNT	r,r/m	1		3	1 1
ANDN	r,r,r	1		1	0.5
		1		1	1 1
ANDN DISLDISMSK DISD	r,r,m	·		-	1 1
BLSI BLSMSK BLSR	r,r/m	1 2		1 7	7
BEXTR	r,r,r			'	
BEXTR	r,r,m	3			8
BZHI	r,r,r	1		3	1 1
PDEP	r,r,r	1		3	1
PEXT	r,r,r	1		3	1
Control transfer instruction	ns				

JMP 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 1 2 7 8 1 1 3 1 1 10 4	IP1 IP1 IP1 IP1		2 2 1-2 7-18 14-23 14-23 2 2 14 2 2 11 8	Not in x64 mode Not in x64 mode
String instructions LODS REP LODS STOS REP STOS MOVS REP MOVS SCAS REP SCAS CMPS REP CMPS		3 ~4n 2 ~0.07B 5 ~ 0.1B 3 ~5n 6 ~6n			8 ~2n 7 ~0.054B 9 ~0.08B 8 ~3n 9 ~3n	per byte, best 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 4 1 8 6 13 11 19		9 24 13 11 26 15 29 31 48		
Other NOP (90) Long NOP (0F 1F) PAUSE ENTER ENTER LEAVE XGETBV CPUID RDTSC RDTSCP RDPMC RDRAND RDSEED	a,0 a,b r r	1 6 15 19+6b 4 7 40-83 15 17 19	IP0/1 IP0/1	25 14 66+4b 8 63-270 30 36 20	0.5 0.5 5 14 200 200	

Floating point x87 instructions

	Operands	μops	Unit	Latency	Reciprocal throughput	Remarks
Move instructions						
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	
-STP	m32/m64	6		14	13	
FSTP	m80	8		11	14	
FBSTP	m80	189		''	264	
FXCH	r	3	FP0+1	9	9	
FILD	m	1	11011	5	2	
		6		18	12	
FIST(P) FISTTP	m			10	14	
	m	6				
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 1	
FCOMI(P) FUCOMI(P)	r	3			9	
FIADD FISUB(R)	m	3			17	
FIMUL		3			17	
	m					
FIDIV(R)	m	3			41	
FICOM(P)	m	3			8	
FTST		1			1 1	
FXAM		1			1	
PREM		27		26-47	25-32	
FPREM1		27		33-72	25-32	
FRNDINT		18		36	36	

Math			
FSCALE	30	31	
FXTRACT	15	19	18
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	61	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,(x)mm	1		4	1	
MOVD MOVQ	(x)mm,r32/64	1		5	1	
MOVD MOVQ	m32/64,(x)mm	1	Mem	5	1	
MOVD MOVQ	(x)mm,m32/64	1	Mem	5	0.5	
MOVQ	(x)mm, (x)mm	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 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	(x)mm, (x)mm	1	FP0	2-6	1	
VPACKSSWB/DW						
VPACKUSWB/DW	y/z,y/z,y/z	5		11-14	9	
VPACKSSWB/DW						
/PACKUSWB/DW	y/z,y/z,m	6			9	
PUNPCKH/LBW/WD/DQ	(x)mm, (x)mm	1	FP0	2-6	1	
VPUNPCKH/LBW/WD	y/z,y/z,y/z	5		11-14	9	
VPUNPCKH/LBW/WD	y/z,y/z,m	6			9	

	Tringing Landing							
PUNPCKH/LQDQ	(x)mm, (x)mm	1	FP0	2-6	1 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			
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	y,y,y	12	FP0	23-25	12			
VPSHUFB	y,y,m	13	FP0		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,m,i	5	FP0		9			
(V)PSHUFD	v,v,i	1	FP0	3-6	1			
PALIGNR	mm,mm,i	1	FP0	2-6	1 1			
PALIGNR	x,x,i	1	FP0	2-6	2			
VPALIGNR	y,y,y,i	5	FP0	11-14	9			
VPALIGNR	y,y,y,i y,y,m,i	6	FP0	11-1-	9			
VALIGND/Q	z,z,z,i	1	FP0	3-6	1			
VPCOMPRESSD/Q	z{k},z	1	110	3-6	3			
VPEXPANDD/Q	z{k},z z{k},z	1		3-6	3			
PBLENDVB	x,x,xmm0	5	FP0	9-10	9			
PBLENDVB		6	FP0	9-10	9			
VPBLENDVB	x,m,xmm0	4		8-10	8			
PBLENDW	V,V,V,V	1		2	2			
VPBLENDW	x,x/m,i	1		2	0.5			
VPBLENDD	y,y,y/m,i	1	FP0/1	2	0.5			
VPBLENDMD/Q	v,v,v/m,i	1	FP0/1	2	0.5			
·	z{k},z,z		_					
VPERMO	V,V,V	1	FP0	3-6	1			
VPERMQ	V,V,İ	1	FP0	3-6	1			
VPERM2I128	V,V,V,İ	1	FP0	4-7	2 2			
VPERMI2D VPERMT2D	V,V,V	1	FP0	4-7				
VPERMI2Q VPERMT2Q	V,V,V :	1	FP0	4-7	2			
VSHUFI32X4	Z,Z,Z,İ	1	FP0	4-7	2			
VSHUFI64X2	Z,Z,Z,İ	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	V,X	2		8	7			
VPBROADCASTD/Q	V,X	1	FP0	3	1			
VBROADCASTI128	y,m128	1		5	0.5			

	ı			1	I.
VBROADCASTI32X4	z,m128	1		5	0.5
VBROADCASTI64X4	z,m256	1		5	0.5
Gather and scatter					
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
VPGATHERDQ		6			12
VPGATHERQQ	x,[r+s*x],x	6			12
	y,[r+s*y],y	1			7
VPGATHERQQ VPSCATTERDD	z,[r+s*z],z	4			17
	z,[r+s*z],z				
VPSCATTERDO	y,[r+s*z],y	4			11
VPSCATTERDQ	z,[r+s*y],z	4			11
VPSCATTERQQ	z,[r+s*z],z	4			11
Arithmetic instructions					
PADD/SUB(U,S)B/W/D/Q	(x)mm, (x)mm	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	(x)mm,(x)mm	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	(x)mm,(x)mm	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	y,y,y	5	FP0	16	9
	נינינ	. •			_

