## 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 2016-01-09.

### 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.

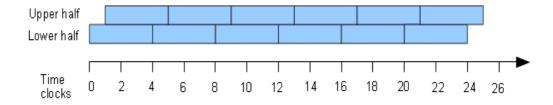
#### **Operands**

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  $\mu$ op is an abbreviation for micro-operation. Processors with out-of-order cores are capable of splitting complex instructions into  $\mu$ ops. For example, a read-modify instruction may be split into a read- $\mu$ op and a modify- $\mu$ op. The number of  $\mu$ ops that an instruction generates is important when certain bottlenecks in the pipeline limit the number of  $\mu$ 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  $\mu$ ops, for example floating point additions. The information about which execution unit a particular  $\mu$ op goes to can be useful for two purposes. Firstly, two  $\mu$ 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  $\mu$ op executing in one execution unit is needed as input for a  $\mu$ op in another execution unit.

# Execution port

The execution units are clustered around a few execution ports on most Intel processors. Each  $\mu$ 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  $\mu$ op at a time. Two  $\mu$ 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,  $\mu$ 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  $\mu$ 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  $\mu$ op can execute simultaneously with another instruction/ $\mu$ 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/ $\mu$ 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 $\neq$ 1.
80286	System instructions for 16-bit protected mode.
80386	The eight general purpose registers are extended from 16 to 32 bits. 32-bit addressing. 32-bit protected mode. Scaled index addressing. MOVZX, MOVSX, IMUL r,r, SHLD, SHRD, BT, BTR, BTS, BTC, BSF, BSR, SETcc. BSWAP. Later versions have CPUID.
80486	
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 Pentium	FPREM1, FSIN, FCOS, FSINCOS. 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.

thread environment blocks and processor environment blocks

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

#### 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, VPHADDWD, VPHADDWQ, VPHSUBBW, VPHSUBDQ, VPHSUBWD, VPMACSDD, VPMACSDQH, VPMACSDQL,

VPMACSSDD, VPMACSSDQH, VPMACSSDQL, VPMACSSWD,

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

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

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

# Microprocessor versions tested

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

Processor name	Microarchitecture Code name	Family number (hex)	Model number (hex)	Comment
AMD K7 Athlon		6	6	Step. 2, rev. A5
AMD K8 Opteron		F	5	Stepping A
AMD K10 Opteron		10	2	2350, step. 1
AMD Bulldozer	Bulldozer, Zambezi	15	1	FX-6100, step 2
AMD Piledriver	Piledriver	15	2	FX-8350, step 0. And others
AMD Steamroller	Steamroller, Kaveri	15	30	A10-7850K, step 1
AMD 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	Е	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
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

pipeline.

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

any of the three integer ALU's. ALU0\_1 means that ALU0 and ALU1 are both used. AGU means any of the three integer address generation units. FADD means floating point adder unit. FMUL means floating point multiplier unit. FMISC means floating point store and miscellaneous unit. FA/M means FADD or FMUL is used. FANY means any of the three floating point units can be used. Two macro-operations can execute simultaneously if they go to different

execution units.

### Integer instructions

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

MOV	sr,r/m	6	9-13	8		
MOVZX, MOVSX	r,r	1	1	1/3	ALU	
MOVZX, MOVSX	r,m	1	4	1/2	ALU, AGU	
CMOVcc	r,r	1	1	1/3	ALU	
CMOVcc	r,m	1		1/2	ALU, AGU	
XCHG	r,r	3	2	1	ALU ALU	
XCIIC	','	3		'	ALO	Timing depends
XCHG	r,m	3	16	16	ALU, AGU	on hw
XLAT		2	5		ALU, AGU	
PUSH	r	1		1	ALU, AGU	
PUSH	i	1		1	ALU, AGU	
PUSH	m l	2		1	ALU, AGU	
PUSH	sr	2		1	ALU, AGU	
PUSHF(D)	31	1		1	ALU, AGU	
PUSHA(D)		9		4	ALU, AGU	
POP		2		1	ALU, AGU	
	r	3				
POP	m DO/FO/FO/OC			1	ALU, AGU	
POP	DS/ES/FS/GS	6		10	ALU, AGU	
POP	SS	9		18	ALU, AGU	
POPF(D)		2		1	ALU, AGU	
POPA(D)		9		4	ALU, AGU	
LEA	r16,[m]	2	3	1	AGU	Any addr. size
LEA	r32,[m]	1	2	1/3	AGU	Any addr. size
LAHF		4	3	2	ALU	
SAHF		2	2	2	ALU	
SALC		1	1	1	ALU	
LDS, LES,	r,m	10		9		
BSWAP	r	1	1	1/3	ALU	
Arithmetic instructions						
ADD, SUB	r,r/i	1	1	1/3	ALU	
ADD, SUB	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 ALU	
CMP		1	<b>'</b>	1/3	ALU, AGU	
INC, DEC, NEG	r,m	1	4	1/2	ALU, AGU ALU	
INC, DEC, NEG	r	-	1 7			
	m	1	7	3	ALU, AGU	
AAA, AAS		9	5	5	ALU	
DAA		12	6	6	ALU	
DAS		16	7	7	ALU	
AAD		4	5		ALU0	
AAM		31	13	_	ALU	
MUL, IMUL	r8/m8	3	3	2	ALU0	
MUL, IMUL	r16/m16	3	3	2	ALU0_1	latency ax=3, dx=4
MUL, IMUL	r32/m32	3	4	3	ALU0_1 ALU0_1	
IMUL	r16,r16/m16	2	3	2	ALUU_1	
IIVIOL	110,110/11110	2	၂		ALUU	

T	I		ı	ı	
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	11102	1	1	1/3	ALU
CWD, CDQ		1	1	1/3	ALU
CVVD, CDQ		!	, I	1/3	ALU
Logic instructions					
AND, OR, XOR	r,r	1	1	1/3	ALU
AND, OR, XOR	r,m	1	1	1/2	ALU, AGU
AND, OR, XOR	m,r	1	7	2.5	ALU, AGU
TEST	r,r	1	1	1/3	ALU, AGU
TEST	•	1	1	1/3	
NOT	r,m	1	1		ALU, AGU ALU
	r			1/3	
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	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
ВТ	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
BTC	m,i	5	7	2	ALU, AGU
BTR, BTS	m,i	4	7	2	ALU, AGU
BTC, BTR, BTS	m,r	8	6	3	ALU, AGU
BSF		19	7	7	ALU, AGU ALU
	r,r				
BSR	r,r	23	9	9	ALU

boe	l		•	1 0		1
BSF	r,m	20	8	8	ALU, AGU	
BSR	r,m	23	10	10	ALU, AGU	
SETcc	r	1	1	1/3	ALU	
SETcc	m	1		1/2	ALU, AGU	
CLC, STC		1		1/3	ALU	
CMC		1	1	1/3	ALU	
CLD		2		1	ALU	
STD		3		2	ALU	
Control transfer instruction	ons					
JMP	short/near	1 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	
OALL	near				ALO	low values = real
CALL	far	16-22	23-32			mode
CALL	r	4	3	3	ALU	
CALL		5	3	3	ALU, AGU	
CALL	m(near)	5	J	3	ALU, AGU	low values = real
CALL	m(far)	16-22	24-33			mode
RETN	iii(iai)	2	3	3	ALU	mode
RETN	i	2	3	3	ALU	
REIN	<b>'</b>		3	3	ALU	low values = real
RETF		15-23	24-35			low values = real mode
RETF	i	15-24	24-35			low values = real mode
IRET		32	81			real mode
INT	i	33	42			real mode
			'-			values are for no
BOUND	m	6		2		jump
INTO		2		2		values are for no jump
		_		_		
String instructions						
LODS		4	2	2		
REP LODS		5	2	2		values per count
STOS		4	2	2		
REP STOS		3	1	1		values per count
MOVS		7	3	3		.
REP MOVS		4	1-4	1-4		values per count
SCAS		5	2	2		
REP SCAS		5	2	2		values per count
CMPS		7	6	6		
REP CMPS		6	3-4	3-4		values per count
	I		J 1	, ,	I	, and of poi oddint

Other						
NOP (90)	1	0	1/3	ALU		
Long NOP (0F 1F)	1	0	1/3	ALU		
ENTER	i,0	12	12	12		
					3 ops, 5 clk if 16	
LEAVE	3		3		bit	
CLI	8-9		5			
STI	16-17		27			
CPUID	19-28	44-74				
RDTSC	5		11			
RDPMC	9		11			

Floating point x87 instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
Move instructions						
FLD	r	1	2	1/2	FA/M	
FLD	m32/64	1	4	1/2	FANY	
FLD	m80	7	16	4		
FBLD	m80	30	41	39		
FST(P)	r	1	2	1/2	FA/M	
FST(P)	m32/64	1	3	1	FMISC	
FSTP	m80	10	7	5		
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	<b>'</b>		0	1/3	FANY	
I INCOTT, I DECOTT		'		1/3	I ANI	Low latency im-
						mediately after
FNSTSW	AX	2	6-12	12	FMISC, ALU	FCOM FTST
FSTSW	AX	3	6-12	12	FMISC, ALU	do.
FNSTSW	m16	2		8	FMISC, ALU	do.
FNSTCW	m16	3		1	FMISC, ALU	
						faster if
FLDCW	m16	14		42	FMISC, ALU	unchanged
Arithmetic instructions						
FADD(P),FSUB(R)(P)	r/m	1	4	1	FADD	
FIADD,FISUB(R)	m	2	4	1-2	FADD,FMISC	
FMUL(P)	r/m	1	4	1	FMUL	
FIMUL	m	2	4	2	FMUL,FMISC	
EDIV/DVD)	m/ma	4	11 05	0.00		Low values are for round divisors
FDIV(R)(P)	r/m	1	11-25	8-22	FMUL	ioi iouiiu 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 PSHUFW MASKMOVQ PMOVMSKB PEXTRW PINSRW	mm,r/m mm,mm,i mm,mm r32,mm r32,mm,i mm,r32,i	1 1 32 3 2 2	2 2 5 12	2 1/2 24 3 2	FA/M FA/M FADD FMISC, ALU FA/M	
Arithmetic instructions	,. 52,1	_		_	.,,,,,,	
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	
Other						
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)**

obition monactions (observe)									
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			

mm,mm	1	2	1/2	FA/M	
mm,mm	1	3	1	FMUL	
ıs					
mm,mm	1	4	1	FADD	
mm,mm	1	2	1	FADD	
mm,mm	1	2	1	FADD	
mm,mm	1	4	1	FMUL	
mm,mm	1	4	1	FADD	
mm,mm	1	4	1	FADD	3DNow E
mm,mm	1	3	1	FMUL	
mm,mm	1	4	1	FMUL	
mm,mm	1	3	1	FMUL	
mm,mm	1	4	1	FMUL	
mm mm	1		1/3	FANY	
	mm,mm	mm,mm 1	mm,mm 1 3  mm,mm 1 4  mm,mm 1 2  mm,mm 1 2  mm,mm 1 4  mm,mm 1 4  mm,mm 1 4  mm,mm 1 3  mm,mm 1 3  mm,mm 1 4  mm,mm 1 4  mm,mm 1 3  mm,mm 1 4  mm,mm 1 4	mm,mm     1     3     1       ms     1     4     1       mm,mm     1     2     1       mm,mm     1     2     1       mm,mm     1     4     1       mm,mm     1     4     1       mm,mm     1     4     1       mm,mm     1     3     1       mm,mm     1     3     1       mm,mm     1     3     1       mm,mm     1     3     1       mm,mm     1     4     1       mm,mm     1     4     1	mm,mm

## 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 throughput	Execution unit	Notes
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	ÁLU	
MOVSXD	r64,m32	1	•	1/2	ALU, AGU	
CMOVcc	r,r	1	1	1/3	ALU	
CMOVcc		1	'	1/3	ALU, AGU	
	r,m		2			
XCHG	r,r	3	2	1	ALU	
		_				Timing depends on
XCHG	r,m	3	16	16	ALU, AGU	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	
	33					
POPF(D/Q)		25	10	10	ALU, AGU	
POPA(D)	40.5.3	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		4	3	2	ALU	
SAHF		1	1	1/3	ALU	
SALC		1	1	1/3	ALU	
LDS, LES,	r,m	10		9		
BSWAP	r	1	1	1/3	ALU	
PREFETCHNTA	m	1		1/2	AGU	
PREFETCHT0/1/2	m	1		1/2	AGU	
SFENCE		6		8	7.00	
LFENCE		1		5		
MFENCE		7		16		
IN	r i/DV	270		10		
	r,i/DX					
OUT	i/DX,r	300				
A 141 41 4 41						
Arithmetic instruction		4	4	4/2	A111	
ADD, SUB	r,r/i	1	1	1/3	ALU	
ADD, SUB	r,m	1	1	1/2	ALU, AGU	
ADD, SUB	m,r	1	7	2.5	ALU, AGU	
ADC, SBB	r,r/i	1	1	1/3	ALU	
ADC, SBB	r,m	1	1	1/2	ALU, AGU	
ADC, SBB	m,r/i	1	7	2.5	ALU, AGU	
CMP	r,r/i	1	1	1/3	ALU	
CMP	r,m	1		1/2	ALU, AGU	
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	<b>'</b>	ALU0	
אאט		4	ວ		ALUU	

Δ Δ Δ Δ	1	24	40	I		I I
AAM	#0/ma0	31	13	4	ALU	
MUL, IMUL	r8/m8	1	3	1	ALU0	
MUL, IMUL	r16/m16	3	3-4	2	ALU0_1	latency ax=3, dx=4
MUL, IMUL	r32/m32	2	3	1	ALU0_1	
MUL, IMUL	r64/m64	2	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			15		_	
II.	r8/m8	31	15	15	ALU	
DIV	r16/m16	46	23	23	ALU	
DIV	r32/m32	78	39	39	ALU	
DIV	r64/m64	143	71	71	ALU	
IDIV	r8	40	17	17	ALU	
IDIV	r16	55	25	25	ALU	
IDIV	r32	87	41	41	ALU	
IDIV	r64	152	73	73	ALU	
IDIV	m8	41	17	17	ALU	
IDIV	m16	56	25	25	ALU	
IDIV	m32	88	41	41	ALU	
IDIV	m64	153	73	73	ALU	
CBW, CWDE, CDQE		1	1	1/3	ALU	
CWD, CDQ, CQO		1	1	1/3	ALU	
OVVD, ODQ, OQO		'	'	1,0	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		1	7	2.5	ALU, AGU	
TEST	m,r	1	1	1/3	ALU, AGU ALU	
	r,r		_			
TEST	r,m	1	1	1/2	ALU, AGU	
NOT	r	1	1	1/3	ALU	
NOT	m	1	7	2.5	ALU, AGU	
SHL, SHR, SAR	r,i/CL	1	1	1/3	ALU	
ROL, ROR	r,i/CL	1	1	1/3	ALU	
RCL, RCR	r,1	1	1	1/3	ALU	
RCL	r,i	9	3	3	ALU	
RCR	r,i	7	3	3	ALU	
RCL	r,CL	9	4	4	ALU	
RCR	r,CL	7	3	3	ALU	
SHL,SHR,SAR,ROL,R	·					
OR	m,i /CL	1	7	3	ALU, AGU	
RCL, RCR	m,1	1	7	4	ALU, AGU	
RCL	m,i	10	9	4	ALU, AGU	
RCR	m,i	9	8	4	ALU, AGU	
RCL	m,CL	9	7	4	ALU, AGU	
RCR	m,CL	8	8	3	ALU, AGU	
SHLD, SHRD	r,r,i	6	3	3	ALU	
SHLD, SHRD	r,r,cl	7	3	3	ALU	

SHLD, SHRD 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 8 21 22 28 20 22 25 28 1 1 1 1 2	6 1 2 7 5 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 ALU ALU ALU ALU ALU ALU, AGU ALU ALU ALU ALU	
Control transfer instru	short/near	1		2	ALU	
OIVII	SHULVILEAL	'			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	4/0 -		mode
Jcc	short/near	1		1/3 - 2	ALU	recip. thrp.= 2 if jump
J(E/R)CXZ	short	2	<b>6</b> 4	1/3 - 2	ALU	recip. thrp.= 2 if jump
LOOP	short	7	3-4	3-4	ALU	
CALL	near	3	2	2	ALU	
CALL	for	16 22	22.22			low values = real
CALL CALL	far	16-22 4	23-32	3	ALU	mode
CALL	r m(near)	5	3 3	3	ALU, AGU	
UALL	m(near)	ן ט	S	3	ALU, AGU	low volues = reel
CALL	m(far)	16-22	24-33			low values = real mode
RETN	111(101)	2	3	3	ALU	IIIOGC
RETN	i	2	3	3	ALU	
		_	J		,	low values = real
RETF		15-23	24-35			mode
						low values = real
RETF	i	15-24	24-35			mode
IRET		32	81			real mode
INT	i	33	42			real mode
BOUND	m	6		2		values are for no jump
INTO		2		2		values are for no jump
String instructions						

LODS	4	2	2		
REP LODS	5	2	2		values are per count
STOS	4	2	2		
REP STOS	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	
FCOM(P), FUCOM(P)	r/m	1	2	1	FADD	
FCOMPP, FUCOMPP		1	2	1	FADD	
FCOMI(P)	r	1	3	1	FADD	
FICOM(P)	m	2		1	FADD, FMISC	
FTST		1	2	1	FADD	
FXAM		2	_	1	FMISC, ALU	
FRNDINT		5	10	3	,	
FPREM		1	7-10	8	FMUL	
FPREM1		1	8-11	8	FMUL	
Math						
FSQRT		1	27	12	FMUL	
FLDPI, etc.		1		1	FMISC	
FSIN		66	140-190			
FCOS		73	150-190			
FSINCOS		98	170-200			
FPTAN		67	150-180			
FPATAN		97	217			
FSCALE		5	8			
FXTRACT		7	12	7		
F2XM1		53	126			
FYL2X		72	179			
FYL2XP1		75	175			
Other						
FNOP		1	0	1/3	FANY	
(F)WAIT		1	0	1/3	ALU	
FNCLEX		8		27	FMISC	
FNINIT		26		100	FMISC	
FNSAVE		77		171		
FRSTOR		70		136		
FXSAVE		61		56		
FXRSTOR		101		95		

## Integer MMX and XMM instructions

nteger with and Alith instructions								
Instruction	Operands	Ops	Latency	Reciprocal throughput		Notes		
Move instructions	·							
MOVD	r32, mm	2	4	2	FMICS, ALU			
MOVD	mm, r32	2	9	2	FANY, ALU			
MOVD	mm,m32	1		1/2	FANY			
MOVD	r32, xmm	3	2	2	FMISC, ALU			
MOVD	xmm, r32	3	3	2				
MOVD	xmm,m32	2		1	FANY			
MOVD	m32, r	1		1	FMISC			

1	1 1		I.	- 	I	l
MAON (D. (MAON (O.)	.04	•			ENAIGO ALLI	Moves 64 bits.Name of instruction differs
MOVD (MOVQ)	r64,mm/xmm	2 2	4	2 2	FMISC, ALU	
MOVD (MOVQ)	mm,r64		9		FANY, ALU	do.
MOVD (MOVQ)	xmm,r64	3	9	2	FANY, ALU	do.
MOVQ	mm,mm	1	2 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	mm,r/m	1	2	1/2	FA/M	
PADDB/W/D/Q PADDSB/W ADDUSB/W PSUBB/W/D/Q PSUBSB/W PSUBUSB/W						
	xmm,r/m	2	2	1	FA/M	
PCMPEQ/GT B/W/D	mm,r/m	1	2	1/2	FA/M	
PCMPEQ/GT B/W/D	xmm,r/m	2	2	1	FA/M	

PMULLW PMULHW PMULHUW 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						
MOVAPS/D	r,r	2	2	1	FA/M	
MOVAPS/D	r,m	2		2	FMISC	
MOVAPS/D	m,r	2		2	FMISC	
MOVUPS/D	r,r	2	2	1	FA/M	
MOVUPS/D	r,m	4		2		
MOVUPS/D	m,r	5		2		
MOVSS/D	r,r	1	2	1	FA/M	
MOVSS/D	r,m	2	4	1	FANY FMISC	
MOVSS/D	m,r	1	3	1	FMISC	
MOVHLPS,						
MOVLHPS	r,r	1	2	1/2	FA/M	
MOVHPS/D,						
MOVLPS/D	r,m	1		1	FMISC	
MOVHPS/D,						
MOVLPS/D	m,r	1		1	FMISC	
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	
UNPCK H/L PS/D	r,r/m	2	3	3	FMUL	
5111 5111/21 0/B	,,,,,,	_				
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	
CVTDQ2FD CVT(T)PS2DQ	r,r/m	2	5	2	FMISC	
CVT(T)PD2DQ	r,r/m	4	8	3	FMISC	
CVT(T)PD2DQ CVTPI2PS	*	1	4	1	FMISC	
CVTPI2PD	xmm,mm	2		2		
	xmm,mm		5		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	
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						
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	
To the state of th		1	1	I	-	T. T.