PSADBW PSADBW PSADBW	mm,mm x, x y,y,y	1 1 5	FP0 FP0 FP0	6 7 16	1 2 9	
MPSADBW	x,x,i	3		9	9	
VMPSADBW	y,y,y,i	9		19	13	
PAVGB/W	(x)mm,(x)mm	1	FP0/1	2	0.5	
PAVGB/W	y,y,y	1	FP0/1	2	0.5	
PMIN/MAXUB/SW	(x)mm,(x)mm	1	FP0/1	2	2	
VPMIN/MAXUB/SW/D/Q	y,y,y	1	FP0/1	2	0.5	
PHMINPOSUW	X,X	1	FP0	3	2	
PABSB/W/D	· ·	1	FP0/1	2	0.5	
PABSB/W/D	mm,mm	1	FP0/1	2	2	
	X,X	·				
VPABSB/W/D/Q	y,y	1	FP0/1	2	0.5	
PSIGNB/W/D	mm,mm	1	FP0/1	2	0.5	
PSIGNB/W/D	X,X	1	FP0/1	2	2	
VPSIGNB/W/D	y,y	1	FP0/1	2	0.5	
Logic instructions						
PAND(N)/OR/XOR	(x)mm,(x)mm	1	FP0/1	2	0.5	
VPAND(N)/OR/XOR	y,y,y	1	FP0/1	2	0.5	
VPAND(N)/OR/XORD/Q	z,z,z	1	FP0/1	2	0.5	
VPTERNLOGD/Q	z,z,z,i	1	FP0+1	2	1	
PTEST	x,x	4		9	9	
VPTEST(N)MD/Q	k,z,z	1	FP0/1	2	0.5	
VPTEST	y,y	4		9	8	
PSLL/RL/RAW/D/Q	mm,mm	1	FP0	2	1	
PSLL/RL/RAW/D/Q	x,x	2	FP0	13	13	
PSLL/RL/RAW/D/Q	(x)mm,i	1	FP0	2	1	
VPSLL/RL/RAW/D/Q	1 ' '	4	FP0	11	8	
VPSLL/RA/RLVD/Q	y,y,i	1	FP0	2	1	
	Z,Z,Z	=	FP0	2	1	
VPROL/RD/Q	Z,Z,İ	1		2	•	
VPROL/RVD/Q	Z,Z,Z	1	FP0		1	
VPLZCNTD/Q	Z,Z	1	FP0	2	1	
VPCONFLICTD/Q	Z,Z	1	FP0	3	1	
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	1	FP0	3-6	2	+1 if mem
AESDEC, AESDECLAST, AESENC, AESENCLAST, AESIMC,						
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	
(V)MOVAPS/D	m,v	1	Mem	5	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	1.5	
(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 1	
MOVDDUP	x,m	1	110	14	'	
VMOVDDUP	V,V	1		3-6	1	
(V)MOVSH/LDUP	V,V V,V	1	FP0	3-6	1	
VBROADCASTSS/D		1	FFU	3-6	1 1	
	V,X	1				
VBROADCASTSS/D	v,m			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	
VINSERTF64X4	z,z,m256	1		7	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		7	7	
BLENDVPS/PD	x,m,xmm0	3			8	
VBLENDMPS/D	z{k},z,z	1	FP0/1	2	0.5	
SHUFPS/D	x,x,i	1	FP0	4	2	

		J	Ü		
VSHUFPS/D	v,v,v,i	1	FP0	4-7	2
VSHUFF32X4	z,z,z,i	1	FP0	4-7	2
VSHUFF64X2	z,z,z,i	1	FP0	4-7	2
VPERMILPS/PD	v,v/m,i	1	FP0	3-6	1
VPERMILPS/PD	v,v,v/m	1	FP0	3-6	1
VPERM2F128	y,y,y/m,i	1	FP0	4-7	2
VPERMPS/PD	y,y,y/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
Gather and scatter					
VPGATHERDPS	x,[r+s*x],x	6			12
VPGATHERDPS	y,[r+s*y],y	6			12
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		6			12
VPGATHERDPD	y,[r+s*x],y	1			7
	z,[r+s*y],z				
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
VGATHERPF0DPD	z,[r+s*y],z	1			~100
VGATHERPF0QPD	z,[r+s*z],z	1			~100
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
Conversion					
(V)CVTSD2SS	x,x	1	FP0	2	1
(V)CVTSS2SD	x,x	1	FP0	2	1
(V)CVTPD2PS		2	FP0	7	7
(V)CVTPS2PD	V,V	2	FP0	7	7
VCVTPS2PH	V,V	2		7	7
	V,V			'	
VCVTPH2PS	m,v	5		7	9 7
VCVTPH2PS	V,V	2		7	/

		rangino i	-arrainig		
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 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
CVT(T)PS2PI	mm,x	1	FP0	3	1
CVTPI2PS	x,mm	2	FP0	7	7
CVT(T) PD2PI		2	FP0	7	7
CVT(1) PD2P1 CVTPI2PD	mm,x	2	FP0	7	7
	x,mm			2	
(V)CVT(T) PS2DQ	V,V	1	FP0		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	v,v,v	1	FP0/1	6	0.5
ADDPD SUBPD	x,x	1	FP0/1	6	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	110/1	15	8
VHADDPS/D VHSUBPS/D	yy,y,	3		15	8
MULSS/D	X,X	1	FP0/1	6	0.5
MULPS/D	X,X X,X	1	FP0/1	6	0.5
VMULPS/D		1	FP0/1	6	0.5
DIVSS	V,V,V	3	FP0	27	17
DIVSD	X,X	3	FP0	42	42
DIVPS	X,X	18	FP0	32	
VDIVPS	X,X	18	FP0	32	20 32
DIVPD	V,V,V	18	FP0	32	
VDIVPD	X,X	18	FP0	32	20 32
	V,V,V			32 7	
RCPSS	X,X	1	FP0		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		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	X,X	1	FP0/1	2	0.5
VMAXPS/D VMINPS/D	V,V,V	1	FP0/1	2	0.5

		3	3			
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		6	0.5	
VSCALEFSS/D	x,x,x	1	FP0/1	6	0.5	
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	
Math						
SQRTSS	x,x	3	FP0	28	18	
SQRTPS	x,x	18	FP0	38	16	
VSQRTPS	V,V	18	FP0	38	16	
SQRTSD	x,x	30	FP0	43	35	
SQRTPD	x,x x,x	18	FP0	37	16	
VSQRTPD	V,V	18	FP0	37	16	
RSQRTSS		10	FP0	7	2	
	X,X	=		1		
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	V,V	1	FP0	7	3	
VRSQRT28SD	V,V,V	1	FP0	7	2	
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	
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	v,v,v	1	FP0/1	2	0.5	
VORPS/D VXORPS/D	v,v,v	1	FP0/1	2	0.5	
Other						
VZEROUPPER	1	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	o i bit illouc
STMXCSR	m32	5			15	
FXSAVE		90			113	32 bit mode
FXSAVE	m m	98			113	64 bit mode
	m					
FXRSTOR	m	98			122	32 bit mode
FXRSTOR	m	114			130	64 bit mode

Knights Landing

FNSAVE	m	135	205	205	
FRSTOR	m	78	191	191	
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 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, (x)mm = 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 $\mu\text{ops}.$ Therefore the number of μops 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	
MOVSX MOVSXD		_				
MOVZX	r,r	1	I2	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	r,m			20	20	Implicit lock
XLAT	m			6		
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	
POP	(E/R)SP				4	
POP	m				5	
POP	sr				20	
POPF(D/Q)	<u>.</u>			9	9	
POPA(D)					12	Not in x64 mode
LAHF		1	I1	1	1	Trock in No 1 mode
SAHF		1	11	1	1	
SALC		'		9	6	Not in x64 mode
LEA	r,m	1	SA	1	1	3 clock latency on
	1,111	'	OA	•	'	input register
BSWAP	r	1	12	1	1	
LDS LES LFS LGS LSS						
	m			30	30	
PREFETCHNTA	m		LD		1-2	
PREFETCHT0/1/2	m		LD		1-2	
LFENCE					14	
MFENCE					14	
SFENCE					14	
Arithmetic instructions		_				
ADD SUB	r,r/i	1	l12	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					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
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		1	I1	1	1	
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 I12		1	
AND OR XOR	m,r/i	3	LD I12 SA ST	5	2	
TEST	r,r/i	1	l12	1	1/2	
TEST	m,r/i	2	LD I12		1	
SHR SHL SAR	r,i/cl	1	I1	1	1	
ROR ROL	r,i/cl	1	I1	1	1	
RCR RCL	r,1	1	I1	1	1	
RCR RCL	r,i/cl		I1	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		l1	7	7	
SHLD	r64,r64,cl		l1	33	33	
SHRD	r64,r64,cl		l1	43	43	
BT	r,r/i	1	l1	1	1	
BT	m,r		I1		8	
BT	m,i	2	l1		1	
BTR BTS BTC	r,r/i	2	l1	2	2	
BTR BTS BTC	m,r		I1	10	10	
BTR BTS BTC	m,i		l1	8	8	
BSF BSR	r,r		l1	3	2	

SETcc	r		I 1	2	1	
SETcc	m				1	
CLC STC CMC			I1	3	3	
CLD STD				3	3	
OLD OTD						
Control transfer instruc	tions					
JMP	short/near	1	12	3	3	8 if >2 jumps in 16 bytes block
JMP	far			58		Not in x64 mode
	_					8 if >2 jumps in 16
JMP	r		12	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.
Conditional jump	Shortmean			1-0-0	1-0-0	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.
	short			25	25	do.
LOOP(N)E	SHOIL			25	25	O if > O it man a in 16
CALL	near			3	3	8 if >2 jumps in 16 bytes block
CALL	far			72	72	Not in x64 mode
						8 if >2 jumps in 16
CALL	r			3	3	bytes block
CALL	m(near)			4	3	do.
CALL	m(far)			72	72	
	, ,					8 if >2 jumps in 16
RETN				3	3	bytes block
RETN	i			3	3	do.
RETF				39	39	
RETF	i			39	39	
BOUND	r,m				13	Not in x64 mode
INTO	.,				7	Not in x64 mode
					1	Not in xo4 mode
String instructions						
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	
REP MOVSB/W/D/Q					Small: 2n+45,	
THE WOVE DIWING					Big: 6 bytes	
					per clock	
SCASB/W/D/Q					1	
REP SCASB					2.2n	
1	1			1	I .	1 I

REP SCASW/D/Q					Small: 2n+50 Big: 5 bytes per clock	
CMPSB/W/D/Q					6	
REP CMPSB/W/D/Q					2.4n+24	
Other						
NOP (90)		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		
RDTSC					39	
RDPMC				40	40	

Floating point x87 instructions

	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			7		
FILD	m32			5		
FILD	m64			5		
FIST(T)(P)	m16			6		
FIST(T)(P)	m32			5		
FIST(T)(P)	m64			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						

	,		MD			Lower precision:
FADD(P) FSUB(R)(P)	r/m	1	MB	2	1 2	Lat: 4, Thr: 2
FMUL(P)	r/m	1	MA	4		
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 MB		1 1	
FCOMI(P) FUCOMI(P)	r	'	MB		2	
FIADD FISUB(R) FIMUL	m		IVID		4	
FIDIV(R)	m m				4 42	
FICOM(P)	m	1			2	
FTST	111	1	MB		1	
FXAM		'	IVID		41	
FPREM				151-171	41	
FPREM1				106-155		
FRNDINT				29		
TATOMY						
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						
MOVD	r32/64,(x)mm	1		3	1	
MOVD	n32/64,(x)mn	1	SA ST	2-3	1-2	
MOVD	(x)mm,r32/64			4	1	
MOVD	x)mm,m32/64	1	LD	2-3	1	
MOVQ	x)mm, (x)mm	1	MB	1	1	
MOVQ	(x)mm,m64	1	LD	2-3	1	
MOVQ	m64, (x)mm	1	SA ST	2-3	1-2	
MOVDQA	xmm, xmm	1	MB	1	1	
MOVDQA	xmm, m128	1	LD	2-3	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
LDDQU	xmm, m128	1	LD	2-3	1
MOVDQ2Q	mm, xmm	1	MB	1	1
MOVQ2DQ	xmm,mm	1	MB	1	1
MOVNTQ	m64,mm	3		~300	2
MOVNTDQ	m128,xmm	3		~300	2
PACKSSWB/DW	111120,2111111	O		000	_
PACKUSWB	V,V	1	MB	1	1
PUNPCKH/LBW/WD/	, v, v	•	IVID		'
DQ	V,V	1	MB	1	1
PUNPCKH/LQDQ	V,V V,V	1	MB	1	1
PSHUFB	1	1	MB	1	1
	V,V	-		-	1
PSHUFW	mm,mm,i	1	MB	1	
PSHUFL/HW	X,X,İ	1	MB	1	1
PSHUFD	x,x,i	1	MB	1	1
PALIGNR	x,x,i	1	MB	1	1
MASKMOVQ	mm,mm				1-3
MASKMOVDQU	xmm,xmm				1-3
PMOVMSKB	r32,(x)mm			3	1
PEXTRW	r32 ,(x)mm,i			3	1
PINSRW	(x)mm,r32,i			9	9
Arithmetic instructions	ı				
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

Floating point XIVIIV		1		T -		
	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	
SHUFPD	xmm,xmm,i	1	MB	1	1	
MOVDDUP	xmm,xmm	1	MB	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	
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		
CVT(T)PD2DQ	xmm,xmm			3		
CVTPI2PS	xmm,mm			4		
CVT(T)PS2PI	mm,xmm			3		
CVTPI2PD	xmm,mm			4		
CVT(T) PD2PI	mm,xmm			3		
CVTSI2SS	xmm,r32			5		
CVT(T)SS2SI	r32,xmm			4		
CVTSI2SD	xmm,r32			5		
CVT(T)SD2SI	r32,xmm			4		
Arithmetic						
ADDSS SUBSS	xmm,xmm	1	MBfadd	2-3	1	

ADDSD SUBSD	xmm,xmm	1	MBfadd	2-3	1	
ADDPS SUBPS	xmm,xmm	1	MBfadd	2-3	1 1	
ADDPD SUBPD	xmm,xmm	1	MBfadd	2-3	1	
ADDSUBPS	xmm,xmm	1	MBfadd	2-3	1	
ADDSUBPD	xmm,xmm	1	MBfadd	2-3	1	
HADDPS HSUBPS	xmm,xmm	-	MBfadd	5	3	
HADDPD HSUBPD	xmm,xmm		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	,					
	xmm,xmm			3	1	
MAXSS/D MINSS/D	xmm,xmm	1	MBfadd	2	1	
MAXPS/D MINPS/D	xmm,xmm	1	MBfadd	2	1	
Math						
SQRTSS	xmm,xmm		MA	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