RSQRTPS	r,r/m	2	3	2	FMUL	
Other						
LDMXCSR	m	8		9		
STMXCSR	m	3		10		

## List of instruction timings and macro-operation breakdown

Explanation of column headings:

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

JNE, etc.

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

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

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

structions with more than 2 macro-operations use microcode.

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

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

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

Reciprocal through-

put:

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

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

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

of the three integer ALU's. ALU0\_1 means that ALU0 and ALU1 are both used. AGU means any of the three integer address generation units. FADD means floating point adder unit. FMUL means floating point multiplier unit. FMISC means floating point store and miscellaneous unit. FA/M means FADD or FMUL is used. FANY means any of the three floating point units can be used. Two macro-opera-

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

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution unit	Notes
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 PUSH(D/Q) PUSHA(D) POP POP POP POP POP POP POPE POP POPA(D) LEA LEA	r,m r64,r32 r64,m32 r,r r,m r,r r,m 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	4 1 4 1 21 5 6 3 10 26 16 6 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	ALU, AGU	Timing depends on hw  Any address size ≤ 2 source operands
LEA	r32/64,[m]	1 1	2	1/3	AGU	W. scale or 3 opr.
LAHF SAHF		4	3 1	2 1/3	ALU ALU	
SALC			1	1/3	ALU	
LDS, LES,	r,m	10	'	10	ALO	
BSWAP	r	1	1	1/3	ALU	
PREFETCHNTA	m	1	•	1/2	AGU	
PREFETCHT0/1/2	m	1		1/2	AGU	
PREFETCH(W)	m	1		1/2	AGU	3DNow
SFENCE		6		8		
LFENCE		1		1		
MFENCE		4		33		
IN	r,i/DX	~270				
OUT	i/DX,r	~300				
Arithmetic instruction	_					
ADD, SUB	r,r/i	1	1	1/3	ALU	
ADD, SUB	r,m	1	-	1/2	ALU, AGU	
ADD, SUB	m,r	1	4	1	ALU, AGU	
ADC, SBB	r,r/i	1	1	1/3	ÁLU	
ADC, SBB	r,m	1		1/2	ALU, AGU	
ADC, SBB	m,r/i	1	4	1	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	2	ALU, AGU	
AAA, AAS DAA		9 12	5 6	5	ALU	
DAS		16	6 7	6 7	ALU ALU	
AAD		4	, 5	<i>7</i> 5	ALU0 ALU0	
AAM		30	13	13	ALU	

MUL, IMUL         r8/m8         1         3         1         ALU0           MUL, IMUL         r16/m16         3         3         2         ALU0_1         latency ax=3, dx=4           MUL, IMUL         r32/m32         2         3         1         ALU0_1         latency ax=3, dx=4           MUL, IMUL         r64/m64         2         4         2         ALU0_1         latency rax=4, rdx=5           IMUL         r16,r16/m16         1         3         1         ALU0         latency rax=4, rdx=5           IMUL         r16,r16/m16         1         3         1         ALU0         latency rax=4, rdx=5           IMUL         r64,r64/m64         1         4         2         ALU0_1         latency rax=4, rdx=5           IMUL         r16,(r16),i         2         4         1         ALU0         latency rax=4, rdx=5           IMUL         r16,(r64),i         1         4         2         ALU0_1         ALU0           IMUL         r16,(r64),i         1         4         2         ALU0         ALU0           IMUL         r32,m32,i         3         2         ALU0_1         ALU           IDIV         r8         19         19	41.11 18.41.11	1 0/ 0		۱ ۵	1		I
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         latency rax=4, rdx=5           IMUL         r16,r16/m16         1         3         1         ALU0         latency rax=4, rdx=5           IMUL         r32,r32/m32         1         3         1         ALU0         latency rax=4, rdx=5           IMUL         r64,r64/m64         1         4         2         ALU0_1         latency rax=4, rdx=5           IMUL         r64,r64/m64         1         4         2         ALU0_1         latency rax=4, rdx=5           IMUL         r64,r64/m64         1         4         2         ALU0_1         latency rax=4, rdx=5           IMUL         r16,(r16),i         2         4         1         ALU0         latency rax=4, rdx=5           IMUL         r16,(r16),i         2         4         1         ALU0         latency rax=4, rdx=5           IMUL         r16,(r16),i         1         4         2         ALU0         latency rax=4, rdx=5		r8/m8	1	3	1	ALU0	
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           IMUL         r16,m16,i         3         2         ALU0           IMUL         r32,m32,i         3         2         ALU0           IMUL         r64,m64,i         3         2         ALU0_1           IDIV         r8/m8         17         17         ALU           IDIV         r8         19         19         ALU           DIV         r16/m16         15-30         15-30         ALU         Depends on number of significant bits in absolute value of dividend. See AMD software optimization guide.           IDIV         r16/m16         24						_	latency ax=3, dx=4
IMUL		r32/m32				_	
IMUL	лul, imul	r64/m64	2	4	2	ALU0_1	latency rax=4, rdx=5
IMUL	MUL	r16,r16/m16	1	3	1	ALU0	
IMUL	MUL	r32,r32/m32	1	3	1	ALU0	
IMUL	MUL	· ·	1	4	2	ALU0 1	
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_1           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 of significant bits in absolute value of dividend. See AMD software optimization guide.           IDIV         r16/m16         24-39         24-39         ALU         ALU           IDIV         r16/m16         24-39         24-39         ALU         ALU           IDIV         r32/m32         24-55         24-55         ALU           IDIV         r64/m64         24-87         24-87         ALU           CBW, CWDE, CDQE         1         1         1         1/3         ALU			2			_	
IMUL         r64,(r64),i         1         4         2         ALU0           IMUL         r16,m16,i         3         2         ALU0           IMUL         r32,m32,i         3         2         ALU0           IMUL         r64,m64,i         3         2         ALU0_1           DIV         r8/m8         17         17         ALU           IDIV         r8         19         19         ALU           IDIV         r16/m16         15-30         15-30         ALU         Depends on number of significant bits in absolute value of dividend. See AMD software optimization guide.           IDIV         r16/m16         24-39         24-39         ALU and and an approximate ap		, ,			·		
IMUL         r16,m16,i         3         2         ALU0           IMUL         r32,m32,i         3         2         ALU0           IMUL         r64,m64,i         3         2         ALU0_1           DIV         r8/m8         17         17         ALU           IDIV         r8         19         19         ALU           IDIV         r16/m16         15-30         15-30         ALU         Depends on number of significant bits in absolute value of dividend. See AMD software optimization guide.           IDIV         r16/m16         24-39         24-39         ALU absolute value of dividend. See AMD software optimization guide.           IDIV         r32/m32         24-55         24-55         ALU absolute value of dividend. See AMD software optimization guide.           IDIV         r64/m64         24-87         24-87         ALU           CBW, CWDE, CDQE         1         1         1/3         ALU		, ,	-		<u>-</u>		
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         r16/m16         15-30         15-30         ALU         Depends on number of significant bits in absolute value of dividend. See AMD software optimization guide.           IDIV         r16/m16         24-39         24-39         ALU absolute value of dividend. See AMD software optimization guide.           IDIV         r32/m32         24-87         24-87         ALU         ALU           IDIV         r64/m64         24-87         24-87         ALU         Ware optimization guide.           IDIV         r64/m64         24-87         24-87         ALU         ALU           CBW, CWDE, CDQE         1         1         1/3         ALU         ALU		, ,		7			
IMUL         r64,m64,i         3         2         ALU0_1           DIV         r8/m8         17         17         ALU           IDIV         r8         19         19         ALU           IDIV         r16/m16         15-30         15-30         ALU         Depends on number of significant bits in absolute value of dividend. See AMD software optimization guide.           DIV         r64/m64         15-78         15-78         ALU absolute value of dividend. See AMD software optimization guide.           IDIV         r32/m32         24-39         24-39         ALU absolute value of dividend. See AMD software optimization guide.           IDIV         r64/m64         24-87         24-87         ALU absolute value of dividend. See AMD software optimization guide.           CBW, CWDE, CDQE         1         1         1/3         ALU							
DIV         r8/m8         17         17         ALU           IDIV         r8         19         19         ALU           IDIV         m8         22         22         ALU           DIV         r16/m16         15-30         15-30         ALU         Depends on number           DIV         r32/m32         15-46         15-46         ALU         of significant bits in           DIV         r64/m64         15-78         15-78         ALU         absolute value of dividend. See AMD software optimization           IDIV         r32/m32         24-39         24-39         ALU         ware optimization           IDIV         r64/m64         24-87         24-87         ALU         guide.           CBW, CWDE, CDQE         1         1         1/3         ALU         ALU							
IDIV   r8   r8   r8   r19   r8   r19   r19   r19   r16/m16   r16/m16   r32/m32   r15-46   r15-46   r15-78   r15-78   r15-78   r15-78   r16/m16   r16/m16			3			_	
IDIV   m8							
DIV         r16/m16         15-30         15-30         ALU         Depends on number of significant bits in absolute value of dividend. See AMD software optimization guide.           DIV         r64/m64         15-46         15-46         ALU         Depends on number of significant bits in absolute value of dividend. See AMD software optimization guide.           IDIV         r32/m32         24-39         24-39         ALU         ALU         ware optimization guide.           IDIV         r64/m64         24-87         24-87         ALU         guide.           CBW, CWDE, CDQE         1         1         1/3         ALU         ALU							
DIV         r32/m32         15-46         15-46         ALU absolute value of dividend. See AMD software optimization guide.           DIV         r64/m64         15-78         15-78         ALU absolute value of dividend. See AMD software optimization guide.           IDIV         r32/m32         24-39         24-39         ALU absolute value of dividend. See AMD software optimization guide.           IDIV         r64/m64         24-87         24-87         ALU absolute value of dividend. See AMD software optimization guide.           CBW, CWDE, CDQE         1         1         1/3         ALU	DIV	m8		22	22	ALU	
DIV         r64/m64         15-78         15-78         ALU absolute value of dividend. See AMD software optimization guide.           IDIV         r32/m32 IDIV         24-39 24-39 24-55 24-55 ALU absolute value of dividend. See AMD software optimization guide.           CBW, CWDE, CDQE         1         1         1/3         ALU	OIV	r16/m16		15-30	15-30	ALU	
IDIV	OIV	r32/m32		15-46	15-46	ALU	
IDIV	OIV	r64/m64		15-78	15-78	ALU	
IDIV	DIV	r16/m16		24-39	24-39	ALU	
IDIV r64/m64 24-87 24-87 ALU guide. CBW, CWDE, CDQE 1 1 1 1/3 ALU							
CBW, CWDE, CDQE 1 1 1/3 ALU							guide.
		104/11104	1				
CWD, CDQ, CQO							
	JVVD, CDQ, CQO		ı	I I	1/3	ALU	
Logio instructions	agia inatruationa						
AND, OR, XOR r,r 1 1 1/3 ALU		rr	1	4	1/2	ALLI	
				I I			
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			-	1			
TEST r,m 1 1/2 ALU, AGU		r,m					
NOT r 1 1 1/3 ALU		r	1				
NOT m 1 7 1 ALU, AGU	TOV	m	1	7	1	ALU, AGU	
SHL, SHR, SAR r,i/CL 1 1 1/3 ALU		r,i/CL	1	1	1/3	ALU	
ROL, ROR r,i/CL 1 1 1/3 ALU	ROL, ROR	r,i/CL	1	1	1/3	ALU	
RCL, RCR r,1 1 1 1 ALU	RCL, RCR	r,1	1	1	1	ALU	
RCL r,i 9 3 3 ALU		r,i	9	3	3	ALU	
RCR r,i 7 3 3 ALU	RCR		7	3	3	ALU	
RCL r,CL 9 4 4 ALU			9	4	4		
RCR r,CL 7 3 3 ALU							
SHL,SHR,SAR,ROL,R( m,i /CL 1 7 1 ALU, AGU							
RCL, RCR m,1 1 7 1 ALU, AGU					·		
RCL m,i 10 7 5 ALU, AGU			-				
		·					
RCL m,CL 9 8 6 ALU, AGU							
RCR m,CL 8 7 5 ALU, AGU							
SHLD, SHRD r,r,i 6 3 2 ALU							
SHLD, SHRD r,r,cl 7 3 3 ALU							
SHLD, SHRD m,r,i/CL 8 7.5 6 ALU, AGU			8				
BT r,r/i 1 1 1/3 ALU		r,r/i	1	1	1/3	ALU	
BT m,i 1 1/2 ALU, AGU	3T	m,i	1		1/2	ALU, AGU	
BT m,r 5 7 2 ALU, AGU	3T	m,r	5	7	2	ALU, AGU	
BTC, BTR, BTS r,r/i 2 2 1/3 ALU	3TC, 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 7 8 1 1 1	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 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	ıctions					
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			low values = real mode
RETN	_	2	3	3	ALU	
RETN	i	2	3	3	ALU	
RETF		15-23	24-35			low values = real mode
RETF	ı	15-24	24-35			low values = real mode
IRET	:	32	81			real mode
INT BOUND	i	33	42	2		real mode
INTO	m	6 2		2 2		values are for no jump
INTO						values are for no jump
String instructions						
LODS		4	2	2		
REP LODS		5	2	2		values are per count
STOS		4	2	2		Tanada ana panadani
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	1	16-17		27		
CPUID	2	22-50	47-164			
RDTSC		30		67		
RDPMC		13		5		

Floating point x87 instructions

Instruction	Floating point x87						
Move instructions	Instruction	Operands	Ops	Latency			Notes
FLD         r         1         2         1/2         FA/M         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         1/2         FA/M           FST(P)         m32/64         1         2         1/2         FA/M           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         1         5         FMISC           FCMOVcc         st0,r         9         FMISC, FA/M         FANY           FINCSTP, FDECSTP         1         0         1/3         FANY					throughput	unit	
FLD         m32/64   1 m80         1 m80         7 m80         13 m80         4 m80         7 m80         30 m80         20 m80         94 m80         30 m80         FANY           FST(P)         r         1 2 m80         10 m80         20 m80         10 m80	Move instructions						
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         FMISC           FSTP         m80         218         167         163         FMISC           FXCH         r         1         0         1/3         FMISC           FILD         m         1         6         1         FMISC           FILDZ, FLD1         1         1         1         FMISC           FCMOVcc         st0,r         9         FMISC, FA/M         FANY           FINCST(P)         1         0         1/3         FANY           FINCSTP, FDECSTP         1         0         1/3         FANY           FNSTSW         AX         2         16         FMISC, ALU         Low latency immediately after FCOMI           FNSTSW         AX         2         9         FMISC, ALU         Low latency immediately after FCOMI           FNSTSW         AX </td <td>FLD</td> <td>r</td> <td>1</td> <td>2</td> <td>1/2</td> <td>FA/M</td> <td></td>	FLD	r	1	2	1/2	FA/M	
FBLD	FLD	m32/64	1	4	1/2	FANY	
FST(P)	FLD	m80	7	13	4		
FST(P)	FBLD	m80	20	94	30		
FSTP	FST(P)	r	1	2	1/2	FA/M	
FBSTP	FST(P)	m32/64	1	2	1	FMISC	
FXCH         r         1         0         1/3 result to the part of the property of the product	FSTP	m80	10	8	7		
FXCH         r         1         0         1/3 result to the part of the property of the product	FBSTP	m80	218	167	163		
FILD         m         1         6         1         FMISC           FIST(P)         m         1         4         1         FMISC           FLDZ, FLD1         1         4         1         FMISC           FCMOVcc         st0,r         9         FMISC, FA/M         Low latency immediately ately after FCOMI           FFREE         r         1         0         1/3         FANY           FINSTSW         AX         2         16         FMISC, ALU FANY           FNSTSW         AX         3         14         FMISC, ALU FMISC, ALU FMISC, ALU GO.           FNSTSW         m16         2         9         FMISC, ALU GO.         do.           FNSTCW         m16         3         2         FMISC, ALU GO.         faster if unchanged           FLDCW         m16         12         14         FMISC, ALU FMISC, ALU GO.         faster if unchanged           Arithmetic instructions         T/m         1         4         1         FADD, FMISC           FMUL(P)         r/m         1         4         1         FMUL, FMISC           FMUL(P)         r/m         1         4         FMUL, FMISC           FABS, FCHS         1 <td< td=""><td></td><td>r</td><td></td><td>0</td><td></td><td></td><td></td></td<>		r		0			
FIST(P)		m	1			FMISC	
FLDZ, FLD1         1         1         FMISC         Low latency immediately after FCOMI           FCMOVcc         st0,r         9         FMISC, FA/M         FMISC, FA/M         Low latency immediately after FCOMI           FFREE         r         1         0         1/3         FANY         Low latency immediately after FCOM FTST           FINSTSW         AX         2         16         FMISC, ALU         Low latency immediately after FCOM FTST           FSTSW         AX         3         14         FMISC, ALU         Low latency immediately after FCOM FTST           FSTSW         AX         3         14         FMISC, ALU         do.           FNSTSW         m16         2         9         FMISC, ALU         do.           FNSTSW         m16         3         2         FMISC, ALU         do.           FNSTSW         m16         12         14         FMISC, ALU         faster if unchanged           Arithmetic instructions           FADD(P),FSUB(R)(P)         r/m         1         4         FADD,FMISC           FMUL(P)         r/m         1         4         1         FMUL,FMISC           FMUL(P)         r/m         1         2         4         FMUL		m	1		1	FMISC	
FCMOVcc	` '						
FCMOVcc	,						I ow latency immedi-
FFREE         r         1         0         1/3         FANY           FINCSTP, FDECSTP         1         0         1/3         FANY           FNSTSW         AX         2         16         FMISC, ALU FMISC, ALU FMISC, ALU FMISC, ALU FMISC, ALU Go.           FNSTSW         m16         2         9         FMISC, ALU FMISC, ALU Go.         do.           FNSTCW         m16         3         2         FMISC, ALU	FCMOVcc	st0.r	9			FMISC, FA/M	
NSTSW					1/3	1	,
NSTSW				0			
FNSTSW							Low latency immediately
FSTSW	FNSTSW	AX	2		16	FMISC, ALU	
FNSTSW         m16         2         9         FMISC, ALU         do.           FNSTCW         m16         3         2         FMISC, ALU         faster if unchanged           FLDCW         m16         12         14         FMISC, ALU         faster if unchanged           Arithmetic instructions           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,FMISC           FABS, FCHS         1         2         2         FMUL           FCOM(P), FUCOM(P)         r/m         1         1         FADD           FCOM(P), FUCOM(P)         r         1         1         FADD           FCOMI(P)         m         2         1         FADD, FMISC           FTST         1         1         FADD         FADD						· ·	do.
National Process		m16					
Arithmetic instructions					1		
Arithmetic instructions						· ·	faster if unchanged
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,FMISC           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						,	
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,FMISC           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, FMISC           FIST         1         1         FADD, FMISC	Arithmetic instruction	is					
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	FADD(P),FSUB(R)(P)	r/m	1	4	1	FADD	
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	FIADD,FISUB(R)	m	2		4	FADD,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	FMUL(P)	r/m	1	4	1	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	FIMUL	m	2		4	FMUL,FMISC	
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	FDIV(R)(P)	r/m	1	?	24	FMUL	
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       TADD		m	2	31	24	FMUL,FMISC	
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         TADD	• •		1	2	2	FMUL	
FCOMPP, FUCOMPP         1         1         FADD           FCOMI(P)         r         1         1         FADD           FICOM(P)         m         2         1         FADD, FMISC           FTST         1         1         FADD		r/m	1		1	FADD	
FCOMI(P)         r         1         1         FADD           FICOM(P)         m         2         1         FADD, FMISC           FTST         1         1         FADD	` ,		1		1		
FICOM(P) m 2 1 FADD, FMISC FTST 1 1 FADD		r	1		1		
FTST 1 1 FADD	• •	m	2		1		
					1		
FAAIVI   Z   1   FMISC, ALU	FXAM		2		1	FMISC, ALU	
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		

l	1	_		l <u>-</u>	1 1	
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/	111111,17111	•	_	172	I /VIVI	
DQ	xmm,r/m	1	3	1/2	FA/M	
			3		FA/M	
PUNPCKHQDQ	xmm,r/m	1	1	1/2		
PUNPCKLQDQ	xmm,r/m	1	3	1/2	FA/M	
PSHUFD	xmm,xmm,i	1	3	1/2	FA/M	
PSHUFW	mm,mm,i	1	2	1/2	FA/M	
PSHUFL/HW	xmm,xmm,i	1	2	1/2	FA/M	
MASKMOVQ	mm,mm	32		13		
MASKMOVDQU	xmm,xmm	64		24		
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						
PMULHUW						
PMULUDQ	mm/xmm,r/m	1	3	1	FMUL	
PMADDWD	mm/xmm,r/m	1	3	1	FMUL	
PAVGB/W	mm/xmm,r/m	1	2	1/2	FA/M	
PMIN/MAX SW/UB	mm/xmm,r/m	1	2	1/2	FA/M	
PSADBW	mm/xmm,r/m	1	3	1	FADD	
Logic						
PAND PANDN POR	1					
PXOR	mm/xmm,r/m	1	2	1/2	FA/M	
PSLL/RL W/D/Q						
PSRAW/D	mm,i/mm/m	1	2	1/2	FA/M	
PSLL/RL W/D/Q						
PSRAW/D	x,i/(x)mm	1	3	1/2	FA/M	
PSLLDQ, PSRLDQ	xmm,i	1	3	1/2	FA/M	
	<i>'</i>					
T. Control of the con	1 1		1	1	1	·

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 111100	
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 ins	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 instruc	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 = 256 any memory operand including indirect operands, m64 means m64 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

memory store instructions are included in the latency counts.