1 11 1 Op 0 0 1110 1110		
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, (x)mm = 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 $\mu\textsc{ops}.$ Therefore the number of $\mu\textsc{ops}$ 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	µорѕ	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	r,r	1	I12	1	1/2	
MOVSXD	r64,r32	1	112	1	1	
MOVSX MOVSXD	r,m	2	LD, I12	3	1	
MOVZX	r,m	1	LD	2	1	
CMOVcc	r,r	1	l12	1	1/2	
CMOVcc	r,m		LD, I12	5	1	
XCHG	r,r	3	I12	3	1,5	
XCHG	r,m			18	18	Implicit lock
XLAT	m	3	LD, I1	6	2	Implicit rook
PUSH	r	1	SA, ST		1-2	
PUSH	i i	1	SA, ST		1-2	
PUSH	m .		LD, SA, ST		2	
PUSH	sr		22, 3, 1, 3.		6	
PUSHF(D/Q)	01	3		2	2	
PUSHA(D)		9		_	_ 15	Not in x64 mode
POP	r	2	LD		1,25	Troc in Ao T modo
POP	(E/R)SP	_			4	
POP	m (Z/13)	3			2	
POP	sr				11	
POPF(D/Q)	01	3			1	
POPA(D)		16			12	Not in x64 mode
LAHF		1	l1	1	1	140t III XO I IIIOGO
SAHF		1	11	1	1	
SALC		2		10	6	Not in x64 mode
S. 1.25		_		. •		Extra latency to other
LEA	r,m	1	SA	1	1	ports
BSWAP	r	1	12	1	1	
LDS LES LFS LGS LSS						
	m	12		28	28	
PREFETCHNTA	m	1	LD		1	
PREFETCHT0/1/2	m	1	LD		1	
LFENCE MFENCE						
SFENCE					15	
Arithmetic instructions				,		
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	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	I12	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	^ -	Notice Of
AAA		12			37	Not in x64 mode
AAS		12			22	Not in x64 mode
DAA		14			22	Not in x64 mode

DAS AD 14 7 24 Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode Mull immul. MUL immul. r8 1 12 2 24 Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode Not in x64 mode Mull immul. MUL immul. r16 3 12 3 Extra latency to other ports MUL immul. r64, r64 1 12 2 1 Extra latency to other ports IMUL r64, r64 1 MA 5 2 ports IMUL r16, r16, i 1 12 2 1 Extra latency to other ports IMUL r16, r16, i 1 12 2 1 Extra latency to other ports IMUL r164, r64, i 1 MA 5 2 ports IMUL r164, r64, i 1 MA 5 2 1 Extra latency to other ports IMUL r164, r64, i 1 MA 5 2 1 Extra latency to other ports IMUL							
AAM MUL IMUL MUL IMUL R8 1 1 12 2 MUL IMUL R64 3 12 3 MUL IMUL R64 3 MAA 8 8 8 MINUL IMUL R64 3 MAA 8 8 8 MINUL IMUL R64 1 12 2 1 MUL MUL R64 1 1 12 2 1 MUL MUL R64 1 1 MAA 5 2 2 MINUL MUL R64 1 1 MAA 5 2 2 MINUL MUL R64 1 1 MAA 5 2 2 MINUL MUL R64, 664, 1 1 12 2 1 MINUL R64, 664, 1 1 MAA 5 2 2 MINUL R64, 664, 1 1 MAA 5 2 2 MINUL MINUL R64, 664, 1 1 MAA 5 2 2 MINUL MINUL R64, 664, 1 1 MAA 5 2 2 MINUL R64, 664, 1 1 MAA 5 2 2 MINUL R64, 664, 1 1 MAA 24-28 24-28 MINUL MINUL R64 MAA 24-28 24-28 MINUL MINUL R64 MAA 145-162 145-162 MINUL MINUL R64 MAA 145-162 145-162 MINUL MINUL R64 MAA 145-162 145-162 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL R64 MAA 18-26 18-26 MINUL MINUL MINUL R64 MAA 18-26 18-26 MINUL MI	DAS		14			24	Not in x64 mode
MUL IMUL R8 1 12 2 2 AUL IMUL FRANCE TRANCE AUL IMUL FRANCE TRANCE AUL IMUL FRANCE TRANCE AUL IMUL FRANCE TRANCE AUL IMUL FRANCE TRANCE AUL IMUL FRANCE AUL IMUL AUL IMUL FRANCE AUL IMUL AUL IMUL FRANCE AUL IMUL AUL IMUL FRANCE AUL IMUL AUL IMUL AUL IMUL AUL IMUL AUL IMUL AUL IMUL AUL IMUL AUL IMUL AUL IMUL AUL IMUL AUL IMUL AUL IMUL	AAD		7			24	Not in x64 mode
MUL IMUL r16 3 12 3 Lexical latency to other ports MUL IMUL r64 3 IA 2 3 Extra latency to other ports MUL IMUL r16,r16 1 I2 2 1 Image: control imag	AAM		13			31	Not in x64 mode
MUL IMUL r16 3 12 3 Extra latency to other ports MUL IMUL r64 3 MA 8 8 8 Extra latency to other ports IMUL r16,r16 1 12 2 1 Extra latency to other ports IMUL r64,r64 1 MA 5 2 1 Extra latency to other ports IMUL r64,r64,ii 1 MA 5 2 1 Extra latency to other ports IMUL r64,r64,ii 1 MA 5 2 1 Extra latency to other ports IMUL r64,r64,ii 1 MA 5 2 1 Extra latency to other ports IMUL r64,r64,ii 1 MA 5 2 2 2 2 2 4 2 2 2 2 4 2 2 2 2 4 2 2 2 2 2 3 1 1 1 1 1	MUL IMUL	r8	1	12	2		
MUL IMUL r32 3 12 3 Extra latency to other ports MUL IMUL r64 3 MAA 8 8 8 IMUL r16,r16 1 12 2 1 Extra latency to other ports IMUL r64,r64 1 MA 5 2 1 Extra latency to other ports IMUL r16,r16,i 1 I2 2 1 Extra latency to other ports IMUL r64,r64,i 1 MA 5 2 1 Extra latency to other ports IMUL r64,r64,i 1 MA 5 2 1 Extra latency to other ports IMUL r64,r64,i 1 MA 22-24	MUL IMUL	r16	3	12			
MULI IMUL r64 3 MA 8 8 ports IMUL r16,r16 1 12 2 1 Extra latency to other ports IMUL r64,r64 1 MA 5 2 ports IMUL r16,r16,i 1 12 2 1 Extra latency to other ports IMUL r16,r16,i 1 IZ 2 1 Extra latency to other ports IMUL r16,r16,i 1 MA 5 2 ports IMUL r164,r64,i 1 MA 22-24 22-24 22-24 IMUL r164,r64,i 1 MA 22-30 22-30 22-30 DIV r16 MA 22-24 22-24 22-24 22-24 DIV r64 MA 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162	MUL IMUL	r32	3	12			
MULI IMUL r64 3 MA 8 8 ports IMUL r16,r16 1 12 2 1 Extra latency to other ports IMUL r64,r64 1 MA 5 2 ports IMUL r16,r16,i 1 12 2 1 Extra latency to other ports IMUL r16,r16,i 1 IZ 2 1 Extra latency to other ports IMUL r16,r16,i 1 MA 5 2 ports IMUL r164,r64,i 1 MA 22-24 22-24 22-24 IMUL r164,r64,i 1 MA 22-30 22-30 22-30 DIV r16 MA 22-24 22-24 22-24 22-24 DIV r64 MA 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162 145-162							Extra latency to other
MUL	MUL IMUL	r64	3	MA	8	8	
MULL	IMUL	r16,r16	1	12	2	1	
IMUL	IMUL	r32,r32	1	12	2	1	
MUL							Extra latency to other
MUL		r64,r64	1	MA	5		ports
MULL r64,r64,i 1 MA 5 2 Extra latency to other ports	IMUL	r16,r16,i	1	12		1	
IMUL	IMUL	r32,r32,i	1	12	2	1	
DIV							
DIV			1				ports
DIV F32							
DIV F64 F8							
IDIV					22-30		
IDIV					145-162		
IDIV	IDIV	r8		MA	21-24	21-24	
IDIV	IDIV	r16		MA	24-28	24-28	
CBW CWDE CDQE 1 12 1 1 CWD CDQ CQO 1 1 12 1 1 Logic instructions 7,r/i 1 112 1 1/2 AND OR XOR 7,rm 2 LD 112 1 1/2 AND OR XOR 7,r/i 3 LD 112 SA ST 5 2 2 TEST 7,r/i 1 112 1 1/2 1/2 1 1 <td>IDIV</td> <td>r32</td> <td></td> <td>MA</td> <td>18-26</td> <td>18-26</td> <td></td>	IDIV	r32		MA	18-26	18-26	
Logic instructions 1 12 1 1 AND OR XOR r,r/i 1 112 1 1/2 AND OR XOR r,m 2 LD 112 1 1 AND OR XOR m,r/i 3 LD 112 SAST 5 2 TEST r,r/i 1 112 1 1/2 TEST m,r/i 2 LD 112 1 1/2 SHR SHL SAR r,i/cl 1 112 1 1/2 ROR ROL r,i/cl 1 112 1 1/2 ROR ROL r,i/cl 1 11 2 2 2 1 1 2 2 2 1	IDIV	r64		MA	182-200	182-200	
Logic instructions	CBW CWDE CDQE		1	12	1	1	
AND OR XOR AND OR XOR	CWD CDQ CQO		1	12	1	1	
AND OR XOR AND OR XOR							
AND OR XOR AND OR XOR AND OR XOR AND OR XOR 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 112 1 1 SHR SHL SAR r,i/cl 1 112 1 1/2 ROR ROL r,i/cl 1 11 1 1 1 1 1 1 RCR RCL r,1 1 1 11 1 1 1 RCR RCL r,i/cl 5+2n 11 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 SHRD r64,r64,i/cl 16 11 32 32 SHRD r64,r64,i/cl 23 11 42 42 BT r,r/i 1 11 1 1 1 BT BT m,r 6 11 BT m,r 6 11 BT BT m,r 6 11 BT BT m,r 6 11 BT BT m,r 6 11 BT BT m,r 6 11 BT BT m,r 6 11 BT BT m,r 6 11 BT BT BT BT BT BT BT BT BT BT BT BT BT							
AND OR XOR TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TEST T,r/i TI TI TI TI TI TI TI TI TI T					1	1/2	
TEST r,r/i 1 I12 1 1/2 TEST m,r/i 2 LD I12 1 1/2 SHR SHL SAR r,i/cl 1 I12 1 1/2 ROR ROL r,i/cl 1 I1 1 1 RCR RCL r,i/cl 5+2n I1 28+3n 28+3n SHLD SHRD r16,r16,i/cl 2 I1 2 2 SHLD SHRD r32,r32,i/cl 2 I1 2 2 SHLD SHRD 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 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8							
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 I1 1 1 1 RCR RCL r,1 1 I1 1 1 1 RCR RCL r,i/cl 5+2n I1 28+3n 28+3n SHLD SHRD r16,r16,i/cl 2 I1 2 2 SHLD SHRD r32,r32,i/cl 2 I1 2 2 SHLD F64,r64,i/cl 16 I1 32 32 SHRD r64,r64,i/cl 23 I1 42 42 BT m,r'i 1 I1 1 1 BT m,r 6 I1 8 8 BT BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2							
SHR SHL SAR r,i/cl 1 I12 1 1/2 ROR ROL r,i/cl 1 I1 1 1 RCR RCL r,1 1 I1 1 1 RCR RCL r,i/cl 5+2n I1 28+3n 28+3n SHLD SHRD r16,r16,i/cl 2 I1 2 2 SHLD SHRD r32,r32,i/cl 2 I1 2 2 SHLD r64,r64,i/cl 16 I1 32 32 SHRD r64,r64,i/cl 23 I1 42 42 BT m,r/i 1 I1 1 1 BT m,r/i 1 I1 1 1 BT m,r 6 I1 8 8 BT m,i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETC		r,r/i			1		
ROR ROL r,i/cl 1 I1 1 1 1 RCR RCL r,1 1 I1 1 1 1 RCR RCL r,i/cl 5+2n I1 28+3n 28+3n SHLD SHRD r16,r16,i/cl 2 I1 2 2 SHLD SHRD r32,r32,i/cl 2 I1 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 8 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1		m,r/i	2				
RCR RCL r,1 1 I1 1 1 28+3n 28+3n SHLD SHRD r16,r16,i/cl 2 I1 2 2 SHLD SHRD r32,r32,i/cl 2 I1 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 1 BTR BTS BTC r,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETCC r8 1 I1 1 1			1		1	1/2	
RCR RCL r,i/cl 5+2n I1 28+3n 28+3n SHLD SHRD r16,r16,i/cl 2 I1 2 2 SHLD SHRD r32,r32,i/cl 2 I1 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 8 BT BTS BTC r,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETCC r8 1 I1 1 1		r,i/cl	1		1	1	
SHLD SHRD r16,r16,i/cl 2 I1 2 2 SHLD SHRD r32,r32,i/cl 2 I1 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 m,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1		r,1	1		1	1	
SHLD SHRD r32,r32,i/cl 2 I1 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 r,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1					28+3n	28+3n	
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 r,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1		r16,r16,i/cl					
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 r,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1	SHLD SHRD	r32,r32,i/cl	2	l1	2	2	
BT r,r/i 1 I1 1 1 1 1 1 1 1 1 1 1 1 1 8 </td <td>SHLD</td> <td>r64,r64,i/cl</td> <td>16</td> <td>l1</td> <td>32</td> <td>32</td> <td></td>	SHLD	r64,r64,i/cl	16	l1	32	32	
BT m,r 6 I1 8 BT m,i 2 I1 1 BTR BTS BTC r,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1	SHRD	r64,r64,i/cl	23	l1	42	42	
BT m,i 2 I1 1 BTR BTS BTC r,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1		r,r/i	1	l1	1		
BTR BTS BTC r,r/i 2 I1 2 2 BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1	ВТ	m,r	6	I1		8	
BTR BTS BTC m,r 8 I1 10 10 BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1	BT	m,i	2	l1			
BTR BTS BTC m,i 5 I1 8 8 BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1	BTR BTS BTC	r,r/i	2	l1	2	2	
BSF BSR r,r 2 I1 2 2 SETcc r8 1 I1 1 1	BTR BTS BTC	m,r	8	l1	10	10	
SETcc r8 1 I1 1 1	BTR BTS BTC	m,i	5	l1	8	8	
	BSF BSR	r,r	2	l1	2	2	
SETcc	SETcc	r8	1	l1	1	1	
	SETcc	m	2			2	