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

store instruction.

## Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipes	Notes
Move instructions					• •	
MOV	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	
XOI IO	1,1	2	'	'	LXOT	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
	102,[111]	2	2-5		LXOT	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	102/01,[]	4	3	2	27101	
SAHF		2	2	1		
SALC		1	1	1		
BSWAP	r	1	1	0.5	EX01	
PREFETCHNTA	m	1	'	0.5	LXOT	
PREFETCHT0/1/2	m	1		0.5		
PREFETCH/W	m	1		0.5		AMD 3DNow
SFENCE	"	6		89		AIVID JUNUW
LFENCE		1		0.25		
MFENCE		6		89		
Arithmetic instructio	ns					
ADD, SUB	r,r	1	1	0.5	EX01	
ADD, SUB	r,i	1	1	0.5	EX01	

ADD, SUB	r,m	1		0.5	EX01
ADD, SUB	m,r	1	7-8	1	EX01
ADD, SUB	m,i	1	7-8	1	EX01
ADC, SBB	r,r	1	1		EX01
ADC, SBB	r,i	1	1		EX01
ADC, SBB	r,m	1	1	1	EX01
ADC, SBB	m,r	1	9	1	EX01
ADC, SBB	m,i	1	9	1	EX01
CMP	r,r	1	1	0.5	EX01
CMP	r,i	1	1	0.5	EX01
CMP	r,m	1	'	0.5	EX01
INC, DEC, NEG	r	1	1	0.5	EX01
		1	7-8		
INC, DEC, NEG	m			1	EX01
AAA, AAS		10	6		
DAA		16	9		
DAS		20	10		
AAD		4	6		
AAM		9	20	20	
MUL, IMUL	r8/m8	1	4	2	EX1
MUL, IMUL	r16/m16	2	4	2	EX1
MUL, IMUL	r32/m32	1	4	2	EX1
MUL, IMUL	r64/m64	1	6	4	EX1
IMUL	r16,r16/m16	1	4	2	EX1
IMUL	r32,r32/m32	1	4	2	EX1
IMUL	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	20-27	EX0
IDIV	r64/m64	36	22-79	20-45	EX0
CBW, CWDE, CDQE	104/11104	1	1	20-73	EX01
		1	1	0.5	EX01
CDQ, CQO				0.5	
CWD		2	1	1	EX01
Logio inctructions					
Logic instructions		_	_	0.5	EV04
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

			Danaozei			
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		7	4	3.5	EX01	
SHLD, SHRD	r,r,cl		4			
· ·	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	
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		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 2 if jumping
LOOPE LOOPNE	short	1		1-2	EX1	
LOOPE LOOPINE	511011	I	1	1-2	ĽΛΙ	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		3 per 10B		Desi Case
REP MOVS						amall n
		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	- ,	2		4		
CPUID		37-63		112-280		
RDTSC		36		42		
RDPMC		22		300		
CRC32	r22 r0	3	3	2		
	r32,r8					
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		Execution	Domain, notes				
Move instructions				throughput	pipes					
FLD	r	1	2	0.5	P01	fp				
FLD	m32/64	1	8	1	101	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	101	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	111	1	0	0.5	P013	fp				
FCMOVcc	st0,r	8	3	3	P0 P1 F3	fp				
FFREE	r	1	3	0.25	none	ıρ				
FINCSTP, FDECSTP	•	1	0	0.25	none	inherit				
FNSTSW	AX	4	~13	22	P0 P2 P3	milent				
FNSTSW	m16	3	~13	19	P0 P2 P3					
FLDCW	m16	1	13	3	F0F2F3					
FNSTCW	m16	3		2						
INSTOV	11110	3								
Arithmetic instructions	 <b>=</b>									
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 MMX and XMM 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	. ,	1		_	D4	
Q	(x)mm,r/m	1	2	1	P1	

DUNDOKUODO		4	۱ ۵	4	D4	1
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	
PMOVSXBW/BD/BQ/						
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	5					
PADDB/W/D/Q/SB/SW					D00	
/USB/USW	(x)mm,r/m	1	2	0.5	P23	
PSUBB/W/D/Q/SB/SW/	(-)	_		0.5	Doo	
USB/USW	(x)mm,r/m	1	2	0.5	P23	00000
PHADD/SUB(S)W/D	X,X	3 4	5 5	2 2	P1 P23 P1 P23	SSSE3 SSSE3
PHADD/SUB(S)W/D PCMPEQ/GT B/W/D	x,m	1	2	0.5	P23	333E3
PCMPEQQ	(x)mm,r/m (x)mm,r/m		2	0.5	P23	SSE4.1
PCMPGTQ	(x)mm,r/m	1 1	2	0.5	P23	SSE4.1
PMULLW PMULHW	( <i>x)</i> 111111,1/111	I		0.5	F23	33E4.2
PMULHUW PMULUDQ						
I MOLITOW I MOLODQ	(x)mm,r/m	1	4	1	P0	
PMULLD	xmm,r/m	1	5	2	P0	SSE4.1
PMULDQ	xmm,r/m	1	4	1	P0	SSE4.1
PMULHRSW	(x)mm,r/m	1	4	1	P0	SSSE3
PMADDWD	(x)mm,r/m	1 1	4	1	P0	00020
PMADDUBSW	(x)mm,r/m	1 1	4	1	P0	
PAVGB/W	(x)mm,r/m	1	2	0.5	P23	
PMIN/MAX SB/SW/ SD	(*)!!!!!;!/!!!	'		0.5	1 23	
UB/UW/UD	(x)mm,r/m	1	2	0.5	P23	
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	333L3
	, ,					CCE4.1
MPSADBW	x,x,i	8	8	4	P1 P23	SSE4.1
VPCOMB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	AMD XOP latency 0 if i=6,7
V CONID/W/D/Q	^,^,*/!!!,!	'		0.5	FZJ	AMD XOP
VPCOMUB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
5511155/11/5/3	A,A,A 111,1	'	_	1 0.0	. 20	1.0.10, 0 11 1 0,7

VPHADDBW/BD/BQ/			1			
WD/WQ/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPHADDUBW/BD/BQ/	χ,χ		_	0.0	1 20	7 IIVIB 7(G)
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	(x)mm,i	1	2	1	P1	
PSLLDQ, PSRLDQ	xmm,i	1	2	1	P1	
PTEST	xmm,r/m	2		1	P1 P3	SSE4.1
VPROTB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
VPROTB/W/D/Q	x,x,i	1	2	1	P1	AMD XOP
VPSHAB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
VPSHLB/W/D/Q	x,x,x/m	1	3	1	P1	AMD XOP
String instructions						
PCMPESTRI	x,x,i	27	17	10	P1 P2 P3	SSE4.2
PCMPESTRM	x,x,i	27	10	10	P1 P2 P3	SSE4.2
PCMPISTRI	x,x,i	7	14	3	P1 P2 P3	SSE4.2
PCMPISTRM	x,x,i	7	7	4	P1 P2 P3	SSE4.2
Encryption						
PCLMULQDQ	x,x/m,i	5	12	7	P1	pclmul
AESDEC	X,X/111,1	2	5	2	P01	aes
AESDECLAST	X,X X,X	2	5	2	P01	aes
AESENC	x,x X,X	2	5	2	P01	aes
AESENCLAST		2	5	2	P01	
AESIMC	X,X	1	5		P01	aes
	X,X			1		aes
AESKEYGENASSIST	x,x,i	1	5	1	P0	aes
Other						
EMMS		1		0.25		

# Floating point XMM and YMM instructions

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

MOVA DC/D				I		
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	3.5					
MOVUPS/D	x,m128	1	6	0.5		
VMOVAPS/D						
VMOVUPS/D	y,m256	2	6	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	1	6	0.5		
MOVSS/D	m32/64,x	1	5	1		
MOVHPS/D						
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	
VMOVMSKPS/D	r32,y					
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,u,i 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		P1	ivec
VBLENDVPS/PD	y,y,y/m,y	2	2	2	P1	ivec
MOVDDUP	y,y,y/111,y X,X	1	2	1	P1	ivec
MOVDDUP	x,m64			0.5	Г	IVEC
VMOVDDUP		2	2	2	P1	ivoo
VMOVDDUP	y,y v m256	2		1	FI	ivec
	y,m256		6			
VBROADCASTSS	x,m32	1	6	0.5	DOO	
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	

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,x x,y	4	7	2	P01	fp
CVTPS2PD	X,Y X,X	2	7	1	P01	fp
VCVTPS2PD		4	7	2	P01	fp
CVTSD2SS	y,x x,x	1	4	1	P0	fp
CVTSB2SS CVTSS2SD		1	4	1	P0	fp
CVT3323D CVTDQ2PS	X,X	1	4	1	P0	
VCVTDQ2PS	X,X	2	4	2	P0	fp fn
	y,y	1		1		fp
CVT(T) PS2DQ	X,X		4		P0	fp
VCVT(T) PS2DQ	y,y	2	4	2	P0	fp
CVTDQ2PD	X,X	2	7	1	P01	fp
VCVTDQ2PD	y,x	4	8 7	2	P01	fp
CVT(T)PD2DQ	X,X	2		1	P01	fp
VCVT(T)PD2DQ	x,y	4	7	2	P01	fp
CVTPI2PS	x,mm	1	4	1	P0	fp
CVT(T)PS2PI	mm,x	1	4	1	P0	fp
CVTPI2PD	x,mm	2	7	1	P0 P1	fp
CVT(T) PD2PI	mm,x	2	7	1	P0 P1	fp
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
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,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	L C	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		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		bove	I			AMD FMA4
Math						
SQRTSS/PS	x,x/m	1	14-15	4.5-12	P01	fp
VSQRTPS	y,y/m	2	14-15	9-24	P01	fp
SQRTSD/PD	x,x/m	1	24-26	4.5-16.5	P01	fp
VSQRTPD	y,y/m	2	24-26	9-33	P01	fp
RSQRTSS/PS	x,x/m	1	5	1	P01	fp
VRSQRTPS	y,y/m	2	5	2	P01	fp
VFRCZSS/SD/PS/PD	y,y/111 X,X	2	10	2	P01	AMD XOP
VFRCZSS/SD/PS/PD	x,x x,m	3	10	2	P01	AMD XOP
VI NOZOGIODIF SIFD	۸,۱۱۱		10		F 0 1	AIVID AOF
Logic						
Logic AND/ANDN/OR/XORPS/						
PD	x,x/m	1	2	0.5	P23	ivec
VAND/ANDN/OR/XOR	7,77111		_	0.0	1 20	1000
PS/PD	y,y,y/m	2	2	1	P23	ivec
	J,J,J''''	_	_	'	. 20	
Other						
VZEROUPPER		9		4		32 bit mode
VZEROUPPER		16		5		64 bit mode
VZEROUFFER		10	1	၂ ၁		04 DIL IIIOGE

VZEROALL		17		6	P2 P3	32 bit mode	
VZEROALL		32		10	P2 P3	64 bit mode	
LDMXCSR	m32	1	10	4	P0 P3		
STMXCSR	m32	2	19	19	P0 P3		
FXSAVE	m4096	67	136	136	P0 P1 P2 P3		
FXRSTOR	m4096	116	176	176	P0 P1 P2 P3		
XSAVE	m	122	196	196	P0 P1 P2 P3		
XRSTOR	m	177	250	250	P0 P1 P2 P3		

## **AMD Piledriver**

#### List of instruction timings and macro-operation breakdown

Explanation of column headings:

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

etc.

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

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

memory 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						
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		

LFENCE		1		0.25	
MFENCE		7		81	
Arithmetic instructions	\$				
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		10	6		
DAA		16	9		
DAS		20	10		
AAD		4	6		
AAM		10	15	15	
MUL, IMUL	r8/m8	1	4	2	EX1
MUL, IMUL	r16/m16	2	4	2	EX1
MUL, IMUL	r32/m32	1	4	2	EX1
MUL, IMUL	r64/m64	1	6	4	EX1
IMUL	r16,r16/m16	1	4	2	EX1
IMUL	r32,r32/m32	1	4	2	EX1
IMUL	r64,r64/m64	1	6	4	EX1
IMUL	r16,(r16),i	2	5	2	EX1
IMUL	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	13-25	EX0
DIV	r32/m32	2	12-40	12-40	EX0
DIV	r64/m64	2	13-71	13-71	EX0
IDIV	r8/m8	9	17-21	13-18	EX0
IDIV	r16/m16	7	13-26	13-25	EX0
IDIV	r32/m32	2	13-40	13-40	EX0
IDIV	r64/m64	2	13-71	13-71	EX0
CBW, CWDE, CDQE		1	1		EX01
CDQ, CQO		1	1	0.5	EX01
CWD		2	1	1	EX01
Logic instructions			4	0.5	EVO
AND, OR, XOR	r,r	1	1	0.5	EX01
AND, OR, XOR	r,i	1	1	0.5	EX01

AND, OR, XOR	r,m	1		0.5	EX01	
AND, OR, XOR	m,r	1	7-8	1	EX01	
AND, OR, XOR	m,i	1	7-8	1	EX01	
TEST	r,r	1	1	0.5	EX01	
TEST	r,i	1	1	0.5	EX01	
TEST	m,r	1		0.5	EX01	
TEST	m,i	1		0.5	EX01	
NOT	r	1	1	0.5	EX01	
NOT	m	1	7-8	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
BLCFILL	r,r	2	2	1		AMD TBM
	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

BLSIC	r,r	2	2	1		AMD TBM
T1MSKC	r,r	2	2	1		AMD TBM
TZMSK	r,r	2	2	1		AMD TBM
	.,.		_			
Control transfer instru	ctions					
JMP	short/near	1 1		2	EX1	
JMP	r	1 1		2	EX1	
JMP	m	1 1		2	EX1	
Jcc	short/near	1 1		1-2	EX1	2 if jumping
fused CMP+Jcc	short/near	1 1		1-2	EX1	2 if jumping
J(E/R)CXZ	short	1 1		1-2	EX1	2 if jumping
LOOP	short	1 1		1-2	EX1	2 if jumping
LOOPE LOOPNE	short	1 1		1-2	EX1	
CALL		2		2	EX1	2 if jumping
	near					
CALL	r	2		2	EX1	
CALL	m	3		2	EX1	
RET	_	1 1		2	EX1	
RET	i	4		2	EX1	_
BOUND	m	11		5		for no jump
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		3		3		
REP STOS		1n		1n		small n
REP STOS		3 per 16B		3 per 16B		best case
MOVS		5		3		
REP MOVS		1-3n		1n		small n
REP MOVS		4.5 pr 16B		3 per 16B		best case
SCAS		3		3		2001 0000
REP SCAS		7n		3-4n		
CMPS		6		3		
REP CMPS		9n		4n		
REF CIVIFS		911		411		
Synchronization						
LOCK ADD	m r	1 1	- 40			
XADD	m,r	1 1	~40			
	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	
PAUSE		40		40		

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	instructions	5				
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						
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 MMX and XMM 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	0021.1
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	33L4.1
EXTRQ	X,i,i	1	3	1	P1	AMD SSE4A
EXTRQ		1	1	1	P1	AMD SSE4A
INSERTQ	X,X	1	1	1	P1	AMD SSE4A
	x,x,i,i	-	1		P1	
INSERTQ	x,x	1	I	1	PI	AMD SSE4A
PMOVSXBW/BD/BQ/		_		4	D4	00544
WD/WQ/DQ	x,x	1	2	1	P1	SSE4.1
PMOVZXBW/BD/BQ/W		_		4	D4	00544
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
A -141 41 - 1 4 41						
Arithmetic instructions	<b>S</b>					
PADDB/W/D/Q/SB/SW /USB/USW	(14) ma ma m/ma	4	_	0.5	DOO	
	(x)mm,r/m	1	2	0.5	P23	
PSUBB/W/D/Q/SB/SW/ USB/USW		_		0.5	DOO	
	(x)mm,r/m	1	2	0.5	P23	00050
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	00544
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	
DMIIIID	(x)mm,r/m	1	4	1	P0	00544
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
						AMD XOP
VPCOMB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
	. , ,					AMD XOP
VPCOMUB/W/D/Q	x,x,x/m,i	1	2	0.5	P23	latency 0 if i=6,7
VPHADDBW/BD/BQ/	. , ,					,
WD/WQ/DQ	x,x/m	1	2	0.5	P23	AMD XOP
VPHADDUBW/BD/BQ/						
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
VI WIN ID COOVED	λ,λ,λ,τι,λ					7 WIB 7(O)
Logic						
PAND PANDN POR						
PXOR	(x)mm,r/m	1	2	0.5	P23	
PSLL/RL W/D/Q	(24)		_	0.0	. 20	
PSRAW/D	(x)mm,r/m	1	3	1	P1	
PSLL/RL W/D/Q	(24)					
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
	71,71,70111	•		·		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
	71,71,1					0022
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,111,1	2	5	2	P01	aes
AESDECLAST	X,X X,X	2	5	2	P01	aes
AESENC	X,X X,X	2	5	2	P01	aes
AESENCLAST	X,X X,X	2	5	2	P01	aes
AESIMC	X,X X,X	1	5	1	P0	aes
AESKEYGENASSIST	x,x,i	1	5	1	P0	aes
	7,7,1					
T	I	L	T.	I	I	1

Other				
EMMS	1	0.25		

# Floating point XMM and YMM instructions

Instruction	Operands	Ops	Latency	Reciprocal	Execution	Domain, notes
•• • • •				throughput	pipes	
Move instructions						
MOVAPS/D		_		0.05		to be with all a consider
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 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		1	2	1	P1	ivec
MOVDDUP	x,x x,m64	1		0.5	ГІ	IVEC
VMOVDDUP	*	2	2	0.5	P1	ivoo
	y,y	2		1	۲I	ivec
VMOVDDUP	y,m256					
VBROADCASTSS	x,m32	1	6	0.5	Doo	
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		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	
	73.73					
Conversion						
CVTPD2PS	x,x	2	8	1	P01	ivec/fp
VCVTPD2PS	x,y	4	7	2	P01	ivec/fp
CVTPS2PD	x,x	2	8	1	P01	ivec/fp
VCVTPS2PD	y,x	4	8	2	P01	ivec/fp
CVTSD2SS	x,x	1	4	1	P0	fp
CVTSS2SD	x,x	1	4	1	P0	fp
CVTDQ2PS	x,x	1	4	1	P0	fp
VCVTDQ2PS	y,y	2	4	2	P0	fp
CVT(T) PS2DQ	x,x	1	4	1	P0	fp
VCVT(T) PS2DQ	y,y	2	4	2	P0	fp
CVTDQ2PD	x,x	2	8	1	P01	ivec/fp
VCVTDQ2PD	y,x	4	8	2	P01	ivec/fp
CVT(T)PD2DQ	x,x	2	8	1	P01	fp/ivec
VCVT(T)PD2DQ	x,y	4	7	2	P01	fp/ivec
CVTPl2PS	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
, 2002. 0/B	23/24/111	'	, 55	0.0		