CLC STC CMC		3	l1	3	3	
CLD STD		3	l1	3	3	
Control transfer instruc	tions					
JMP	short/near	1	12	3	3	8 if >2 jumps in 16 bytes block
JMP	far	14	12		50	Not in x64 mode
				_		8 if >2 jumps in 16
JMP	r	2	I2	3	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	short	2		1-3-8	1-3-8	
LOOP	short	2		1-3-8	1-3-8	
LOOP(N)E	short	5		24	24	0 if a 0 in man a in 40
CALL	near	2		3	3	8 if >2 jumps in 16 bytes block
CALL	far	17			58	Not in x64 mode
O' LE	141					8 if >2 jumps in 16
CALL	r	2		3	3	bytes block
CALL	m(near)	3		4	3	do.
CALL	m(far)	19			54	
						8 if >2 jumps in 16
RETN		3		3	3	bytes block
RETN	i	4		3	3	do.
RETF RETF	i	20 20			49 49	
BOUND	r,m	20 9			13	Not in x64 mode
INTO	1,111	3			7	Not in x64 mode
		3			,	Not in xo+ 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	
REP STOSB/W/D/Q					bytes/clk	
MOVSB/W/D/Q		3			2	
					Small:	
					2n+20,	
REP MOVSB/W/D/Q					Big: 6-7 bytes/clk	
SCASB/W/D/Q		3			1	
REP SCASB		J			2.4n	
					Small:	
					2n+31,	
					Big: 5	
REP SCASW/D/Q					bytes/clk	

CMPSB/W/D/Q		5			6	
REP CMPSB/W/D/Q					2.2n+30	
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	a,b				52+5b	
LEAVE		3		2	2	
CPUID				55-146		
RDTSC					37	
RDPMC					40	

Floating point x87 instructions

	Operands	µops	Port	Latency	Reciprocal thruogh- put	Remarks
Move instructions						
FLD	r	1	MB	1	1 1	
FLD	m32/m64	2	LD MB	4	1 1	
FLD	m80	2	LD MB	4	1 1	
FBLD	m80	36		54	54	
FST(P)	r	1	MB	1	1 1	
FST(P)	m32/m64	3	MB SA ST	5	1-2	
FSTP	m80	3	MB SA ST	5	1-2	
FBSTP	m80	80		125	125	
FXCH	r	1	12	0	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		1	MB		1 1	
FLDPI FLDL2E etc.		3			10	
FCMOVcc	r	1	MB	2	2	
FNSTSW	AX	1			1	
FNSTSW	m16	3			2	
FLDCW	m16	5			8	
FNSTCW	m16	3			2	
FINCSTP FDECSTP		1	12	0	1 1	
FFREE(P)		1	MB		1 1	
FNSAVE	m	122		319	319	
FRSTOR	m	115		196	196	
Arithmetic instructions	 }					
FADD(P) FSUB(R)(P)	r/m	1	MB	2	1 1	

FMUL(P)	r/m	1	MA	4	2	
FDIV(R)(P)	r/m		MA	14-23	14-23	
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	2	1	
FIADD FISUB(R)	m	3	MB		2	
FIMUL	m	3			4	
FIDIV(R)	m	3			16	
FICOM(P)	m	3			2	
FTST		1	MB	2	1	
FXAM		15		38	38	
FPREM				~130		
FPREM1				~130		
FRNDINT		11		27		
B# - 41-						
Math				07		
FSCALE		22		37		
FXTRACT		13		57		L aga of lawer
FSQRT				73		Less at lower precision
FSIN FCOS				~150		
FSINCOS				270-360		
F2XM1				50-200		
FYL2X				~50		
FYL2XP1				~50		
FPTAN				300-370		
FPATAN				~180		
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,(x)mm	1	MB	3	1 1	
MOVD	m,(x)mm	1	SA ST	2	1-2	
MOVD	(x)mm,r	1	12	4	1 1	
MOVD	(x)mm,m	1	LD	2	1 1	
MOVQ	V,V	1	MB	1	1 1	
MOVQ	(x)mm,m64	1	LD	2	1 1	
MOVQ	m64, (x)mm	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 x, m128 1 LD 2 1 MOVDQU x, m128 1 LD 2 1 MOVDQQ x, m128 1 LD 2 1 MOVDQQQ x, m128 1 LD 2 1 MOVNTQ m64, mm 2 ~360 2 MOVNTDQ m128, x 2 ~360 2 MOVNTDQ m128, x 2 ~360 2 MOVNTDQA m128, x 2 1 1 PACKSSWB/DW v,v 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1	1	l	ı .	l	l .	
MOVDQU m128, x 1 SAST 2 1-2 MOVDQU x, m128 1 LD 2 1 MOVDQ2Q mm, x 1 MB 1 1 MOVQ2DQ mm, x 1 MB 1 1 MOVNTQ m64,mm 2 ~360 2 MOVNTDQA x,m128 1 2 360 2 MOVNTDQA x,m128 1 2 1 7 PACKSSWB/DW pACKUSDW v,v 1 MB 1 1 1 PACKUSDW v,v 1 MB 1 <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>						
MOVDQU						
LDDQU			1	SA ST		
MOVDQ2Q mm, x 1 MB 1 1 MOVAQ2DQ x,mm 1 MB 1 1 MOVNTQ m64,mm 2 ~360 2 MOVNTDQA m128,x 2 ~360 2 MOVNTDQA x,m128 1 2 1 PACKUSWB v,v 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PSHUED x,x 1 MB 1 1 PSHUED x,x,i 1 MB	· ·		1			· ·
MOVQ2DQ x,mm 1 MB 1 1 MOVNTQ m64,mm 2 ~360 2 MOVNTDQ m128,x 2 ~360 2 MOVNTDQA m128,x 2 -360 2 MOVNTDQA x,m128 1 2 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PUNPCKH/LQDQ v,v 1 MB 1 1 PSHUEL/BW mm,mmi 1 MB 1 1 PSHUFU/HW x,x,i 1 MB 1 1 PSHUFD x,x,i 1 MB <td>LDDQU</td> <td>x, m128</td> <td>1</td> <td>LD</td> <td>2</td> <td>1</td>	LDDQU	x, m128	1	LD	2	1
MOVNTQ m64,mm 2 ~360 2 MOVNTDQA m128,x 2 ~360 2 MOVNTDQA x,m128 1 2 1 PACKSSWB/DW x,m128 1 2 1 PACKSSWB/DW v,v 1 MB 1 1 PACKSSWB/DW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PUNPCKH/LBWMD/DQ v,v 1 MB 1 1 PSHUFB v,v 1 MB 1 1 1 PSHUFW mm,mm,mi 1 MB 1 1 1 1 1 1 1 1 1 1 1 1 1	MOVDQ2Q	mm, x	1	MB	1	1
MOVNTDQA m128,x 2 ~360 2 MOVNTDQA x,m128 1 2 1 PACKSSWB/DW x,xm128 1 2 1 PACKUSWB V,V 1 MB 1 1 PACKUSDW x,X 1 MB 1 1 PACKUSDW x,X 1 MB 1 1 PACKUSDW x,X 1 MB 1 1 PACKUSDW x,X 1 MB 1 1 PACKUSDW x,X 1 MB 1 1 PUNPCKH/LQDQ y,V 1 MB 1 1 PSHUFD mMB 1 <td< td=""><td>MOVQ2DQ</td><td>x,mm</td><td>1</td><td>MB</td><td>1</td><td>1 1</td></td<>	MOVQ2DQ	x,mm	1	MB	1	1 1
MOVNTDQA x,m128 1 2 1 PACKSSWB/DW y,v 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PUNPCKH/LBW/WD/DQ v,v 1 MB 1 1 PUNPCKH/LQDQ v,v 1 MB 1 1 PSHUFB v,v 1 MB 1 1 PSHUFW mm,mm,i 1 MB 1 1 PSHUFW mm,mm,i 1 MB 1 1 PSHUFL/HW x,x,i 1 MB 1 1 PSHUFU/HW x,x,i 1 MB 1 1 PSHUFD x,x,i 1 MB 1 1 PSHUFL/HW x,x,x,i 1 MB 1 1 PBLENDW x,x,i 1 MB 1 1 PBLENDW<	MOVNTQ	m64,mm	2		~360	2
PACKSSWB/DW PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSWB PACKUSDW PACKUSDW PACKUSDW PACKUSDW PACKUSDW PUNPCKH/LBWWD/DQ V,V 1 MB 1 1 PUNPCKH/LQDQ V,V 1 MB 1 1 PUNPCKH/LQDQ V,V 1 MB 1 1 PSHUFB V,V 1 MB 1 1 PSHUFW PSHUFW PSHUFD 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 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 3 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 1 1 PALIGNR X,X,i 1 MB 3 1 PEXTRB/D/Q PMOVMSKB REXTRB/D/Q PINSRW (x)mm,r32,i 2 MB 5 1 PHOVSX/ZXBW/BD/ BQ/WD/WQ/DQ X,X 1 MB 1 1 PADD/SUB(U)(S)B/W/D V,V 1 MB 1 1 PADD/SUB(U)(S)B/W/D PHADD(S)W PHSUB(S)W PHSUB(S)W PHSUB(S)W PHSUB(S)W V,V 3 MB 3 3 PHADDD PHSUBD V,V 3 MB 3 3 PHADDD PHSUBD V,V 1 MB 1 1 PCMPEQQ X,X 1 MB 1 1 PCMPEQQ X,X 1 MB 1 1 PCMPEQQ X,X 1 MB 1 1 PCMPEQQ X,X 1 MB 1 1 PCMPEQQ X,X 1 MB 1 1 PCMPEQQ X,X 1 MA 3 1 PMULL/HW PMULHUW V,V 1 MA 3 1 PMULLDQ X,X 1 MA 3 1 PMULLDQ X,X 1 MA 3 1 PMULLDQ Y,V 1 MA 3 1 PMULLDQ Y,V 1 MA 3 1 PMULLDQ Y,V 1 MA 3 1 PMULLDQ PMADDUBSW V,V 7 1 MB 2 1 NBSADBW V,V 1 MB 2 1 NBSADBW X,X,i 1 MB 2 1	MOVNTDQ	m128,x	2		~360	2
PACKUSWB v,v 1 MB 1 1 PACKUSDW x,x 1 MB 1 1 PUNPCKH/LBWWD/DQ v,v 1 MB 1 1 PUNPCKH/LQDQ v,v 1 MB 1 1 PSHUFB v,v 1 MB 1 1 PSHUFW mm,mm,i 1 MB 1 1 PSHUFW mm,mm,i 1 MB 1 1 PSHUFLHW x,x,i 1 MB 1 1 PSHUFLHW x,x,i 1 MB 1 1 PSHUFLHW x,x,i 1 MB 1 1 PSHUFLHW x,x,i 1 MB 1 1 PSHUFLHW x,x,i 1 MB 1 1 PBLENDVB x,x,i 1 MB 1 1 PBLENDW x,x,i 1 MB 1 1	MOVNTDQA	x,m128	1		2	1
PACKUSDW	PACKSSWB/DW					
PUNPCKH/LBW/WD/DQ	PACKUSWB	V,V	1	MB	1	1
PUNPCKH/LQDQ	PACKUSDW	x,x	1	MB	1	1
PSHUFB	PUNPCKH/LBW/WD/DQ	V,V	1	MB	1	1
PSHUFW	PUNPCKH/LQDQ	V,V	1	MB	1	1
PSHUFL/HW	PSHUFB	V,V	1	MB	1	1
PSHUFD	PSHUFW	mm,mm,i	1	MB	1	1
PSHUFD	PSHUFL/HW	x,x,i	1	MB	1	1
PBLENDVB x,x,xmm0 1 MB 2 2 PBLENDW x,x,i 1 MB 1 1 PALIGNR x,x,i 1 MB 1 1 MASKMOVQ mm,mm 1-2 1 MB 1 1 MASKMOVDQU x,x 1 MB 1 1 1 MASKMOVDQU x,x 732,(x)mm 3 1 1-2	PSHUFD		1	МВ	1	1
PBLENDW x,x,i 1 MB 1 1 PALIGNR x,x,i 1 MB 1 1 MASKMOVQ mm,mm 1-2 1-2 MASKMOVDQU x,x 1-2 1-2 MASKMOVDQU x,x 1-2 1-2 MASKMOVDQU x,x 1 MB 3 1 PEXTRW r32,(x)mm,i 1 MB 3 1 PEXTRB/D/Q r32/64,x,i 1 MB 3 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PMOVSX/ZXBW/BD/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 1 PHADD(S)W v,v 3 MB 3 3 3			1		2	2
PALIGNR x,x,i 1 MB 1 1 MASKMOVQ mm,mm 1-2 1-2 MASKMOVDQU x,x 1-2 1-2 PMOVMSKB r32,(x)mm 3 1 PEXTRW r32,(x)mm,i 1 MB 3 1 PEXTRB/D/Q r32/64,x,i 1 MB 3 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PMOVSX/ZXBW/BD/BQ/WD/WD/WQ/DQ x,x 1 MB 1 1 Arithmetic instructions PADD/SUB(U)(S)B/W/D v,v 1 MB 1 1 PADD/SUB(U)(S)B/W/D v,v 1 MB 1 1 PHADD(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						
MASKMOVQ mm,mm x,x 1-2 MASKMOVDQU x,x 1-2 PMOVMSKB r32,(x)mm 3 1 PEXTRW r32,(x)mm,i 1 MB 3 1 PEXTRB/D/Q r32/64,x,i 1 MB 3 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PMOVSX/ZXBW/BD/BQ/WD/WQ/DQ x,x 1 MB 1 1 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 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 PMULLD x,x 1						
MASKMOVDQU x,x r32,(x)mm 3 1 PEXTRW r32,(x)mm,i 1 MB 3 1 PEXTRB/D/Q r32/64,x,i 1 MB 3 1 PINSRW (x)mm,r32,i 2 MB 5 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PMOVSX/ZXBW/BD/BQ/WD/WQ/DQ x,x 1 MB 1 1 Arithmetic instructions V,v 1 MB 1 1 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 3 MB 3 3 PHADDD PHSUBD V,v 3 MB 3 3 PCMPEQ/GTB/W/D V,v 1 MB 1 1 PCMPEQQ x,x 1 MB 1 1 PMULHRSW V,v 1 MA			•	2	•	_
PMOVMSKB		· ·				
PEXTRW r32 , (x)mm,i 1 MB 3 1 PEXTRB/D/Q r32/64,x,i 1 MB 3 1 PINSRW (x)mm,r32,i 2 MB 5 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PMOVSX/ZXBW/BD/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 PADD/SUB(U)(S)B/W/D v,v 1 MB 1 1 1 PADDQ PSUBQ v,v 1 MB 1 1 1 PHADD(S)W v,v 3 MB 3 3 3 1	· ·				3	
PEXTRB/D/Q r32/64,x,i 1 MB 3 1 PINSRW (x)mm,r32,i 2 MB 5 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PMOVSX/ZXBW/BD/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 PADD/SUBQ PSUBQ v,v 1 MB 1 1 1 PHADD(S)W v,v 3 MB 3 3 3 PHADDD PHSUBD v,v 3 MB 3 3 PCMPEQ/GTB/W/D v,v 1 MB 1 1 1 PMULL/HW PMULHUW v,v 1 MB 1 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULLD x,x 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMULDQ x,x 1 MA 3		` ′	1	MR		
PINSRW (x)mm,r32,i 2 MB 5 1 PINSRB/D/Q x,r32/64,i 2 MB 5 1 PMOVSX/ZXBW/BD/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 1 PHADD(S)W v,v 3 MB 3 3 3 PHADDD PHSUBD v,v 3 MB 3 3 3 PCMPEQ/GTB/W/D v,v 1 MB 1 1 1 PCMPEQ/GTB/W/D v,v 1 MB 1 1 1 PCMPEQQ x,x 1 MB 1 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA <t< td=""><td></td><td>` ′</td><td></td><td></td><td></td><td></td></t<>		` ′				
PINSRB/D/Q x,r32/64,i 2 MB 5 1 PMOVSX/ZXBW/BD/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 PHADDUS(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 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ v,v 1 MA 3	· ·					
PMOVSX/ZXBW/BD/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 3 MB 3 3 PHADDD PHSUBD v,v 3 MB 3 3 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQ/GTB/W/D v,v 1 MB 1 1 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 PMULDQ x,x 1 MA 3 1 PMADDWD		` '				
BQ/WD/WQ/DQ x,x 1 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 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 PCMPEQQ TB/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 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 3 1 PMADDWBSW v,v 1 MB 2 1 MB <td>· ·</td> <td>X,132/04,1</td> <td></td> <td>IVID</td> <td>5</td> <td>l</td>	· ·	X,132/04,1		IVID	5	l
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 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 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQ/GTB/W/D v,v 1 MA 3 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULHRSW v,v 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PM		V V	1	MD	1	1
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 3 MB 3 3 PHADDD PHSUBD v,v 3 MB 3 3 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQQ x,x 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 PMULUDQ v,v 1 MA 3 1 PMADDWD v,v 1 MA 3 1 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1	DQ/WD/WQ/DQ	X,X	'	IVID	'	'
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 3 MB 3 3 PHADDD PHSUBD v,v 3 MB 3 3 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQQ x,x 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 PMULUDQ v,v 1 MA 3 1 PMADDWD v,v 1 MA 3 1 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1	Arithmetic instructions					
V,V 1 MB 1 1 PADDQ PSUBQ V,V 1 MB 1 1 PHADD(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 PCMPEQ/GTB/W/D V,V 1 MB 1 1 PCMPEQ/GTB/W/D V,V 1 MB 1 1 PCMPEQ/GTB/W/D V,V 1 MB 1 1 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 PMULLD X,X 1 MA 3 1 PMULUDQ X,X 1 MA 3 1 PMADDWD V,V 1 MA 4 2 PMAD						
PADDQ PSUBQ v,v 1 MB 1 1 PHADD(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 PCMPEQQ x,x 1 MB 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULHRSW v,v 1 MA 3 1 PMULUDQ x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMADDWD v,v 1 MA 3 1 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1	PADD/30B(U)(3)B/W/D	VV	1	MR	1	1
PHADD(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 PCMPEQQ x,x 1 MB 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULHRSW v,v 1 MA 3 1 PMULUD x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 3 1 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1	PADDO PSUBO					
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 PCMPEQQ x,x 1 MB 1 1 PCMPEQQ x,x 1 MB 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULHRSW v,v 1 MA 3 1 PMULUDD x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1	1	V, V	'	IVID	'	'
PHADDD PHSUBD v,v 3 MB 3 3 PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQQ x,x 1 MB 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULHRSW v,v 1 MA 3 1 PMULUD x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1		VV	3	MB	3	3
PCMPEQ/GTB/W/D v,v 1 MB 1 1 PCMPEQQ x,x 1 MB 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULHRSW v,v 1 MA 3 1 PMULLD x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1	` '	· ·				
PCMPEQQ x,x 1 MB 1 1 PMULL/HW PMULHUW v,v 1 MA 3 1 PMULHRSW v,v 1 MA 3 1 PMULLD x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1						
PMULL/HW PMULHUW v,v 1 MA 3 1 PMULHRSW v,v 1 MA 3 1 PMULLD x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1					-	
PMULHRSW v,v 1 MA 3 1 PMULLD x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1						
PMULLD x,x 1 MA 3 1 PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1						
PMULUDQ v,v 1 MA 3 1 PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1						
PMULDQ x,x 1 MA 3 1 PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1						
PMADDWD v,v 1 MA 4 2 PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1						
PMADDUBSW v,v 7 10 8 PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1	· ·					
PSADBW v,v 1 MB 2 1 MPSADBW x,x,i 1 MB 2 1				IVIA		
MPSADBW x,x,i 1 MB 2 1			-	MB		
		l .				