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		7	P01 P23	SSE4.1
VDPPS	y,y,y,i	25	27	13	P01 P3	SSE4.1
VDPPS	y,m,i	29		13	P01 P3	SSE4.1
DPPD	x,x,i	15	15	5	P01 P23	SSE4.1
DPPD	x,m,i	17		6	P01 P23	SSE4.1
VFMADD132SS/SD	x,x,x/m	1	5-6	1	P01	FMA3
VFMADD132PS/PD	x,x,x/m	1	5-6	1	P01	FMA3
VFMADD132PS/PD	y,y,y/m	2	5-6	1	P01	FMA3
All other FMA3 instruction		bove	L C	0.5	D04	FMA3
VFMADDSS/SD	x,x,x,x/m	1	5-6	0.5	P01	AMD FMA4
VFMADDPS/PD	x,x,x,x/m	1 2	5-6	0.5	P01	AMD FMA4 AMD FMA4
VFMADDPS/PD	y,y,y,y/m	l	5-6	1	P01	AMD FMA4 AMD FMA4
All other FMA4 instruction	ons: same as a	bove				AMD FMA4
Math						
SQRTSS/PS	x,x/m	1	13-15	5-12	P01	fp
VSQRTPS	y,y/m	2	14-15	9-24	P01	fp
SQRTSD/PD	x,x/m	1	24-26	5-15	P01	fp
VSQRTPD	y,y/m	2	24-26	9-29	P01	fp
RSQRTSS/PS	x,x/m	1	5	1	P01	fp
VRSQRTPS	y,y/m	2	5	2	P01	fp
VFRCZSS/SD/PS/PD	x,x	2	10	2	P01	AMD XOP
VFRCZSS/SD/PS/PD	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 m64 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

memory 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	EX01	
XCHG	r32,r32	2		0.5	EX01	
XCHG	r64,r64	2		0.5	EX01	
ACIIG	104,104	2	'	0.5		Timing depends on
XCHG	r,m	2	~38	~38	EX01	Timing depends on hw
XLAT	1,111	2	6	2	LXUI	
PUSH	r	1		1		
PUSH	i			1		
PUSH		1 2		1		
	m	8		4		
PUSHF(D/Q)				9		
PUSHA(D) POP	_	9				
	r	1		1		
POPE(D(O)	m	2		1		
POPF(D/Q)		34		19		
POPA(D)		14		8		
POP	sp	1	2		EV04	and a data at a s
LEA	r16,[m]	2	2-3		EX01	any addr. size
LEA	r32,[m]	1	2		EX01	16 bit addr. size
	200/04 Facil	4		0.5	EV04	scale factor > 1
LEA	r32/64,[m]	1	2	0.5	EX01	or 3 operands
LEA	r32/64,[m]	1	1	0.5	EX01	all other cases
LAHF		4	3	2		
SAHF		2	2	1		
SALC		1	1	1		
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	 					

			oleannone	1	
ADD, SUB	r,r	1	1	0.5	EX01
ADD, SUB	r,i	1	1	0.5	EX01
ADD, SUB	r,m	1	-	0.5	EX01
ADD, SUB	m,r	1	7	1	EX01
ADD, SUB	m,i	1	7	1	EX01
ADC, SBB	r,r	1	1	'	EX01
ADC, SBB	r,i	1	1		EX01
ADC, SBB			1	1	EX01
	r,m		9	1	EX01
ADC, SBB ADC, SBB	m,r	-	9	1	
1	m,i	1	1		EX01
CMP	r,r	-		0.5	EX01
CMP	r,i	1	1	0.5	EX01
CMP	r,m	1		0.5	EX01
CMP	m,i	1		0.5	EX01
INC, DEC, NEG	r	1	1	0.5	EX01
INC, DEC, NEG	m	1	7	1	EX01
AAA, AAS		10	6		
DAA		16	8		
DAS		20	10		
AAD		4	6		
AAM		10	15	15	
MUL, IMUL	r8/m8	1	4	2	EX1
MUL, IMUL	r16/m16	2	4	2	EX1
MUL, IMUL	r32/m32	1	4	2	EX1
MUL, IMUL	r64/m64	1	6	4	EX1
IMUL	r16,r16/m16	1	4	2	EX1
IMUL	r32,r32/m32	1	4	2	EX1
IMUL	r64,r64/m64	1	6	4	EX1
IMUL	r16,(r16),i	2	5	2	EX1
IMUL	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	15-17	EX0
DIV	r32/m32	2	13-23	13-23	EX0
DIV	r64/m64	2	13-39	13-39	EX0
IDIV	r8/m8	9	17-22	13-70	EX0
IDIV	r16/m16	7 2	14-25	14-24	EX0
IDIV	r32/m32		13-39	13-39	EX0
IDIV	r64/m64	2	13-70	13-70	EX0
CBW, CWDE, CDQE		1	1	0.5	EX01
CDQ, CQO		1	1	0.5	EX01
CWD		2	1	1	EX01
Logic instructions		_	_	0.5	EV04
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
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	5
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	
I						
RCR	r,i	15	7		EX01	
RCR	r,cl	16	7		EX01	
SHLD, SHRD	r,r,i	6	3	3	EX01	
SHLD, SHRD	r,r,cl	7-8	4	4	EX01	
I	m,r,i/CL			4	EX01	
SHLD, SHRD		8	_			
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		4	_		EX01	
	m,i			2		
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		9	_	5	EX01	
	r,m					
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		
I						
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	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		2	2	1		AMD TBM
	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
BLSIC	r,r	2	2	1		AMD TBM
T1MSKC	r,r	2	2	1		AMD TBM
TZMSK	r,r	2	2	1		AMD TBM
		_	_			
	1		l			

		I I		1 1		 
Control transfer instru					F.V.4	
JMP	short/near	1 1		2	EX1	
JMP	r	1 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		1-2	EX1	2 if jumping
LOOP	short	1 1		1-2	EX1	2 if jumping
LOOPE LOOPNE	short	1 1		1-2	EX1	2 if jumping
CALL	near	2		2	EX1	, , ,
CALL	r	2		2	EX1	
CALL	m	3		2	EX1	
RET	•••	1 1		2	EX1	
RET	i	4		2	EX1	
BOUND		11		5		for no iumn
	m					for no jump
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	11132/11104	3		3		
						a ma a II m
REP STOS		1n		~1n		small n
REP STOS		3 per 16B		2 per 16B		best case
MOVS		5		3		
REP MOVS		~1n		~1n		small n
REP MOVS		4-5 pr 16B		~2 per 16B		best case
SCAS		3		3		
REP SCAS		7n		3-4n		
CMPS		6		3		
REP CMPS		9n		4n		
Synchronization						
LOCK ADD	m,r	1	~39			
XADD		4	9-12			
	m,r m r		9-12 ~39			
LOCK XADD	m,r	4				
CMPXCHG	m,r8	5	15 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			
Other						
NOP (90)		1		0.25	none	
Long NOP (0F 1F)				0.25		
, ,		1			none	
PAUSE	- 0	8		4		
ENTER	a,0	13		21		
ENTER	a,b	11+5b		20-30		

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

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	 S					
FADD(P),FSUB(R)(P)	r/m	1	5	1	P01	fma
FIADD,FISUB(R)	m	2		2	P01	fma
FMUL(P)	r/m	1	5	1	P01	fma
FIMUL	m	2		2	P01	fma
FDIV(R)(P)	r	1	9-37	4-16	P01	fp
FDIV(R)	m	1			P01	fp
FIDIV(R)	m	2		4	P01	fp
FABS, FCHS		1	2	0.5	P01	fp
FCOM(P), FUCOM(P)	r/m	1		0.5	P01	fp
FCOMPP, FUCOMPP		1		0.5	P01	fp
FCOMI(P)	r	2	2	1	P01 P2	fp
FICOM(P)	m	2		1	P01	fp
FTST		1		0.5	P01	fp
FXAM		1	26	0.5	P01	fp
FRNDINT		1	4	1	P0	fp
FPREM FPREM1		1	17-60	12-53	P0	fp

Math					
FSQRT		1	10-50	5-20	P01
FLDPI, etc.		1		0.5	P01
FSIN		10-164	60-210	60-165	P0 P1 P2
FCOS		18-166	76-158		P0 P1 P2
FSINCOS		12-168		90-165	P0 P1 P2
FPTAN		11-192	90-245	90-210	P0 P1 P2
FPATAN		10-365	60-440	60-365	P0 P1 P2
FSCALE		10	49	5	P0 P1 P2
FXTRACT		12	8	5	P0 P1 P2
F2XM1		10-18	60-74		P0 P1 P2
FYL2X		9-183	60-280		P0 P1 P2
FYL2XP1		206	~390		P0 P1 P2
Other					
FNOP		1		0.25	none
(F)WAIT		1		0.25	none
FNCLEX		18		63	P0
FNINIT		31		131	P0
FNSAVE	m864	98	256	256	P0 P1 P2
FRSTOR	m864	73	166	166	P0 P2

**Integer MMX and XMM 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		

			J. 10 a. 1 . 1 . 1 . 1 . 1	•		
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	P02	SSE4.1
MASKMOVQ	mm,mm	31	32	16	P2	002111
MASKMOVDQU	xmm,xmm	65	45	31	P0 P1 P2	
PMOVMSKB	r32,mm/x	2	5	1	P1 P2	
PEXTRB/W/D/Q	r,x/mm,i	2	5	1	P1 P2	SSE4.1
PINSRB/W/D/Q	x/mm,r,i	2	6	1	P1	33L4.1
EXTRQ	x,i,i	1	3	1	P1	AMD SSE4A
EXTRQ		1	1	1	P1	AMD SSE4A
INSERTQ	X,X		-	1	P1	AMD SSE4A
INSERTQ	x,x,i,i	1 1	1	1	P1	AMD SSE4A
· ·	x,x	I	I	I	PI	AIVID SSE4A
PMOVSXBW/BD/BQ/ WD/WQ/DQ	V V	4	2	4	P1	SSE4.1
	x,x	1		1	PI	SSE4.1
PMOVZXBW/BD/BQ/W		4	_	4	D4	CCE4.4
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
Arithmetic instructions	3					
PADDB/W/D/Q/SB/SW	(-)			0.5	Doo	
/USB/USW	(x)mm,r/m	1	2	0.5	P02	
PSUBB/W/D/Q/SB/SW/	( )	_		0.5	Doo	
USB/USW	(x)mm,r/m	1	2	0.5	P02	00050
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	00=4.4
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	
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	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
PSADBW	(x)mm,r/m	2	4	1	P02	
	,,,,,,	_				ı

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,x,i	7	11	7	P1	pclmul
PCLMULQDQ	x,x,m,i	8	_	7	P1	pclmul
AESDEC	X,X	2	5	1	P01	aes
AESDECLAST	X,X	2	5	1	P01	aes
AESENC	x,x	2	5	1	P01	aes
AESENCLAST	X,X	2	5	1	P01	aes
AESIMC	X,X	1	5	1	P0	aes
AESKEYGENASSIST	x,x,i	1	5	1	P0	aes
Other						
EMMS		1		0.25		

# Floating point 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	•
VBLENDVPS/PD	y,y,y/m,y	2	2	1	P01	
MOVDDUP	x,x	1	2	1	P1	ivec
MOVDDUP	x,m64	1		0.5		
VMOVDDUP	y,y	2	2	2	P1	ivec
VMOVDDUP	y,m256	2		1		
VBROADCASTSS	x,m32	1	8	0.5		
VBROADCASTSS	y,m32	2	8	0.5	P02	
VBROADCASTSD	y,m64	2	8	0.5	P02	
VBROADCASTF128	y,m128	2	8	0.5	P02	
MOVSH/LDUP	x,x	1	2	1	P1	ivec
MOVSH/LDUP	x,m128	1	_	0.5	- ·	

VMOVSH/LDUP	y,y	2	2	2	P1	ivec
VMOVSH/LDUP	y,m256	2		1		
UNPCKH/LPS/D	x,x/m	1	2	1	P1	ivec
VUNPCKH/LPS/D	y,y,y/m	2	2	2	P1	ivec
EXTRACTPS	r32,x,i	2		1	P1 P2	
EXTRACTPS	m32,x,i	2	10	1	P1 P2	
VEXTRACTF128	x,y,i	1	2	0.5	P02	ivec
VEXTRACTF128	m128,y,i	2	10	1	P0 P2	
INSERTPS	x,x,i	1	2	1	P1	
INSERTPS	x,m32,i	1	9	2	P1	
VINSERTF128	y,y,x,i	2	2	1	P02	ivec
VINSERTF128	y,y,m128,i	2	10	1	P02	
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	
VIVIAGINIO VI GID	111230,y,y	71	33	10	101112	
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		P0 P2	fp
CVTSI2SD	x,r32/64	2	12		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		7	2	_	
VCVTPH2PS	y,x/m	4	/	2	P0 P1	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
VADDPS/D VSUBPS/D	yyylm	2	5-6	2	P01	fma
ADDSUBPS/D	y,y,y/m x,x/m	1	5-6	1	P01	fma
VADDSUBPS/D	y,y,y/m	2	5-6	1	P01	fma
	y,y,y/111	_	J-0	l <b>'</b>	101	iiia

		1	1			 
HADDPS/D HSUBPS/D	X,X	4	10	2	P0 P1	ivec/fma
VHADDPS/D	71,71	•		_		1700/11110
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	<b>y,y</b>	_		_	. • .	. 19
CMPPS/D	x,x/m	1	2	0.5	P01	fp
VCMPPS/D	y,y,y/m	2	2	1	P01	fp
COMISS/D	y,y,y/111	_	_	•		۱۴
UCOMISS/D	x,x/m	2		1	P01 P2	fp
MAXSS/SD/PS/PD	25,24111	_		•	2	۲,
MINSS/SD/PS/PD	x,x/m	1	2	0.5	P01	fp
Militoorobri on b	7,77111		_	0.0		۱۴
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		I	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		I	J -0	'	101	AMD FMA4
All other riving instruction			1			AIVID I IVIA
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	y,y/111 X,X	2	10	2	P01	AMD XOP
VFRCZSS/SD/PS/PD	x,x x,m	4	10	2	P01	AMD XOP
VINOZOOIODIFOIFD	۸,۱۱۱	4		2	гиі	AIVID AUF
Logic						
AND/ANDN/OR/XORPS/						
PD	x,x/m	1	2	0.5	P02	ivec
I	,	1	_		·	

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 through-

put:

This is also called issue latency. This value indicates the average number of clock cycles from the execution of an instruction begins to a subsequent independent instruction of the same kind can begin to execute. A value of 1/2 indicates that the execution with a subsequent independent instruction of the same kind can begin to execute. A value of 1/2 indicates that the execution with a subsequent indicates the average number of clock cycles in the same kind can begin to execute.

cates 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	10/1	
MOVZX, MOVSX	r,m	1	5	1		
MOVSXD	r64,r32	1	1	0.5		
MOVSXD	r64,m32	1	5	1		
CMOVcc	r,r	1	1	0.5	10/1	
CMOVcc	r,m	1		1		
XCHG	r,r	2	1	1	IO/1	
XCHG	r,m	3	20			Timing dep. on hw

			Вов	Jul		
XLAT		2	5			
PUSH	r	1		1		
PUSH	i	1		1		
PUSH		3		2		
I	m					
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	r32/64,[m]	1	1	0.5	10/1	no scale, no offset
LEA	r32/64,[m]	1	2-4	1	10	w. scale or offset
LEA	r64,[m]	1	'	0.5	10/1	RIP relative
LAHF	104,[111]	4	4	2	10/1	TAIT TOIGHTVC
SAHF		1	1	0.5	10/1	
I				0.5	10/ 1	
SALC		1	1	0.5	1074	
BSWAP	r	1	1	0.5	10/1	
PREFETCHNTA	m	1		1	AGU	
PREFETCHT0/1/2	m	1		1	AGU	
PREFETCH	m	1		1	AGU	AMD only
SFENCE		4		~45	AGU	
LFENCE		1		1	AGU	
MFENCE		4		~45	AGU	
Arithmetic instructions		4	_	0.5	10/4	
ADD, SUB	r,r/i	1	1	0.5	10/1	
ADD, SUB	r,m	1		1		
ADD, SUB	m,r	1		1		
ADC, SBB	r,r/i	1	1	1	10/1	
ADC, SBB	r,m	1		1		
ADC, SBB	m,r/i	1	6-7			
CMP	r,r/i	1	1	0.5	10/1	
CMP	r,m	1		1		
INC, DEC, NEG	r	1	1	0.5	10/1	
INC, DEC, NEG	m	1	6			
AAA		9	5			
AAS		9	10			
DAA		12	7			
DAS		16	8			
AAD		4	5			
AAM				22		
	#0 /ma 0	33	23	23	10	
MUL, IMUL	r8/m8	1	3	1	10	
MUL, IMUL	r16/m16	3	3-5	_	10	latency ax=3, dx=5
MUL, IMUL	r32/m32	2	3-4	2	10	latency eax=3, edx=4
MUL, IMUL	r64/m64	2	6-7		10	latency rax=6, rdx=7
IMUL	r16,r16/m16	1	3	1	10	
IMUL	r32,r32/m32	1	3	1	10	
IMUL	r64,r64/m64	1	6	4	10	
IMUL	r16,(r16),i	2	4	3	10	
IMUL	r32,(r32),i	1	3	1	10	
IMUL	r64,(r64),i	1	7	4	10	
DIV	r8/m8	1	27	27	10	
1 1 5	1	-			. •	ı I

DIV							
DIV	DIV	r16/m16	1	33	33	In	
DIV	1						
IDIV							
IDIV	1						
IDIV		r8/m8	1				
IDIV   CBW, CWDE, CDQE   CWD, CDQ, CQO	IDIV	r16/m16	1	37	37	10	
IDIV   CBW, CWDE, CDQE   CWD, CDQ, CQO	IDIV	r32/m32	1	55	55	10	
CBW, CWDE, CDQE   1	1						
CWD, CDQ, CQQ		104/11104	_		01		
Logic instructions	1						
AND, OR, XOR	CWD, CDQ, CQO		1	1		10/1	
AND, OR, XOR							
AND, OR, XOR							
AND, OR, XOR	AND, OR, XOR	r,r	1	1	0.5	I0/1	
AND, OR, XOR		r.m	1		1		
TEST	1						
TEST	1			1		IO/1	
NOT NOT	1			!		10/1	
NOT SHL, SHR, SAR r, i/CL 1 1 1 0.5 10/1 RCL, SHR, SAR r, i/CL 1 1 1 0.5 10/1 RCL, RCR r, i 1 1 1 1 1 1 10/1 RCL, RCR r, i 9 5 5 5 RCR RCR r, i 7 4 4 4 RCL RCR r, i 7 4 4 4 RCL RCR RCR r, i 7 4 4 4 RCL RCR RCR RCR RCR RCR RCR RCR RCR RCR						10/4	
SHL, SHR, SAR         r,i/CL         1         1         0.5         10/1           ROL, ROR         r,i/CL         1         1         0.5         10/1           ROL, ROR         r,i/CL         1         1         0.5         10/1           RCL, ROR         r,i         1         1         1         10/1           RCR         r,i         7         4         4           RCL         r,CL         9         5         4           SHL,SHR,SAR,ROL,         RCR         r,CL         9         5         4           SHL,SHR,SAR,ROL,         RCR         r,CL         9         5         4         4           RCR         m,i         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1<	1			1		10/1	
ROL, ROR         r,i/CL         1         1         0.5         10/1           RCL, RCR         r,1         1         1         1         1         10/1         10/1           RCL         r,i         9         5         5         5         7         4         4         4         4         7         4         4         4         8	NOT	m	1				
RCL, RCR         r,1         1         1         1         1         10/1           RCL         r,i         9         5         5         10/1           RCR         r,i         7         4         4         4           RCL         r,CL         9         6         5         8           RCR         r,CL         9         5         4         4           SHL,SHR,SAR,ROL,         RCR         r,CL         9         5         4         4           RCL, RCR         m,1         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         8         7         14         4         8         14         8         15         8         8         15         8         15         8         15         8         15         8         <	SHL, SHR, SAR	r,i/CL	1	1	0.5	I0/1	
RCL, RCR         r,1         1         1         1         1         10/1           RCL         r,i         9         5         5         10/1           RCR         r,i         7         4         4         4           RCL         r,CL         9         6         5         8           RCR         r,CL         9         5         4         4           SHL,SHR,SAR,ROL,         RCR         r,CL         9         5         4         4           RCL, RCR         m,1         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         7         1         8         7         14         4         8         14         8         15         8         8         15         8         15         8         15         8         15         8         <	ROL, ROR	r,i/CL	1	1	0.5	I0/1	
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,         m,i /CL         1         7         1           RCL, RCR         m,1         1         7         1           RCL         m,i         10         ~15         1           RCL         m,i         10         ~15         1           RCR         m,i         10         ~15         1           RCL         m,CL         9         15         15           RCR         m,CL         8         18         15           RCL         RCR         m,CL         8         18         15           RCL         RCR         m,r,r,cl	1		1	1		10/1	
RCR         r,i         7         4         4         4         RCL         RCL         r,CL         9         6         5         RCR         RCR         r,CL         9         5         4         SHLSHR,SAR,ROL, RCL         RCR         m,i         1         7         1         TCL         RCL         RCL         M,1         1         7         1         TCL         RCL         RCL         M,1         1         7         1         TCL         RCL         RCL         M,1         1         7         1         TCL         RCL         M,1         1         7         1         TCL         RCL         M,1         1         7         1         TCL         RCL         M,1         1         1         1         1         1         RCL         M,1         1 <td< td=""><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td></td<>	1						
RCL RCR RCR SHL,SHR,SAR,ROL, ROR         r,CL r,CL         9         6         5         4           SHL,SHR,SAR,ROL, ROR         m,i /CL m,i         1         7         1         1         7         1         8         14         8         14         8         14         8         14         8         14         8         15         8         15         8         15         8         15         8         15         8         15         8         15         8         14         8         15         8         15         8         14         8         15         8         15         8         15         8         15         8         15         8	1			1			
RCR SHL,SHR,SAR,ROL, ROR         r,CL         9         5         4           RCL, SCR RCL         m,i         1         7         1           RCL, RCR RCR         m,i         10         ~15           RCR RCR         m,i         9         18         ~14           RCL RCR         m,CL         9         15         8           SHLD, SHRD         r,r,i         6         3         3           SHLD, SHRD         r,r,cl         7         4         4           SHLD, SHRD         r,r,cl         7         4         4           SHLD, SHRD         m,r,i/CL         8         18         15           BT         r,r,cl         7         4         4           SHLD, SHRD         m,r,i/CL         8         18         15           BT         m,r,r/Cl         8         18         15           BT         m,r         5         3         3           BTC, BTR, BTS         r,r/fi         2         2         1           BTR, BTS         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BS							
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         8           RCR         m,CL         8         15         8           SHLD, SHRD         r,r,i         6         3         3           SHLD, SHRD         r,r,i         1         1         5           BT         r,r/i         1         0.5         8           BT         r,r/i         1         1         1           BT         m,i         1         1         1           BT         m,r         5         3         3           BTC, BTR, BTS         r,r/i         2         2         1           BTR, BTS         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11 <td< td=""><td>1</td><td></td><td>1</td><td></td><td></td><td></td><td></td></td<>	1		1				
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           RCR         m,CL         9         15         RCR           RCR         m,CL         8         15         SHLD, SHRD         r,r,i         6         3         3           SHLD, SHRD         r,r,cl         7         4         4         4         SHLD, SHRD         m,r,i//CL         8         18         15         BT         BT         r,r/i         1         0.5         BT         BT         m,r/i/CL         8         18         15         BT         BT         BT         BT         m,r/i/CL         8         18         15         BT		r,CL	9	5	4		
RCL, RCR       m,1       1       7       1         RCL       m,i       10       ~15         RCR       m,i       9       18       ~14         RCL       m,CL       9       15         RCR       m,CL       8       15         SHLD, SHRD       r,r,i       6       3       3         SHLD, SHRD       r,r,cl       7       4       4         SHLD, SHRD       m,r,i/CL       8       18       15         BT       m,r,i/CL       8       18       15         BT       m,r,i/i       1       0.5         BTC, BTR, BTS       m,r,r/i       2       2       1         BTC, BTR, BTS       m,i       4-5       15         BTC       m,r       8       16       13         BTR, BTS       m,r       8       15       15         BSF, BSR       r,r       11       6       6         BSF, BSR       r,m       11       6       8         BSF, BSR       r,r/m       11       0.5       SSE4.A/SSE4.2         SETcc       m       1       0.5       IO/1         CLC, STC <t< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td></t<>							
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,ci       6       3       3         SHLD, SHRD       r,r,ci       7       4       4         SHLD, SHRD       m,r,i/CL       8       18       15         BT       r,r/i       1       0.5         BT       m,i       1       1         BT       m,r       5       3         BTC, BTR, BTS       r,r/i       2       2       1         BTC, BTR, BTS       m,i       4-5       15         BTC       m,r       8       16       13         BTR, BTS       m,r       8       15       15         BSF, BSR       r,r       11       6       6         BSF, BSR       r,m       11       6       8         BSF, BSR       r,m       11       0.5       SSE4.A/SSE4.2         SETcc       r       1       0.5       IO/1         CLC, STC       1       0.5       IO/1	ROR	m,i /CL	1	7	1		
RCL         m,i         10         ~15           RCR         m,i         9         18         ~14           RCL         m,CL         9         15           RCR         m,CL         8         15           SHLD, SHRD         r,r,i         6         3         3           SHLD, SHRD         r,r,cl         7         4         4           SHLD, SHRD         m,r,i/CL         8         18         15           BT         m,r,i/CL         8         18         15           BT         m,r,i/I         1         0.5           BT         m,r         5         3           BTC, BTR, BTS         r,r/i         2         2         1           BTC         m,i         4-5         15           BTR, BTS         m,i         4-5         15           BTC         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11         6         7           SETcc         r         1	RCL, RCR	m,1	1	7	1		
RCR         m,i         9         18         ~14           RCL         m,CL         9         15           RCR         m,CL         8         15           SHLD, SHRD         r,r,i         6         3         3           SHLD, SHRD         r,r,i         7         4         4           SHLD, SHRD         m,r,i/CL         8         18         15           BT         r,r/i         1         0.5         15           BT         m,i         1         1         1           BT         m,i         1         1         1           BTC, BTR, BTS         r,r/i         2         2         1           BTC, BTR, BTS         m,i         4-5         15           BTR, BTS         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         9           POPCNT         r,r/m         8         5         SSE4.A/SSE4.2           SETcc         m         1         0.5         S	1		10		~15		
RCL         m,CL         9         15           RCR         m,CL         8         15           SHLD, SHRD         r,r,i         6         3         3           SHLD, SHRD         r,r,cl         7         4         4           SHLD, SHRD         m,r,i/CL         8         18         15           BT         r,r/i         1         0.5         15           BT         m,i         1         1         1           BT         m,i         1         1         1           BTC, BTR, BTS         r,r/i         2         2         1           BTC         m,i         4-5         15           BTR, BTS         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         9           POPCNT         r,r/m         8         5         SSE4.A/SSE4.2           SETcc         m         1         0.5         SSE4.A, AMD only           SETcc         m         1         0.5	1			18			
RCR         m,CL         8         15           SHLD, SHRD         r,r,i         6         3         3           SHLD, SHRD         r,r,cl         7         4         4           SHLD, SHRD         m,r,i/CL         8         18         15           BT         m,r,i/CL         8         18         15           BT         m,r,i/CL         8         18         15           BT         m,i         1         1         1           BT         m,i         1         1         1           BT         m,r         5         3         3           BTC, BTR, BTS         m,i         4-5         15         5           BTC         m,r         8         16         13         8         15         15         8           BTR, BTS         m,r         8         15         15         8         15         15         8         18         15         15         8         18         15         15         8         18         15         15         8         18         15         15         8         18         15         15         15         10         10	1			10			
SHLD, SHRD         r,r,i         6         3         3           SHLD, SHRD         r,r,cl         7         4         4           SHLD, SHRD         m,r,i/CL         8         18         15           BT         r,r/i         1         0.5           BT         m,i         1         1           BT         m,r         5         3           BTC, BTR, BTS         r,r/i         2         2         1           BTC         m,i         5         15           BTR, BTS         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11         6         8           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           SETcc         r         1         1         0.5         SSE4.A, AMD only           SETcc         m         1         0.5         10/1           CMC         1         1         0.5         10/1           CDD         1         1 <t< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td></t<>							
SHLD, SHRD         r,r,cl         7         4         4           SHLD, SHRD         m,r,i/CL         8         18         15           BT         r,r/i         1         0.5           BT         m,i         1         1           BT         m,i         1         1           BTC, BTR, BTS         r,r/i         2         2         1           BTC, BTS, BTS         m,i         4-5         15           BTC, BTR, BTS         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         8           BSF, BSR         r,m         11         6         9           BSF, BSR         r,r/m         9         12         5         SSE4.A/SSE4.2           LZCNT         r,r/m         8         5         SSE4.A, AMD only           SETcc         m         1         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1 <td< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td></td<>							
SHLD, SHRD         m,r,i/CL         8         18         15           BT         r,r/i         1         0.5           BT         m,i         1         1           BT         m,r         5         3           BTC, BTR, BTS         r,r/i         2         2         1           BTC         m,i         4-5         15           BTC         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11         6         8           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           SETcc         r         1         1         0.5         SSE4.A, AMD only           SETcc         m         1         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         0.5         10/1	1		1				
BT         r,r/i         1         0.5           BT         m,i         1         1           BT         m,i         1         1           BTC, BTR, BTS         r,r/i         2         2         1           BTC         m,i         5         15           BTR, BTS         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         9           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           SETcc         r         1         1         0.5         SSE4.A, AMD only           SETcc         m         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10							
BT         m,i         1	SHLD, SHRD	m,r,i/CL	8	18	15		
BT         m,r         5         3           BTC, BTR, BTS         r,r/i         2         2         1           BTC         m,i         5         15           BTR, BTS         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         7           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           SETcc         r,r/m         8         5         SSE4.A, AMD only           SETcc         m         1         1         0.5           SETcc         m         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	BT	r,r/i	1		0.5		
BT         m,r         5         3           BTC, BTR, BTS         r,r/i         2         2         1           BTC         m,i         5         15           BTR, BTS         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         7           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           LZCNT         r,r/m         8         5         SSE4.A, AMD only           SETcc         m         1         1         0.5           SETcc         m         1         0.5         10/1           CLC, STC         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	ВТ	m,i	1		1		
BTC, BTR, BTS         r,r/i         2         2         1           BTC         m,i         5         15           BTR, BTS         m,i         4-5         15           BTC         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11         6         7           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           SETcc         r,r/m         8         5         SSE4.A, AMD only           SETcc         m         1         0.5         10/1           CLC, STC         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	ВТ		5		3		
BTC         m,i         5         15           BTR, BTS         m,i         4-5         15           BTC         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           LZCNT         r,r/m         8         5         SSE4.A, AMD only           SETcc         r         1         1         0.5         SSE4.A, AMD only           SETcc         m         1         0.5         I0/1         I0/1         I0/1           CLC, STC         1         0.5         I0/1         I0/1 <t< td=""><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td></t<>				2			
BTR, BTS         m,i         4-5         15           BTC         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11         6         7           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           LZCNT         r,r/m         8         5         SSE4.A, AMD only           SETcc         r         1         1         0.5         SSE4.A, AMD only           SETcc         m         1         0.5         IO/1         IO/1           CLC, STC         1         0.5         IO/1         IO/1         IO/1           CLD         1         1         1         IO         IO/1         IO/1	1			_			
BTC         m,r         8         16         13           BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11         6           POPCNT         r,r/m         9         12         5           LZCNT         r,r/m         8         5         SSE4.A/SSE4.2           SETcc         r         1         1         0.5           SETcc         m         1         1         0.5           SETcc         m         1         0.5         10/1           CLC, STC         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	1						
BTR, BTS         m,r         8         15         15           BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11         6         6           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           LZCNT         r,r/m         8         5         SSE4.A, AMD only           SETcc         r         1         1         0.5           SETcc         m         1         0.5         10/1           CLC, STC         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	1			40			
BSF, BSR         r,r         11         6         6           BSF, BSR         r,m         11         6         6           POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           LZCNT         r,r/m         8         5         SSE4.A/SSE4.2         SSE4.A, AMD only           SETcc         r         1         1         0.5         10/1         10/1           CLC, STC         1         0.5         10/1         10/1         10/1         10/1           CLD         1         1         1         10         10/1<	1						
BSF, BSR         r,m         11         6           POPCNT         r,r/m         9         12         5           LZCNT         r,r/m         8         5         SSE4.A/SSE4.2           SETcc         r         1         1         0.5           SETcc         m         1         1         1           CLC, STC         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	1	m,r					
POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           LZCNT         r,r/m         8         5         SSE4.A/SSE4.2           SETcc         r         1         1         0.5           SETcc         m         1         1         1           CLC, STC         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	BSF, BSR	r,r	11	6	6		
POPCNT         r,r/m         9         12         5         SSE4.A/SSE4.2           LZCNT         r,r/m         8         5         SSE4.A/SSE4.2           SETcc         r         1         1         0.5           SETcc         m         1         1         1           CLC, STC         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	BSF, BSR	r,m	11		6		
LZCNT         r,r/m         8         5         SSE4.A, AMD only           SETcc         r         1         1         0.5           SETcc         m         1         1         0.5         10/1           CLC, STC         1         1         0.5         10/1           CMC         1         1         0.5         10/1           CLD         1         1         1         10	1			12			SSE4.A/SSE4.2
SETcc         r         1         1         0.5           SETcc         m         1         1         1           CLC, STC         1         0.5         I0/1           CMC         1         1         0.5         I0/1           CLD         1         1         1         10							
SETcc         m         1         1         1           CLC, STC         1         0.5         I0/1           CMC         1         1         0.5         I0/1           CLD         1         1         1         I0					0.5		Joe Gravid Offing
CLC, STC     1     0.5     10/1       CMC     1     1     0.5     10/1       CLD     1     1     1     10	1			' '			
CMC         1         1         0.5         10/1           CLD         1         1         1         10	1	""				10.44	
CLD 1 1 10							
	1			1			
STD 2 10,11	1						
	STD		2		2	10,11	

Control transfer instru	ctions					
JMP	short/near	1		2		
JMP	r	1		2		
JMP	m(near)	1		2		
Jcc	short/near	1		1/2 - 2		recip. t. = 2 if jump
J(E/R)CXZ	short	2		1 - 2		recip. t. = 2 if jump
LÒOP	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	

FILD	m	1	9	1	FP1
FIST(T)(P)	m	1	6	1	
FLDZ, FLD1		1		1	FP1
FCMOVcc	st0,r	12	7	7	FP0/1
FFREE	r	1		1	FP1
FINCSTP, FDECSTP		1	1	1	FP1
FNSTSW	AX	2	~20	10	FP1
FNSTSW	m16	2	~20	10	FP1
FNSTCW	m16	3		2	FP0
FLDCW	m16	12		10	FP1
Arithmetic instruction	 				
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)	m	1	5	3	FP1
FIMUL	m	2			FP1
FDIV(R)(P)	r	1	19	19	FP1
FDIV(R)(P)	m	1		19	FP1
FIDIV(R)	m	2		19	FP1
FABS, FCHS	****	1	2	2	FP1
FCOM(P), FUCOM(P)	r	1	_	1	FP0
FCOM(P), FUCOM(P)	m	'1		1	FP0
FCOMPP, FUCOMPP	111	1		1	FP0
1	<b>r</b>	1	2	2	FP0
FCOM(P)	r				
FICOM(P)	m	2		1	FP0, FP1
FTST		1		1	FP0
FXAM		2	44	2	FP1
FRNDINT		5	11		FP0, FP1
FPREM		1	11-16		FP1
FPREM1		1	11-19		FP1
Math					
FSQRT		1	31		FP1
FLDPI, etc.		1		1	FP0
FSIN		4-44	27-105	27-105	FP0, FP1
FCOS		11-51	51-94	51-94	FP0, FP1
FSINCOS		11-75	48-110	48-110	FP0, FP1
FPTAN		~45	~113	~113	FP0, FP1
FPATAN		9-75	49-163	49-163	FP0, FP1
FSCALE		5	8		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
I = =		1	I		_, -,

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 FP0/1	
	xmm,mm		13	=	FP1	
MOVNTQ	m,mm	1		1.5		
MOVNTDQ	m,xmm	2	13	3	FP1	
PACKSSWB/DW PACKUSWB		_	4	0.5	ED0/4	
	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		_	4	0.5		
	mm,r/m	1	1	0.5		
PUNPCKH/LBW/WD/D			4	4		
Q	xmm,r/m	2	1 1	1	ED0 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	

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	·	1	2	2	FP0/1	
EXIRQ	xmm,xmm,i,i	ı			FPU/I	SSE4.A, AMD only
A!41						
Arithmetic instructions	S					
PADDB/W/D/Q PADDSB/W						
PADDUSB/W						
PSUBB/W/D/Q						
PSUBSB/W						
PSUBUSB/W	mm r/m	1	1	0.5	FP0/1	
	mm,r/m	ı	1	0.5	FPU/I	
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				· ·		Cumpl CCE2
	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	''
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	Саррі. ООСО
PSADBW	·	2	2	2	FP0, FP1	
FOADDW	xmm,r/m	2			FFU, FF1	
Logio						
PAND PANDN POR						
PXOR	mm r/m	4	4	0.5	FP0/1	
	mm,r/m	1	1	0.5	rPU/1	
PAND PANDN POR	\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\	0	4	4	ED0/4	
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	xmm,i/xmm/m	2	1	1	FP0/1		
PSLLDQ, PSRLDQ	xmm,i	2	1	1	FP0/1		
Other							
EMMS		1		0.5	FP0/1	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 ymm	3	6	2	FP0, FP1	I
CVT(1)FD2F1	mm,xmm xmm,r32	3	12	3	FP0, FP1	
CVTSI2SS CVTSI2SD	xmm,r32	2	11	3	FP1	
CVTSI2SD CVT(T)SS2SI	r32,xmm	2	12	1	FP0, FP1	
1 1		2	11			
CVT(T)SD2SI	r32,xmm			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	
B# - 41-						
Math SQRTSS	r r/m	1	14	14	FP1	
SQRTPS	r,r/m r,r/m	2	14 48	14 48	FP1	
SQRTSD	r,r/m	1	24	24	FP1	
SQRTPD		2	48	48	FP1	
RSQRTSS	r,r/m r,r/m	1	3	1	FP1	
RSQRTPS	r,r/m	2	3	2	FP1	
NOUNTES	1,1/111				FFI	
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:

gh- 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

instruction of the same kind can begin to execute. A value of 1/2 indicates that the execution units can handle 2 instructions per clock cycle in one thread. How-

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

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

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

different execution pipes.

Integer instructions

Instruction	Operands	Ops	Latency	Reciprocal throughput	Execution pipe	Notes
Move instructions						
MOV	r,r	1	1	0.5	10/1	
MOV	r,i	1		0.5	I0/1	
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	I0/1	
MOVZX, MOVSX	r,m	1	4	1		
MOVSXD	r64,r32	1	1	0.5		

MOVSXD	r64,m32	1	3	1		
CMOVcc	r,r	1	1	0.5	IO/1	
CMOVcc	r,m	1	•	1	107 1	
XCHG	r8,r8	3	2	2	10/1	
XCHG	r,r	2	1	1	10/1	
		_				Timing depends on
XCHG	r,m	3	16			hw
XLAT		2	5	3		
PUSH	r	1		1		
PUSH	i	1		1		
PUSH	m	2		1		
PUSH	SP	2		1		
PUSHF(D/Q)		9		6		
PUSHA(D)		9		8		
POP	r	1		1		
POP	m	3		2		
POP	SP	1		2		
1	SF					
POPF(D/Q)		29		18		
POPA(D)		9	_	8		
LEA	r16,[m]	2	3	2	10	Any address size
LEA	r32/64,[m]	1	1	0.5	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	I0/1	RIP relative
LAHF		4	3	2		
SAHF		1	1	0.5	IO/1	
SALC		1	1	1	-	
BSWAP	r	1	1	0.5	IO/1	
MOVBE	r,m	1	•	1	10/1	MOVBE
MOVBE		1		1		MOVBE
PREFETCHNTA	m,r	1		~100	AGU	IVIOVE
1	m					
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	S					
ADD, SUB	r,r/i	1	1	0.5	IO/1	
ADD, SUB	r,m	1		1		
ADD, SUB	m,r	1	6	1		
ADC, SBB	r,r/i	1	1	1	IO/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	10/1	
I	r,m r	1	4		10/1	
INC, DEC, NEG	r		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		

			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	
				1		
IMUL	r16,r16/m16	1	3	-	10	
IMUL	r32,r32/m32	1	3	1	10	
IMUL	r64,r64/m64	1	6	4	10	
IMUL	r16,(r16),i	2	4	1	10	
IMUL	r32,(r32),i	1	3	1	10	
IMUL	r64,(r64),i	1	6	4	10	
DIV	r8/m8	1	11-14	11-14	10	
DIV	r16/m16	2	12-19	12-19	10	
DIV	r32/m32	2	12-13	12-13	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	r32/m32	2	12-27	12-27	10	
IDIV	r64/m64	2	12-43	12-43	10	
CBW, CWDE, CDQE		1	1		IO/1	
CWD, CDQ, CQO		1	1		10/1	
31.2, 324, 343		•	·		10, 1	
Logio instructions						
Logic instructions	<u>.</u> :	4	4	0.5	10/4	
AND, OR, XOR	r,i	1	1	0.5	10/1	
AND, OR, XOR	r,r	1	1	0.5	I0/1	
AND, OR, XOR	r,m	1		1		
AND, OR, XOR	m,r	1	6	1		
ANDN	r,r,r	1	1	0.5		BMI1
ANDN	r,r,m	2		1		BMI1
TEST	r,i	1	1	0.5	I0/1	
TEST	r,r	1	1	0.5	10/1	
TEST	r,m	1	·	1	10, 1	
NOT		1	1	0.5	IO/1	
1	r				10/1	
NOT	m	1	6	1	10/4	
SHL, SHR, SAR	r,i/CL	1	1	0.5	10/1	
ROL, ROR	r,i/CL	1	1	0.5	I0/1	
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,	.,	-				
ROR	m,i /CL	1	6	1		
RCL, RCR	m,1	1		1		
1		-				
RCL	m,i	10		11		
RCR	m,i	9		11		
RCL	m,CL	9		11		
RCR	m,CL	8		11		
SHLD, SHRD	r,r,i	6	3	3		
SHLD, SHRD	r,r,cl	7	4	4		
SHLD, SHRD	m,r,i/CL	8		11		
BT	r,r/i	1		0.5		
BT				1		
1	m,i	1				
BT	m,r	5		3		