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	
PTEST	V,V	1	MB	3	1	
PSLL/RL/RAW/D/Q	V,V	1	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	

Floating point XMM instructions

	Operands	μops	Port	Latency	Reciprocal thruogh-	Remarks
Move instructions					put	
MOVAPS/D	x,x	1	MB	1	1 1	
MOVAPS/D	x,m128	1	LD	2	1 1	
MOVAPS/D	m128,x	1	SA ST	2	1 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 1	
MOVLPS/D	x,m64	2		6	1	
MOVHPS/D	m64,x	3		6	1-2	
MOVLPS/D	m64,x	1		2	1-2	
MOVLHPS MOVHLPS	x,x	1		1	1 1	
MOVMSKPS/D	r32,x			3	1	
MOVNTPS/D	m128,x	2		~360	1-2	
SHUFPS	x,x,i	1	MB	1	1	
SHUFPD	x,x,i	1	MB	1	1	
MOVDDUP	x,x	1	MB	1	1	
MOVSH/LDUP	x,x	1	MB	1	1	
UNPCKH/LPS	x,x	1	MB	1	1 1	
UNPCKH/LPD	x,x	1	MB	1	1	
Conversion						
CVTPD2PS	x,x	2		5	2	
CVTSD2SS	x,x	1		2		
CVTPS2PD	x,x	2		5	1 1	

0./T0000D		4	I	۰ .	1 1
CVTSS2SD	X,X	1	MD	2	_
CVTDQ2PS	X,X	1	MB	3	1 1
CVT(T) PS2DQ	X,X	1		2	1 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
CVTSI2SD	x,r32	2		5	
CVT(T)SD2SI	r32,x	1		4	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 X,X	1	MBfadd	2	1
ADDSUBPD	X,X X,X	'1	MBfadd	2	1
HADDPS HSUBPS		3	MBfadd	5	3
HADDPD HSUBPD	X,X	3	MBfadd	5	3
MULSS	X,X	1	MA	3	1
	X,X	1		4	1
MULSD	X,X	1	MA		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
	•	1	I .	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	
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