BTC, BTR, BTS BTC BTR, BTS BTC, BTR, BTS BSF, BSR BSF, BSR POPCNT LZCNT TZCNT BLSI BLSR BLSI BLSR BLSMSK BLSMSK BEXTR BEXTR SETCC SETCC CLC, STC CMC CLD STD	r,r/i m,i m,i m,r r,r r,r r,m r,r/m r,r r,r r,m r,r r,m r,r,r r,m r,r,r r,m r,r,r	2 5 4 8 7 8 8 1 1 2 2 3 2 3 1 1 1 1 1 1 1 1 1 1 1 1 1	2 4 4 1 1 2 2 2 1 1	1 11 11 4 4 4 0.5 0.5 1 1 2 0.5 1 0.5 1 0.5 1	IO/1 IO/1 IO IO,I1	SSE4A/SSE4.2 SSE4A/LZCNT BMI1 BMI1 BMI1 BMI1 BMI1 BMI1 BMI1
Control transfer instru						
JMP	short/near	1		2		
JMP	r	1		2 2		
JMP	m(near)	1				0 16 1,
Jcc	short/near	1		0.5 - 2		2 if jumping
J(E/R)CXZ	short	2		1 - 2		2 if jumping
LOOP	short	8		5		
LOOPE LOOPNE	short	10		6		
CALL	near	2		2		
CALL	r	2		2		
CALL	m(near)	5		2		
RET		1		3		
RET	i	4		3		
						values are for no
BOUND	m	8		4		jump
INTO		,		_		values are for no
INTO		4		2		jump
String instructions						
LODS		4		2		
REP LODS		~5n		~3n		
STOS		4		2		
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		
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	

			Ū			
FLDCW	m16	12		9	FP1	
Arithmetic instructions						
		1		4	FP0	
FADD(P),FSUB(R)(P)	r	1	3	1	-	
FADD(P),FSUB(R)(P)	m	1		1	FP0	
FIADD,FISUB(R)	m	2	_	2	FP0,FP1	
FMUL(P)	r	1	5	3	FP1	
FMUL(P)	m 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	80	136	136	FP0, FP1	

# **Integer MMX and XMM 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	

140)		1 4			1 4011	1 1
MOVD	mm,m32	1	4	1	AGU	
MOVD	r32, x	1	4	1	FP0	
MOVD	x, r32	2	6	1	FP1	
MOVD	x,m32	1	4	1	AGU	
MOVD	m32,(x)mm	1	3	1	FP1	
MOVED (MOVE)			_	4	ED0	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	,				-D0//	
PACKUSWB	x,r/m	1	2	0.5	FP0/1	
PUNPCKH/LBW/WD/D Q		_		0.5	ED0/4	
PUNPCKH/LBW/WD/D	mm,r/m	1	1	0.5	FP0/1	
Q	x,r/m	1	2	0.5	FP0/1	
PUNPCKH/LQDQ			2	0.5	FP0/1	
PSHUFB	x,r/m	1 1	1	0.5	FP0/1	Suppl. SSE3
PSHUFB	mm,mm	3	4	2	FP0/1	Suppl. SSE3
PSHUFD	X,X X,X,i	1	2	0.5	FP0/1	Suppl. SSL3
PSHUFW	mm,mm,i	1	1	0.5	FP0/1	
PSHUFL/HW	x,x,i	1	1 1	0.5	FP0/1	
PALIGNR	x,x,i	1	2	0.5	FP0/1	Suppl. SSE3
PBLENDW	x,r/m	1	1	0.5	FP0/1	SSE4.1
MASKMOVQ	mm,mm	32	432	17	FP0, FP1	OOL4.1
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	1	FP0/1	
PINSRB/W/D/Q	x,r,i	2	7	1	FP0/1	
PINSRB/W/D/Q	x,m,i	1		1	FP0/1	
PEXTRB/W/D/Q	r,x,i	1	3	1	FP0	SSE4.1
PEXTRB/W/D/Q	m,x,i	1		1	FP1	SSE4.1
INSERTQ	x,x	3	2	2	FP0, FP1	SSE4A, AMD only
INSERTQ	x,x,i,i	3	2	2	FP0, FP1	SSE4A, AMD only
EXTRQ	x,x	1	1 1	0.5	FP0/1	SSE4A, AMD only

			ouge			
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/	7.,72		_			
WD/WQ/DQ	x,x	1	2	0.5	FP0/1	SSE4.1
Arithmetic instruction	 <b>S</b>					
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	(x)mm,r/m	1	2	1	FP0	Suppl. SSE3
PMADDWD	(x)mm,r/m	1	2	1	FP0	
PMADDUBSW	(x)mm,r/m	1	2	1	FP0	Suppl. SSE3
PAVGB/W	(x)mm,r/m	1	1	0.5	FP0/1	
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 PSRAW/D	x,x	1	2	0.5	FP0/1	
PSLL/RL W/D/Q						
PSRAW/D	x,i	1	1	0.5	FP0/1	
PSLLDQ, PSRLDQ	x,i	1	2 3	0.5	FP0/1	0054.4
PTEST	x,x/m	1	3	1	FP0	SSE4.1
String instructions						
PCMPESTRI	x,x,i	9	5	5	FP0/1	SSE4.2
PCMPESTRI	x,m,i	10		5	FP0/1	SSE4.2
PCMPESTRM	x,x,i	9	9	9	FP0/1	SSE4.2
PCMPESTRM	x,m,i	10		9	FP0/1	SSE4.2
PCMPISTRI	x,x,i	3	2	2	FP0/1	SSE4.2
PCMPISTRI	x,m,i	4		2	FP0/1	SSE4.2

PCMPISTRM	x,x,i	3	8	8	FP0/1	SSE4.2
PCMPISTRM	x,m,i	4		2	FP0/1	SSE4.2
Encryption						
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 XMN Instruction	Operands	Ops	Latency	Reciprocal	Execution	Notes
	Operanas	Оро	Laterioy	throughput	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

			3.3			
SHUFPS/D	x,x/m,i	1	2	0.5	FP0/1	
VSHUFPS/D	y,y,y,i	2	2	1	FP0/1	AVX
UNPCK H/L PS/D	x,x/m	1	2	0.5	FP0/1	
VUNPCK H/L PS/D	y,y,y	2	2	1	FP0/1	AVX
EXTRACTPS	r32,x,i	1	3	1	FP0	
EXTRACTPS	m32,x,i	1	3	1	FP1	
VEXTRACTF128	x,y,i	1	1	0.5	FP0/1	AVX
VEXTRACTF128	m128,y,i	1	12	1	FP1	AVX
INSERTPS	x,x,i	1		1	FP0/1	
INSERTPS	x,m32,i	1	6	1	FP0/1	
VINSERTF128	y,y,x,i	2	1	1	FP0/1	AVX
VINSERTF128	y,y,m128,i	2	13	2	FP0/1	AVX
VMASKMOVPS/D	x,x,m128	1	15	1	FP0/1	>300 clk if mask=0
VMASKMOVPS/D	y,y,m256	2	15	2	FP0/1	>300 clk if mask=0
VMASKMOVPS/D	m128,x,x	19	21	16	FP1	AVX
VMASKMOVPS/D	m256,y,y	36	32	22	FP1	AVX
Conversion						
CVTPS2PD	x,x/m	1	3	1	FP1	
VCVTPS2PD	y,x/m	2	4	2	FP1	
CVTPD2PS	x,x/m	1	4	1	FP1	
VCVTPD2PS	x,y	3	6	2	FP0, FP1	
CVTSD2SS	x,x/m	2	5	8	FP1	
CVTSS2SD	x,x/m	2	4	7	FP1	
CVTDQ2PS/PD	x,x/m	1	4	1	FP1	
VCVTDQ2PS/PD	y,y	2	4	2	FP1	
CVT(T)PS2DQ	x,x/m	1	4	1	FP1	
VCVT(T)PS2DQ	y,y	2	4	2	FP1	
CVT(T)PD2DQ	x,x/m	1	4	1	FP1	
VCVT(T)PD2DQ	y,y	3	7	2	FP1	
CVTPI2PS	xmm,mm	1	4	1	FP1	
CVTPI2PD	xmm,mm	1	4	1	FP1	
CVT(T)PS2PI	mm,xmm	1	4	1	FP1	
CVT(T)PD2PI	mm,xmm	1	4	1	FP1	
CVTSI2SS	xmm,r32	2	9	1	FP1	
CVTSI2SD	xmm,r32	2	9	1	FP1	
CVT(T)SS2SI	r32,xmm	2	8	1	FP1	
CVT(T)SD2SI	r32,xmm	2	8	1	FP1	
VCVTPS2PH	x/m,x,i	1	4	1	FP1	F16C
VCVTPS2PH	x/m,y,i	3	6	2	FP0, FP1	F16C
VCVTPH2PS	x,x/m	1	4	1	FP1	F16C
VCVTPH2PS	y,x/m	2	5	2	FP1	F16C
Arithmetic						
ADDSS/D SUBSS/D	x,x/m	1	3	1	FP0	
ADDPS/D SUBPS/D	x,x/m	1	3	1	FP0	
VADDPS/D VSUBPS/D	y,y/m	2	3	2	FP0	
ADDSUBPS/D	x,x/m	1	3	1	FP0	SSE3
VADDSUBPS/D	y,y/m	2	3	2	FP0	
HADD/SUBPS/D	x,x/m	1	4	1	FP0	SSE3
VHADD/SUBPS/D	y,y/m	2	4	2	FP0	
MULSS/PS	x,x/m	1	2	1	FP1	
VMULPS	y,y/m	2	2	2	FP1	

MULSD/PD		ı			i		
DIVSS	MULSD/PD	x,x/m	1	4		FP1	
DIVPS	VMULPD	y,y/m	2	4	2	FP1	
VDIVPS	DIVSS	x,x/m	1	14	14	FP1	
DIVPS	DIVPS	x,x/m	1	19	19	FP1	
DIVSD	VDIVPS		2	38	38	FP1	
DIVPD							
VDIVPD		-					
RCPSS		-					
RCPPS							
VRCPPS		-	1				
MAXSS/D MINSS/D MAXPS/D MINPS/D VMAXPS/D VMINPS/D VMAXPS/D VMINPS/D VMAXPS/D VMINPS/D VMAXPS/D VMINPS/D VMAXPS/D VMINPS/D VMAXPS/D VMINPS/D VMAXPS/D VMINPS/D VMAXPS/D VMINPS/D VXMPCcPS/D VVMPCcPS/D VVMPCcPS/D VVMPCcPS/D VVMPCcPS/D VVMPCcPS/D VVMPCcPS/D VVMPCcPS/D VVMMS/S/D/PS/D VXX,/m         1         2         1         FP0 FP0 FP0 FP0 FP0 VMAXPS/D VX,x/m         1         2         1         FP0 FP0 FP0 FP0 FP0 VMAXPS/D VX,x/m         1         2         2         2         2         FP0 FP0 FP0 FP0 FP0 FP1 SSE4.1         5         11         4         1         FP1 FP0 FP1 SSE4.1         5         11         4         1         FP1 FP0 FP1 SSE4.1         5         11         4         FP0 FP1 FP0 FP1 SSE4.1         SSE4.1         1         5         11         4         FP0 FP1 FP0 FP1 SSE4.1         SSE4.1         1         1         4         FP0 FP1 FP0 FP1 SSE4.1         SSE4.1         1		-					
MAXPS/D MINPS/D VMAXPS/D VMINPS/D VMAXPS/D VMINPS/D CMPccSS/D         x,x/m         1         2         1         FP0 FP0 FP0 FP0 VCMPccPS/D           CMPccPS/D VCMPccPS/D         x,x/m         1         2         1         FP0 FP0 FP0 VCMPccPS/D           VCMPccPS/D VCMPccPS/D         y,y/m         2         2         2         FP0 FP0 FP0 FP0 FP0 FP0 FP0 FP0 FP0 FP0							
VMAXPS/D VMINPS/D CMPccSS/D         y,y/m x,x/m         2         2         2         FP0 TP0 TP0 CMPccPS/D         PP0 X,x/m         PP0 TP0 TP0 TP0 VCMPccPS/D         PP0 Y,y/m         PP0 TP0 TP0 TP0 TP0 TP0 TP0 TP0 TP0 TP0	_ I				1		
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         FP0           (U)COMISS/D         x,x/m         1         4         1         FP1           ROUNDSS/D/PS/D         y,y/m,i         1         4         1         FP1           VROUNDSS/D/PS/D         y,y/m,i         1         4         1         FP1           VROUNDSS/D/PS/D         y,y/m,i         1         4         FP0, FP1         SSE4.1           DPPS         x,x,i         6         4         FP0, FP1         SSE4.1           VDPPS         y,y,y,i         10         12         7         FP0, FP1         SSE4.1           VDPPS         y,y,m,i         12         7         FP0, FP1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1         SSE4.1           Logic         ANDPS/D ANDNPS/D         X,x/m         1         1         0.5         FP0/1           NATH         Y,y/m         2         1         1         FP0, FP1         SSE4.1	MAXPS/D MINPS/D	x,x/m				FP0	
CMPccPS/D	VMAXPS/D VMINPS/D	y,y/m	2		2	FP0	
VCMPccPS/D (U)COMISS/D         y,y/m x,x/m         2         2         2         FP0 FP0 FP0 ROUNDSS/D/PS/PD           ROUNDSS/D/PS/PD VROUNDSS/D/PS/D DPPS         x,x/m,i         1         4         1         FP1 FP1 VROUNDSS/D/PS/D DPPS         x,x/ii         5         11         4         FP0, FP1 FP1 FP0, FP1 SSE4.1         SSE4.1           DPPS         x,m,i         6         4         FP0, FP1 FP0, FP1 SSE4.1         SSE4.1           VDPPS         y,y,m,i         12         7         FP0, FP1 FP0, FP1 SSE4.1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1 SSE4.1         SSE4.1           Logic	CMPccSS/D	x,x/m	1	2	1	FP0	
VCMPccPS/D (U)COMISS/D         y,y/m x,x/m         2         2         2         FP0 FP0           ROUNDSS/D/PS/PD VROUNDSS/D/PS/D DPPS         x,x/m,i         1         4         1         FP1 FP1           DPPS         x,x,i         5         11         4         FP0, FP1 FP0, FP1         SSE4.1           DPPS         x,m,i         6         4         FP0, FP1 FP0, FP1         SSE4.1           VDPPS         y,y,m,i         12         7         FP0, FP1 FP0, FP1         SSE4.1           VDPPS         y,m,i         12         7         FP0, FP1 FP0, FP1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1         SSE4.1           DPPD         x,x,m,i         4         3         FP0, FP1         SSE4.1           Logic         X,x/m         1         1         0.5         FP0/1           ANDPS/D ANDNPS/D ORPS/D XORPS/D         x,x/m         1         1         FP0, FP1         SSE4.1           Logic         X,x/m         2         1         1         FP0, FP1         SSE4.1           SQRTSS         x,x/m         2         1         1         FP0, FP1         FP1           SQRTPS <td>CMPccPS/D</td> <td>x,x/m</td> <td>1</td> <td>2</td> <td>1</td> <td>FP0</td> <td></td>	CMPccPS/D	x,x/m	1	2	1	FP0	
(U)COMISS/D		-	2	2	2		
RÓUNDSS/SD/PS/PD VROUNDSS/D/PS/D DPPS         x,x/m,i         1         4         1         FP1 FP1 FP1 FP1 SSE4.1           DPPS DPPS         x,x,i         5         11         4         FP0, FP1 FP0, FP1 SSE4.1         SSE4.1           DPPS VDPPS         y,y,y,i         10         12         7         FP0, FP1 FP0, FP1 SSE4.1         SSE4.1           DPPD VDPPS         y,m,i         12         7         FP0, FP1 SSE4.1         SSE4.1           DPPD DPD         x,x,i         3         9         3         FP0, FP1 SSE4.1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1 SSE4.1         SSE4.1           DPD         x,x,i         3         9         3         FP0, FP1 SSE4.1         SSE4.1           Logic         ANDPS/D ANDNPS/D ORPS/D XORPS/D VANDPS/D, etc.         x,x/m         1         1         0.5         FP0/1 FP0, FP1           SQRTSS         x,x/m         2         21         21         FP1           SQRTSS         x,x/m         2         21         21         FP1           SQRTSD         x,x/m         1         27         27         FP1           SQRTSD         x,x/m         2         24				_			
VROUNDSS/D/PS/D         y,y/m,i         2         4         2         FP1         SSE4.1           DPPS         x,x,i         5         11         4         FP0, FP1         SSE4.1           DPPS         x,m,i         6         4         FP0, FP1         SSE4.1           VDPPS         y,y,y,i         10         12         7         FP0, FP1         SSE4.1           VDPPS         y,m,i         12         7         FP0, FP1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1         SSE4.1           DPPD         x,x,ii         4         3         FP0, FP1         SSE4.1           Logic         ANDPS/D ANDNPS/D ORPS/D VANDPS/D VANDPS/D VANDPS/D VANDPS/D, etc.         x,x/m         1         1         0.5         FP0/1         SSE4.1           Math         SQRTSD         x,x/m         1         1         0.5         FP0/1         FP0/1         SSE4.1           WSQRTPS         x,x/m         1         16         16         FP1         FP0/1	` '			4			
DPPS         x,x,i         5         11         4         FP0, FP1         SSE4.1           DPPS         x,m,i         6         4         FP0, FP1         SSE4.1           VDPPS         y,y,y,i         10         12         7         FP0, FP1         SSE4.1           VDPPS         y,m,i         12         7         FP0, FP1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1         SSE4.1           DPPD         x,x,m,i         4         3         FP0, FP1         SSE4.1           DPD         x,x/m         1         1         0.5         FP0, FP1         SSE4.1           Logic         x,x/m         1         1         0.5         FP0, FP1         SSE4.1           Logic         x,x/m         2         21         <							
DPPS         x,m,i         6         4         FP0, FP1         SSE4.1           VDPPS         y,y,y,i         10         12         7         FP0, FP1         SSE4.1           VDPPS         y,m,i         12         7         FP0, FP1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1         SSE4.1           DPPD         x,x,m,i         4         3         FP0, FP1         SSE4.1           Logic         x,x/m         1         1         0.5         FP0/1           WANDPS/D         X,x/m         1         1         1         1         FP0/1           WSQRTSS         x,x/m         2         21         2         FP1 <tr< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td>0054.4</td></tr<>							0054.4
VDPPS         y,y,y,i         10         12         7         FP0, FP1         SSE4.1           VDPPS         y,m,i         12         7         FP0, FP1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1         SSE4.1           DPPD         x,x,m,i         4         3         FP0, FP1         SSE4.1           Logic         ANDPS/D ANDNPS/D ORPS/D VANDPS/D, etc.         x,x/m         1         1         0.5         FP0/1           VANDPS/D, etc.         y,y/m         2         1         1         FP0/1           Math         x,x/m         1         16         16         FP1           SQRTSS         x,x/m         2         21         21         FP1           SQRTPS         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				11		· ·	
VDPPS         y,m,i         12         7         FP0, FP1         SSE4.1           DPPD         x,x,i         3         9         3         FP0, FP1         SSE4.1           DPPD         x,xm,i         4         3         FP0, FP1         SSE4.1           Logic         ANDPS/D ANDNPS/D ORPS/D VANDPS/D, etc.         x,x/m         1         1         0.5         FP0/1           Math         SQRTSS         x,x/m         2         1         1         FP0/1           SQRTSS         x,x/m         2         21         21         FP1           SQRTPS         x,x/m         2         21         21         FP1           SQRTSD         x,x/m         1         27         27         FP1           SQRTSD         x,x/m         2         27         27         FP1           SQRTPD         x,x/m         2         27         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Oth						· ·	
DPPD		y,y,y,i		12		· ·	
DPPD		y,m,i					
Logic	DPPD	x,x,i	3	9		FP0, FP1	SSE4.1
ANDPS/D ANDNPS/D ORPS/D X,x/m 1 1 1 0.5 FP0/1 VANDPS/D, etc.	DPPD	x,m,i	4		3	FP0, FP1	SSE4.1
ORPS/D XORPS/D VANDPS/D, etc.         x,x/m         1         1         0.5         FP0/1           Math         SQRTSS         x,x/m         1         16         16         FP1           SQRTPS         x,x/m         2         21         21         FP1           VSQRTPS         y,y/m         2         42         42         FP1           SQRTSD         x,x/m         1         27         27         FP1           SQRTPD         x,x/m         2         27         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Other         D         8         FP0, FP1         FP1           VZEROUPPER         21         30         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         46	Logic						
ORPS/D XORPS/D VANDPS/D, etc.         x,x/m         1         1         0.5         FP0/1           Math         SQRTSS         x,x/m         1         16         16         FP1           SQRTPS         x,x/m         2         21         21         FP1           VSQRTPS         y,y/m         2         42         42         FP1           SQRTSD         x,x/m         1         27         27         FP1           SQRTPD         x,x/m         2         27         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Other         D         8         FP0, FP1         FP1           VZEROUPPER         21         30         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         46	ANDPS/D ANDNPS/D						
Wath         x,x/m         1         16         16         FP1           SQRTSS         x,x/m         2         21         21         FP1           SQRTPS         x,x/m         2         21         21         FP1           VSQRTPS         y,y/m         2         42         42         FP1           SQRTSD         x,x/m         1         27         27         FP1           SQRTPD         x,x/m         2         27         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Other           LDMXCSR         m         3         13         12         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode		x,x/m	1	1	0.5	FP0/1	
Math         X, x/m         1         16         16         FP1           SQRTPS         X, x/m         2         21         21         FP1           VSQRTPS         X, x/m         2         42         42         FP1           SQRTSD         X, x/m         1         27         27         FP1           SQRTPD         X, x/m         2         27         27         FP1           VSQRTPD         Y, y/m         2         54         54         FP1           RSQRTSS/PS         X, x/m         1         2         1         FP1           VRSQRTPS         Y, y/m         2         2         2         FP1           Other           LDMXCSR         m         1         2         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	VANDPS/D. etc.	-				FP0/1	
SQRTSS         x,x/m         1         16         16         FP1           SQRTPS         x,x/m         2         21         21         FP1           VSQRTPS         y,y/m         2         42         42         FP1           SQRTSD         x,x/m         1         27         27         FP1           SQRTPD         x,x/m         2         27         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           VRSQRTPS         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         37         46         64 bit mode           V	1	3,3					
SQRTPS         x,x/m         2         21         21         FP1           VSQRTPS         y,y/m         2         42         42         FP1           SQRTSD         x,x/m         1         27         27         FP1           SQRTPD         x,x/m         2         27         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Other           LDMXCSR         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	Math						
VSQRTPS         y,y/m         2         42         42         FP1           SQRTSD         x,x/m         1         27         27         FP1           SQRTPD         x,x/m         2         27         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Other           LDMXCSR         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	SQRTSS	x,x/m	1	16	16	FP1	
SQRTSD         x,x/m         1         27         27         FP1           SQRTPD         x,x/m         2         27         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Other           LDMXCSR         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	SQRTPS	x,x/m	2	21	21	FP1	
SQRTSD       x,x/m       1       27       27       FP1         SQRTPD       x,x/m       2       27       27       FP1         VSQRTPD       y,y/m       2       54       54       FP1         RSQRTSS/PS       x,x/m       1       2       1       FP1         VRSQRTPS       y,y/m       2       2       2       FP1         Other         LDMXCSR       m       12       9       8       FP0, FP1         STMXCSR       m       3       13       12       FP0, FP1         VZEROUPPER       21       30       32 bit mode         VZEROALL       41       58       32 bit mode         VZEROALL       73       90       64 bit mode	VSQRTPS	y,y/m	2	42	42	FP1	
SQRTPD         x,x/m         2         27         FP1           VSQRTPD         y,y/m         2         54         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Other           LDMXCSR         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode							
VSQRTPD         y,y/m         2         54         FP1           RSQRTSS/PS         x,x/m         1         2         1         FP1           VRSQRTPS         y,y/m         2         2         2         FP1           Other           LDMXCSR         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROHL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode		-					
RSQRTSS/PS VRSQRTPS         x,x/m y,y/m         1 2 2 2         2 2 2         1 5 2 2         FP1 FP1           Other         LDMXCSR LDMXCSR         m         12 3 3 13         9 12 12         8 5 12         FP0, FP1 FP0, FP1           STMXCSR VZEROUPPER VZEROUPPER VZEROALL VZEROALL VZEROALL VZEROALL         37 46 41 41 41 58 90         46 32 32 32 32 36 34 36 36 36 36 36 36 36 36 36 36 36 36 36		-					
VRSQRTPS         y,y/m         2         2         2         FP1           Other         DMXCSR         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROUPPER         37         46         64 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode							
Other         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROUPPER         37         46         64 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	1		1				
LDMXCSR         m         12         9         8         FP0, FP1           STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROUPPER         37         46         64 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	VKSQKIFS	y, y/111			2	FFI	
STMXCSR         m         3         13         12         FP0, FP1           VZEROUPPER         21         30         32 bit mode           VZEROUPPER         37         46         64 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	Other						
VZEROUPPER         21         30         32 bit mode           VZEROUPPER         37         46         64 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	LDMXCSR	m	12	9	8	FP0, FP1	
VZEROUPPER         21         30         32 bit mode           VZEROUPPER         37         46         64 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode	STMXCSR	m	3	13	12	FP0, FP1	
VZEROUPPER         37         46         64 bit mode           VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode							32 bit mode
VZEROALL         41         58         32 bit mode           VZEROALL         73         90         64 bit mode							
VZEROALL 73 90 64 bit mode							
FXSAVE   66   66   32 bit mode	FXSAVE		66	66	66		32 bit mode
FXSAVE 58 58 64 bit mode							
FXRSTOR 115 189 189 32 bit mode							
FXRSTOR   123   198   197   64 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 .		1
Instruction	Operands	Clock cycles	Pairability
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
MUL IMUL	all other versions	9 d)	np
DIV	r8/m8	17	np
DIV	r16/m16	25	np
DIV	r32/m32	41	np
IDIV	r8/m8	22	np
IDIV	r16/m16	30	np
IDIV	r32/m32	46	np
CBW CWDE		3	np
CWD CDQ		2	np
SHR SHL SAR SAL	r,i	1	u
SHR SHL SAR SAL	m, i	3	u
SHR SHL SAR SAL	r/m, CL	4/5	np
ROR ROL RCR RCL	r/m, 1	1/3	u u
ROR ROL	r/m, i(><1)	1/3	np
ROR ROL	r/m, CL	4/5	np
RCR RCL	r/m, i(><1)	8/10	np
RCR RCL	r/m, CL	7/9	np
SHLD SHRD	r, i/CL	4 a)	np
SHLD SHRD	m, i/CL	5 a)	np
BT	r, r/i	4 a)	np
BT	m, i	4 a)	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)	· ·
SETcc	r/m	1/2 a)	np
JMP CALL	short/near	1/2 a) 1 e)	np v
JMP CALL	far	>= 3 e)	
	short/near	,	np
conditional jump CALL JMP	r/m	1/4/5/6 e)	V
	1/111	2/5 e	np
RETN		2/5 e	np
RETN	i	3/6 e)	np
RETF	·	4/7 e)	np
RETF	j	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
CPUID		13-16 a)	np
1	ı	.5 .5 4)	۲۳

RDTSC	6-13 a) j) np
Notes:	
а	This instruction has a 0FH prefix which takes one clock cycle extra to decode on a P1 unless preceded by a multi-cycle instruction.
b	versions with FS and GS have a 0FH prefix. see note a.
С	versions with SS, FS, and GS have a 0FH prefix. see note a.
d	versions with two operands and no immediate have a 0FH prefix, see
е	ମିଫୁଟି ∛alues are for mispredicted jumps/branches.
f	only pairable if register is AL, AX or EAX.
g	add one clock cycle for decoding the repeat prefix unless preceded by a multi-cycle instruction (such as CLD).
h	pairs as if it were writing to the accumulator.
i	9 if SP divisible by 4 (imperfect pairing).
j	on P1: 6 in privileged or real mode; 11 in non-privileged; error in virtual mode. On PMMX: 8 and 13 clocks respectively.

#### Floating point instructions (Pentium and Pentium MMX)

Explanation of column headings

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

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

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	Dperands μops							Reciprocal
		р0	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)											
PUSHA(D)   POPA(D)   LAHF SAHF   LEA	PUSHF(D)		3		11		1	1			
PUSHA(D)   POPA(D)   POP	POPF(D)		10		6	1					
POPA(D)	PUSHA(D)				2		8	8			
LAHF SAHF  LEA  LEA  LEA  LES  S  M  ADD SUB AND OR XOR  ADD SUB A	1 ' '				2	8					
LEA	1 1				1						
LDS LES LFS LGS LSS  M ADD SUB AND OR XOR ADD SUB A		r.m	1						1 c)		
LSS		,,							,		
ADD SUB AND OR XOR ADC SBB  ACC SBB ACC SC ACC SBB ACC SC ACC SBB ACC SC ACC SBB ACC SC ACC		m			8	3					
ADD SUB AND OR XOR ADD SUB AND OR XOR ADD SUB AND OR XOR ADC SBB Cr,rfi ADC SBB CMP TEST CMP											
ADD SUB AND OR XOR ADC SBB ADC SBB CMP TEST CMP TEST INC DEC NEG NOT AAA AAS DAA DAS AAM INC						1					
ADC SBB						-	1	1			
ADC SBB						'	'	'			
ADC SBB  CMP TEST  CMP TEST  INC DEC NEG NOT  INC DEC NEG NOT  INC DEC NEG NOT  INC DEC NEG NOT  AAA AAS DAA DAS  AAM  AAM  AAM  I 1 2 4 4 4  AAM  I 1 2 4 15  IMUL  INUL  INU						4					
CMP TEST						-		_			
CMP TEST						1	1	1			
INC DEC NEG NOT	l l										
INC DEC NEG NOT						1					
AAA AAS DAA DAS  AAD  AAM  AAM  I  I  I  I  I  I  I  I  I  I  I  I  I		r									
AAD  AAM  AAM  AAM  I  I  I  I  I  I  I  I  I  I  I  I  I		m			1	1	1	1			
AAM    MUL				1							
IMUL	AAD		1		2				4		
IMUL         (r),m         1         1         1         4         1           DIV IDIV         r8         2         1         1         19         12           DIV IDIV         r16         3         1         23         21           DIV IDIV         m8         2         1         1         19         12           DIV IDIV         m6         2         1         1         23         21           DIV IDIV         m62         2         1         1         23         21           DIV IDIV         m32         2         1         1         1         23         21           DIV IDIV         m32         2         1         1         1         1         1         1         1         1         1         1 <td>AAM</td> <td></td> <td>1</td> <td>1</td> <td>2</td> <td></td> <td></td> <td></td> <td>15</td> <td></td> <td></td>	AAM		1	1	2				15		
DIV IDIV   R8   2	IMUL	r,(r),(i)	1						4	1	
DIV IDIV	IMUL	(r),m	1			1			4	1	
DIV IDIV   DIV	DIV IDIV	r8	2		1				19	12	
DIV IDIV   DIV	DIV IDIV	r16	3		1				23	21	
DIV IDIV   DIV	DIV IDIV	r32	3		1				39	37	
DIV			1		1	1					
DIV IDIV         m32         2         1         1         39         37           CBW CWDE         CWD CDQ         1 <td></td> <td></td> <td></td> <td></td> <td></td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td>						1					
CBW CWDE CWD CDQ SHR SHL SAR ROR ROL ROL ROL ROL ROL ROL ROL ROL ROL			1			-					
CWD CDQ SHR SHL SAR ROR ROL ROL ROL ROL ROL ROL ROL ROL ROL			_								
SHR SHL SAR ROR       r,i/CL       1         ROL       m,i/CL       1       1       1       1         SHR SHL SAR ROR       m,i/CL       1       1       1       1         ROL       m,i/CL       1       1       1       1       1         RCR RCL       r,1       1			1		'						
ROL SHR SHL SAR ROR ROL ROL ROL ROL RCR RCL RCL RCR RC			'								
SHR SHL SAR ROR   ROL   m,i/CL   1   1   1   1   1   1   1   1   1		r i/Cl	1								
ROL       m,i/CL       1       1       1       1       1         RCR RCL       r,1       1       1       1       1       1         RCR RCL       r8,i/CL       4       4       4       4       4       8       8       RCR RCL       RCR RCL       RCR RCL       RCR RCL       M8,i/CL       4       3       1 <td< td=""><td></td><td>1,1/0L</td><td>'</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></td<>		1,1/0L	'								
RCR RCL       r,1       1       1       4       4       4       RCR RCL       r8,i/CL       4       4       4       RCR RCL       RCR RCL       r16/32,i/CL       3       3       3       3       RCR RCL       m,1       1       2       1		m i/Cl	1			4	4	4			
RCR RCL       r8,i/CL       4       4       4       1       4       4       4       4       4       8       4       4       3       3       3       3       3       3       3       3       3       3       1       <					4	1	'	'			
RCR RCL       r16/32,i/CL       3       3       1			-								
RCR RCL       m,1       1       2       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       1       1       1       1         BT       r,r/i       1       6       1       1         BT BTS BTC       r,r/i       1       6       1       1       1         BTR BTS BTC       m,r/i       1       6       1		· ·									
RCR RCL       m16/32,i/CL       4       2       1       1       1         SHLD SHRD       m,r,i/CL       2       1       1       1       1         BT       m,r/i       1       6       1       1         BT BTS BTC       r,r/i       1       6       1       1         BTR BTS BTC       m,r/i       1       6       1       1       1         BSF BSR       r,r       1       1       1       1       1       1       1         BSF BSR       r,m       1 </td <td></td> <td></td> <td></td> <td></td> <td>1</td> <td>-</td> <td></td> <td></td> <td></td> <td></td> <td></td>					1	-					
SHLD SHRD         r,r,i/CL         2         1         1         1           SHLD SHRD         m,r,i/CL         2         1         1         1           BT         r,r/i         1         6         1           BTR BTS BTC         r,r/i         1         6         1         1           BTR BTS BTC         m,r/i         1         6         1         1         1           BSF BSR         r,r         1         1         1         1         1           BSF BSR         r,m         1         1         1         1         1					1	-					
SHLD SHRD         m,r,i/CL         2         1         1         1           BT         r,r/i         1         6         1           BT BTS BTC         r,r/i         1         6         1           BTR BTS BTC         m,r/i         1         6         1         1           BTR BTS BTC         m,r/i         1         1         1         1           BSF BSR         r,r         1         1         1         1           BSF BSR         r,m         1         1         1         1					2	1	1	1			
BT         r,r/i         1 <td></td>											
BT         m,r/i         1         6         1           BTR BTS BTC         r,r/i         1         1         1           BTR BTS BTC         m,r/i         1         6         1         1         1           BSF BSR         r,r         1         1         1         1         1           BSF BSR         r,m         1         1         1         1         1	SHLD SHRD	m,r,i/CL	2		1	1	1	1			
BTR BTS BTC         r,r/i         1         2	BT	r,r/i			1						
BTR BTS BTC         m,r/i         1         6         1         1         1           BSF BSR         r,r         1         1         1         1           BSF BSR         r,m         1         1         1         1	BT	m,r/i	1		6	1					
BSF BSR         r,r         1         1         1           BSF BSR         r,m         1         1         1	BTR BTS BTC	r,r/i			1						
BSF BSR	BTR BTS BTC	m,r/i	1		6	1	1	1			
BSF BSR r,m 1 1 1 1	BSF BSR	r,r		1	1						
	BSF BSR			1	1	1					
	SETcc				1						

SETcc	m	1	1	1		1	1	1	
JMP	short/near		1	'		'	<b>'</b>		2
JMP	far	21	'		1				
JMP	r	21	1		'				2
JMP	m(near)				1				2
JMP		21	'		2				
	m(far) short/near	21	1						2
conditional jump			1	4		4	_		2
CALL	near	20	'	1	4	1	1		
CALL	far	28	,		1	2	2		_
CALL	r		1	2	,	1	1		2 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	'n					
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							
IN		18						>300	
OUT		18						>300	
PREFETCHNTA d)	m	10			1			5000	
PREFETCHT0/1/2 d)	m m				1				
SFENCE d)	""				'	1	1		6
SI ENCE U)									U

Notes

- a) Faster under certain conditions: see manual 3: "The microarchitecture of Intel, AMD and VIA CPUs".
- b) Has an implicit LOCK prefix.
- c) 3 if constant without base or index register
- d) P3 only.

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

Instruction	Operands				ops			Latency	Reciprocal	
		p0	<b>p1</b>	p01		р3	p4		throughput	
FLD	r	1	ļ .	ľ		ľ				
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-

tency-1).

i) Faster for lower precision.

#### Integer MMX instructions (Pentium II and Pentium III)

Instruction	Operands	Operands µops						Latency	Reciprocal
		p0	p1	p01	<b>p2</b>	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.

k) 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	<b>p2</b>	р3	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	µops fused	μ	ops	unfus	sed d	in	Latency	cal	
		domain	p0	p1	p01	p2	р3	р4		through- put
Move instructions										
MOV	r,r/i	1			1					0.5
MOV	r,m	1				1				1
MOV	m,r	1					1	1		1
MOV	m,i	2					1	1		1
MOV	r,sr	1			1					
MOV	m,sr	2			1		1	1		
MOV	sr,r	8	8						5	
MOV	sr,m	8	7			1			8	
MOVNTI	m,r32	2					1	1		2
MOVSX MOVZX	r,r	1			1				1	0.5
MOVSX MOVZX	r,m	1				1				1
CMOVcc	r,r	2	1		1				2	1.5
CMOVcc	r,m	2	1		1	1				
XCHG	r,r	3			3				2	1.5

XCHG	r,m	7			4	1	1	1	high b)		1
XLAT	.,	2			1	1			g 2)	1	ĺ
PUSH	_	1			'	'	1	1	4	1	
	r						1	-	1		
PUSH	i	2					1	1	1	1	ĺ
PUSH	m	2				1	1	1	2	1	
PUSH	sr	2			1		1	1			
PUSHF(D)		16	3		11		1	1		6	
PUSHA(D)		18			2		8	8	8	8	
POP	r	1			-	1					
POP	(E)SP	3			2	1				_	
POP	m	2				1	1	1	2	1	
POP	sr	10			9	1					
POPF(D)		17	10		6	1				16	
POPA(D)		10			2	8			7	7	
LAHF SAHF		1			1				1	1	
SALC		2	1	1	'					1	
LEA			1	'					4		
	r,m	1	1						1	1	
BSWAP	r	2	1		1						
LDS LES LFS LGS LSS	m	11			8	3					
PREFETCHNTA	m	1				1				1	
PREFETCHT0/1/2	m	1				1				1	
SFENCE/LFENCE/MFENCE	<u> </u>	2					1	1		6	
IN		_	18				-		>300		
OUT			18						>300		
001			10						7300		
Arithmetic instructions											
ADD SUB	r,r/i	1			1				1	0.5	
ADD SUB	r,m	1			1	1			2	1	
ADD SUB	m,r/i	3			1	1	1	1		1	
ADC SBB	r,r/i	2		1	1				2	2	
ADC SBB	r,m	2		1	1	1					
ADC SBB	m,r/i	7			4	1	1	1			
CMP	r,r/i	1			1	'	'		1	0.5	
CMP		1			1 -	4			1		
	m,r				1	1			'	1	
CMP	m,i	2			1	1				1	
INC DEC NEG NOT	r	1			1				1	0.5	
INC DEC NEG NOT	m	3			1	1	1	1			
AAA AAS DAA DAS		1		1							
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,i	2	1			1			4	1	
DIV IDIV	r8	5	4		1				15-16 c)	12	ĺ
DIV IDIV	r16	4	3		1				15-10 c)	12-20 c)	
									1		
DIV IDIV	r32	4	3		1	_			15-39 c)	12-20 c)	ĺ
DIV IDIV	m8	6	4		1	1			15-16 c)	12	ĺ
DIV IDIV	m16	5	3		1	1			15-24 c)	12-20 c)	

DIV IDIV	m32	5	3		1	1			15-39 c)	12-20 c)
CBW CWDE CWD CDQ		1		1					1 1	1 1
CWD CDQ		1							'	I
Logic instructions										
AND OR XOR	r,r/i	1			1				1	0.5
AND OR XOR	r,m	1			1	1			2	1 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	9	9
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	1	2	2		2	'	'	'	2	2
SHLD SHRD	r,r,i/CL		1		1	4	1	1		
	m,r,i/CL	4	'	1	1	1	'	'	4	1
BT BT	r,r/i	1		1	7	4			1	1
BT	m,r	8		1	/	1				
	m,i	2 1		1		1				
BTR BTS BTC	r,r/i				7	4	4	4		
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 CMC	m	2 1		1			1	1		1
CLC STC CMC				1	4					1 1
CLD STD		4			4					7
Control transfer instruction										
Control transfer instruction	ns short/near	1		1						1
JMP	far	22	21	1		1				28
JMP		1	21	1		'				1
JMP	r m(near)	2		1		1				2
JMP	m(near)	25	23	'		2				31
	m(far) short/near	1	23	4		2				
conditional jump	short	2		1 1	4					1 1
J(E)CXZ LOOP		11	2	-	8					6
	short			1						
LOOP(N)E	short	11	2	1	8		4	4		6
CALL CALL	near	4	27	1	1	1	1	1 2		2 27
	far	32	21	4	2	1	2			
CALL	r	4		1	2	,	1	1		9 2
CALL	m(near)	4	20	1		1	1	1		
CALL	m(far)	35	29	4	2	2	2	2		30
RETN		2		1	2	1				2

RETN RETF RETF BOUND INTO	i i r,m	3 27 27 15 5	24 24 7	1	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			10+01	1	1	1		1
REP STOS		5n			¢a. 5r	-	ļ .	'		0.7
MOVS		6			1	3	1	1		0.7
REP MOVS		6n		(	ca. 6r	a)				0.5
SCAS		3			1	2				1.3
REP(N)E SCAS		7n			12+7r					0.6
CMPS		6			4	2				0.7
REP(N)E CMPS		9n			12+9r	1				0.5
Other										
NOP (90)		1			1					0.5
Long NOP (0F 1F)		1			1					1
PAUSE		2			2					
CLI			9							
STI			17							
ENTER	i,0	12			10		1	1		
ENTER	a,b		ca.	18	+4b		b-1	2b		
LEAVE		3	20 5		2	1			400	
CPUID		38-59	38-59	)					ca. 130	40
RDTSC		13	13							42

### Notes:

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

Has an implicit LOCK prefix.

b) 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 earlyc)

out algorithm.

## Floating point x87 instructions

Instruction	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 FLD1 FLDPI FLDL2E etc. FCMOVcc FNSTSW FNSTSW FLDCW FNSTCW FINCSTP FDECSTP FFREE FFREEP FNSAVE FRSTOR	r AX m16 m16 m16	1 2 2 3 2 3 1 1 2 142 72	1 2 2 3 1 1 1 1 1 1 2 142 72		1	1	1	1	2 7	3 19 3 1 2 131 91
Arithmetic instructions  FADD(P) FSUB(R)(P)  FADD(P) FSUB(R)(P)  FMUL(P)  FMUL(P)  FDIV(R)(P)  FDIV(R)(P)  FABS  FCHS  FCOM(P) FUCOM  FCOM(P) FUCOM  FCOMPP FUCOMPP  FCOMI(P) FUCOMI(P)  FIADD FISUB(R)  FIMUL  FIDIV(R)  FICOM(P)  FTST  FXAM  FPREM FPREM1  FRNDINT	r m r m r m r m m	1 1 1 1 1 1 1 1 1 2 1 6 6 6 1 1 26 15	1 1 1 1 1 1 1 3 5 5 3 26 15	1 1 1 1 1 1 1	1 1 1	1 1 1 1 1 1			3 3 5 5 9-38 c) 9-38 c) 1 1 1 1 1 3 5 9-38 c)	1 1 2 2 8-37 c) 8-37 c) 1 1 1 1 1 3 3 8-37 c) 4 1
Math FSCALE FXTRACT FSQRT FSIN FCOS FSINCOS F2XM1 FYL2X FYL2XP1 FPTAN FPATAN		28 15 1 80-100 90-110 ~ 20 ~ 40 ~ 55 ~ 100 ~ 85	28 1 80-10 90-17 ~20 ~40 ~55 ~100 ~85	10	15			80-1 100- ~45 ~60 ~65 ~140	43 9 9 h) 10 130	8
Other FNOP WAIT		1 2	1	1	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	µops fused	μ	ops	unfus	sed d	omai	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		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 PSHUFL/HW PSHUFL/HW MASKMOVQ MASKMOVDQU PMOVMSKB PMOVMSKB PEXTRW PINSRW	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 r32,xmm,i mm,r32,i	1 1 2 3 4 2 3 3 8 1 1 2 4	1 1 1 1 1 1 2 1	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 1 2 2 1 1 1 1 2
PINSRW	xmm,r32,i	2	2						1	2
Arithmetic instructions PADD/SUB(U)(S)B/W/D PADD/SUB(U)(S)B/W/D PADD/SUB(U)(S)B/W/D PADD/SUB(U)(S)B/W/D	mm,mm mm,m64 xmm,xmm xmm,m128	1 1 2 4			1 1 2 2	1 2			1	0.5 1 1 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			4	1 1
PCMPEQ/GTB/W/D	xmm,xmm	2			2				1	1 1
PCMPEQ/GTB/W/D	xmm,m128	2			2	2			2	2
PMULL/HW PMULHUW	mm,mm	1 1			1	4			3	1 1
PMULL/HW PMULHUW PMULL/HW PMULHUW	mm,m64	1 2			1 2	1			3 3	1 2
PMULL/HW PMULHUW	xmm,xmm xmm,m128	4			2	2			3	2 2
PMULUDQ	mm,mm	1	1		_	-			3 4	1
PMULUDQ	mm,m64	1				1			4	1 1
PMULUDQ	xmm,xmm	2	2			'			4	2
PMULUDQ	xmm,m128	4	2			2			4	2
PMADDWD	mm,mm	1			1				3	1 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			4	1
PMIN/MAXUB/SW	xmm,xmm	2			2				1	1 1
PMIN/MAXUB/SW	xmm,m128	4			2	2			4	2
PSADBW BSADBW	mm,mm	2			2	4			4	1 1
PSADBW PSADBW	mm,m64 xmm,xmm	2 4			2 4	1			4 4	1 2
	, ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	7	I		, 7	I	I		7	

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

### Notes:

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

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	μ	ops	unfus	ed d	oma	in	Latency	Recipro-
		fused domain	p0	p1	p01	p2	р3	p4		cal 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	· ·	4	3	1		'		4	3
	xmm,xmm							4	
CVTPD2PS	xmm,m128	6	3	1		2			3
CVTSD2SS	xmm,xmm	2			2			4	2
CVTSD2SS	xmm,m64	3			2	1			2
CVTSS2SD	xmm,xmm	2	2					2	2
CVTSS2SD	xmm,m64	3	2			1			2
CVTDQ2PS	xmm,xmm	2			2			3	2
CVTDQ2PS	xmm,m128	4			2	2			2
CVT(T) PS2DQ	xmm,xmm	2			2			3	2
CVT(T) PS2DQ	xmm,m128	4			2	2			2
CVTDQ2PD	xmm,xmm	4			4	-		4	2
CVTDQ2PD	xmm,m64	5			4	1			2
CVTDQ2FD CVT(T)PD2DQ	· ·	4			4	'		4	3
` '	xmm,xmm							4	
CVT(T)PD2DQ	xmm,m128	6			4	2			3
CVTPI2PS	xmm,mm	1		1		.		3	1
CVTPI2PS	xmm,m64	2		1		1			1
CVT(T)PS2PI	mm,xmm	1		1				3	1
CVT(T)PS2PI	mm,m128	2		1		1			1
CVTPI2PD	xmm,mm	4	2	2				5	2
CVTPI2PD	xmm,m64	5	2	2		1			2
CVT(T) PD2PI	mm,xmm	3			3			4	2
CVT(T) PD2PI	mm,m128	5			3	2			2
CVTSI2SS	xmm,r32	2	1	1		-		4	1
CVT(T)SS2SI	r32,xmm	2	'	1	1			4	1
CVT(T)SS2SI	r32,m32	3		1		1		7	1
, ,			,		'	'		4	
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
CVT(T)SD2SI	r32,m64	3		1	1	1			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		1	1		3			4	1
	xmm,xmm								
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
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 1		9-32 c)	8-31 c)
2.00	, A.IIII,IIIOT	_	1 '	1	I	·	1	0 02 0)	0010)

DIVIDO		0	۱ ۵	l	I	I	I	l	10.24 a)	16 24 6\
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 1
CMPccSS/D	xmm,m32/64	2			1	1			_	1
CMPccPS/D	xmm,xmm	2			2				3	2
CMPccPS/D	xmm,m128	4			2	2				2
COMISS/D UCOMISS/D	xmm,xmm	1		1						1
COMISS/D UCOMISS/D	xmm,m32/64	2		1		1				1
MAXSS/D MINSS/D	xmm,xmm	1			1				3	1
MAXSS/D MINSS/D	xmm,m32/64	2			1	1			3	1
MAXPS/D MINPS/D	xmm,xmm	2			2				3	2
MAXPS/D MINPS/D	xmm,m128	4			2	2			3	2
RCPSS	xmm,xmm	1		1					3	1
RCPSS	xmm,m32	2		1		1				1
RCPPS	xmm,xmm	2		2					3	2
RCPPS	xmm,m128	4		2		2				2
Math										
SQRTSS	xmm,xmm	2	2						6-30	4-28
SQRTSS	xmm,m32	3	2			1				4-28
SQRTSD	xmm,xmm	1	1						5-58	4-57
SQRTSD	xmm,m64	2	1			1				4-57
SQRTPS	xmm,xmm	2	2						8-56	16-55
SQRTPD	xmm,xmm	2	2						16-114	16-114
SQRTPS	xmm,m128	4	2			2				16-55
SQRTPD	xmm,m128	4	2			2				16-114
RSQRTSS	xmm,xmm	1		1					3	1
RSQRTSS	xmm,m32	2		1		1				1
RSQRTPS	xmm,xmm	2		3					3	2
RSQRTPS	xmm,m128	4		2		2				2
Logic										
AND/ANDN/OR/XORPS/D	xmm,xmm	2			2				1	1
AND/ANDN/OR/XORPS/D	xmm,m128	4			2	2				1
Other										
LDMXCSR	m32	9	9							20
STMXCSR	m32	6	6							12
FXSAVE	m4096	118	32				43	43		63
FXRSTOR	m4096	87	43			44				72

### Notes:

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

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

j) Also uses some execution units under port 1.

# Intel Core 2 (Merom, 65nm)

## List of instruction timings and µop breakdown

Explanation of column headings:

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

register, (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.

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  $\mu$ op fusion if the sum of the numbers listed under p015 + p2 + p3 + p4 exceeds the number listed under  $\mu$ ops fused domain. An x under p0, p1 or p5 means that at least one of the  $\mu$ 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  $\mu$ 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  $\mu$ 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

<u>latency figure.</u>

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

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

given by the time stamp counter.

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

pendent instructions of the same kind in the same thread.

#### Integer instructions

Instruction	Operands	μορs fused	μops	un	fuse	d d	oma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	р5	<b>p2</b>	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	Х	х	4	-	-	int		16
MOV	sr,m	8	3	X	^	X	5			int		16
MOVNTI		2	"	^		^	J	1	1	int		2
	m,r							'	'	1110		
MOVSX MOVZX MOVSXD		4	4		١.,	.,				:4	4	0.00
	r,r	1	1	Х	X	Х				int	1	0.33
MOVSX MOVZX	r,m	1	_				1			int		1
CMOVcc	r,r	2	2	X	X	X				int	2	1
CMOVcc	r,m	2	2	X	X	X	1			int		
XCHG	r,r	3	3	X	X	Х				int	2	2
XCHG	r,m	7	Х				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)	_	17	15	×	x	Х		1	1	int		7
PUSHA(D) i)		18	9	^				1	8	int		8
POP	r	1					1	•		int	2	1
POP	(E/R)SP	4	3				1			int	_	•
POP	m (L/N)Si	2	"				1	1	1	int		1.5
POP	sr	10	9				1	'	'	int		1.5
POPF(D/Q)	51	24	23	x	x	x	1			int	20	17
		10	2	^	^	^	8			int	20	7
POPA(D) i)					١.,	.,	0				4	
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	r	2	2	1		1	_			int	4	1
LDS LES LFS LGS LSS	m	11	11				1			int		17
PREFETCHNTA	m	1					1			int		1
PREFETCHT0/1/2	m	1					1			int		1
LFENCE		2						1	1	int		8
MFENCE		2						1	1	int		9
SFENCE		2						1	1	int		9
CLFLUSH	m8	4	2	X	Х	Х		1	1	int	240	117
IN										int		
OUT										int		
Arithmetic instructions												
ADD SUB	r,r/i	1	1	X	X	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	Х	Х	Х				int	2	2
ADC SBB	r,m	2	2	х	Х	Х	1			int	2	2
ADC SBB	m,r/i	4	3	Х	Х	Х	1	1	1	int	7	
CMP	r,r/i	1	1	Х	Х	Х				int	1	0.33
CMP	m,r/i	1	1	Х	Х	Х	1			int	1	1
INC DEC NEG NOT	r	1	1	Х	Х	Х				int	1	0.33
INC DEC NEG NOT	m	3	1	x	x	х	1	1	1	int	6	1

AAA AAS DAA DAS i)													
AAM i)  AAM i)  AAM i)  AAM i)  AAM i)  MUL IMUL  r8 1 1 1 1 1 1 int 17  MUL IMUL  r16 3 3 3 x x x x int 5 1.5  MUL IMUL  r16 3 3 3 x x x x x int 5 1.5  MUL IMUL  r16, r16 1 1 1 1 1 int 3 1  IMUL  r16, r16 1 1 1 1 1 int 3 1  IMUL  r16, r16 1 1 1 1 1 int 3 1  IMUL  r16, r16 1 1 1 1 1 int 3 1  IMUL  r16, r16 1 1 1 1 1 int 3 1  IMUL  r16, r16 1 1 1 1 1 int 3 1  IMUL  r16, r16 1 1 1 1 1 int 3 1  IMUL  r18, r16  MUL IMUL  r18, r16  R1 1 1 1 1 Int 1 int 3 1  IMUL  r18, r16  R1 1 1 1 1 Int Int 1 Int Int 1 Int Int 1 Int	AAA AAS DAA DAS i)		1	1		1					int		1
MULI MUL	AAD i)		3	3	х	х	х				int		1
MULIMUL    16	AAM i)		4	4							int	17	
MULI MUL	MUL IMUL	r8	1	1		1					int	3	1
MULIMUL   r16,r162   1	MUL IMUL	r16	3	3	х	х	х				int	5	1.5
MUL	MUL IMUL	r32	3	3	х	х	х				int	5	1.5
MUL   r32,r32	MUL IMUL	r64	3	3	x	x	х				int		4
IMUL	IMUL	r16,r16	1	1		1					int	3	1 1
IMUL	IMUL	r32.r32	1	1		1					int	3	1 1
IMUL	l l	1			1								
MUL	l l	1 '				1							
MUL		1 ' ' 1				1							
MUL IMUL         m8         1		1 ' '			1	-							
MUL IMUL         m16         3         3         x         x         x         1         int         5         1.5           MUL IMUL         m64         3         2         2         1         int         5         1.5           MUL IMUL         m64         3         2         2         1         int         7         4           IMUL         r16,m16         1         1         1         1         1         int         3         1           IMUL         r64,m64         1         1         1         1         1         int         5         2           IMUL         r16,m16,i         1         1         1         1         1         int         5         2           IMUL         r64,m64,i         1         1         1         1         int         int         1         1           IMUL         r64,m64,i         1         1         1         1         int         int         1         1         1         int         1         1         1         1         int         18         12         2         DIV IDIV         r64         36         56         56 </td <td></td> <td>1 ' '  </td> <td></td> <td></td> <td>  -</td> <td>1</td> <td></td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td>		1 ' '			-	1		1					
MUL IMUL         m32         3         3         x         x         1         int         5         1.5           MUL IMUL         m64         3         2         2         1         1         int         7         4           IMUL         r16,m16         1         1         1         1         1         int         3         1           IMUL         r64,m64         1         1         1         1         1         int         5         2           IMUL         r16,m16,i         1         1         1         1         1         int         5         2           IMUL         r32,m32,i         1         1         1         1         1         int         5         2           IMUL         r64,m64,i         1         1         1         1         1         int         1         1           IMUL         r64,m64,i         1         1         1         1         1         int         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1	l l				x	-	x	-					
MUL IMUL         m64         3         2         2         1	l l												
MUL						^	^	-					
MUL	l l				_	1							
MUL	l l	1 '				· ·							
MUL		1			1	'							
MUL		1 ' 1			ļ '	1							I I
MUL	l l	1 1				· ·		-					
DIV	l l	1 1			1	'		-					
DIV		1 ' ' 1			ļ '			'				10	
DIV   DIV   r32	l l												
DIV   r64   32   32	l l	1											
IDIV   DIV													
DIV   DIV   DIV   m8		1											
DIV   DIV   DIV   m16   6   5								4					
DIV   DIV   m64   32   31													
DIV   m64   56   55	l l	1											· / I
IDIV													
CBW CWDE CDQE	l l	1											· /
CWD CDQ CQO         1         1         x         x         int         1           Logic instructions         7,r/i         1         1         x         x         int         1           AND OR XOR         r,m         1         1         x         x         int         1         0.33           AND OR XOR         r,m         1         1         x         x         1         int         1         1           AND OR XOR         m,r/i         2         1         x         x         1         int         1         1         1         1         1         int         1 <td>l l</td> <td>m64</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>ı</td> <td></td> <td></td> <td></td> <td></td> <td>28-40 C)</td>	l l	m64						ı					28-40 C)
Logic instructions	·					X							
AND OR XOR         r,r/i         1         1         x         x         x         int         1         0.33           AND OR XOR         r,m         1         1         x         x         x         1         int         1           AND OR XOR         r,m         1         1         x         x         x         1         int         1           AND OR XOR         r,m         1         1         x         x         x         1         int         1           AND OR XOR         m,r/i         1         1         x         x         1         int         1         1           AND OR XOR         m,r/i         1         1         x         x         1         1         int         6         1           TEST         m,r/i         1         1         x         x         1         int         1         0.33           TEST         m,r/i         1         1         x         x         1         int         1         0.55           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         1         1	CWD CDQ CQO		1	1	X		X				int	1	
AND OR XOR         r,r/i         1         1         x         x         x         int         1         0.33           AND OR XOR         r,m         1         1         x         x         x         1         int         1           AND OR XOR         r,m         1         1         x         x         x         1         int         1           AND OR XOR         r,m         1         1         x         x         x         1         int         1           AND OR XOR         m,r/i         1         1         x         x         1         int         1         1           AND OR XOR         m,r/i         1         1         x         x         1         1         int         6         1           TEST         m,r/i         1         1         x         x         1         int         1         0.33           TEST         m,r/i         1         1         x         x         1         int         1         0.55           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         1         1													
AND OR XOR  AND OR XOR  M,r/i  TEST  TEST  M,r/i  SHR SHL SAR  ROR ROL  ROR			1	1	.,	,,					int	4	0.22
AND OR XOR  TEST  TEST  TRIFT  T		1						4				'	
TEST         r,r/i         1         1         x         x         x         x         int         1         0.33           TEST         m,r/i         1         1         x         x         int         1         0.33           SHR SHL SAR         r,i/cl         1         1         x         x         int         1         0.5           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         1         0.5           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         1         0.5           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         1         0.5           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         1         1         0.5           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         1         1         1         1         1         1         1         1         1         1								-	_				·
TEST         m,r/i         1         1         x         x         x         1         int         1           SHR SHL SAR         r,i/cl         1         1         x         x         1         1         int         1         0.5           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         6         1           ROR ROL         r,i/cl         1         1         x         x         int         1         1           ROR ROL         m,i/cl         3         2         x         x         1         1         int         6         1           ROR ROL         m,i/cl         3         2         x         x         1         1         int         1         1           RCR RCL         r,1         2         2         x         x         x         int         12         2           RCR         r8,i/cl         9         9         x         x         x         int         11         x         x         x         int         11         x         x         x         x         x         x         x	l l	1						ı	1	1			· .
SHR SHL SAR         r,i/cl         1         1         x         x         1         1         x         x         1         1         0.5           SHR SHL SAR         m,i/cl         3         2         x         x         1         1         int         6         1           ROR ROL         r,i/cl         1         1         x         x         int         1         1           ROR ROL         m,i/cl         3         2         x         x         1         1         int         6         1           RCR RCL         r,1         2         2         x         x         x         int         1         2         2           RCR         r8,i/cl         9         9         x         x         x         int         12         2           RCL         r8,i/cl         8         8         x         x         x         int         11         x         x         x         int         11         x         x         x         x         x         x         x         x         x         x         x         x         x         x         x         x         x	l l	1						4				1	
SHR SHL SAR       m,i/cl       3       2       x       1       1       1       int       6       1         ROR ROL       r,i/cl       1       1       x       x       1       1       int       1       1         ROR ROL       m,i/cl       3       2       x       x       1       1       int       6       1         RCR RCL       r,1       2       2       x       x       x       int       2       2         RCR       r8,i/cl       9       9       x       x       x       int       12         RCL       r8,i/cl       8       8       x       x       x       int       11         RCR RCL       r16/32/64,i/cl       6       6       x       x       x       int       11         RCR RCL       m,1       4       3       x       x       1       1       int       11         RCR       m8,i/cl       12       9       x       x       1       1       int       14		1				X		ı				_	· .
ROR ROL       r,i/cl       1       1       x       x       int       1       1         ROR ROL       m,i/cl       3       2       x       x       1       1       int       6       1         RCR RCL       r,1       2       2       x       x       x       int       2       2         RCR       r8,i/cl       9       9       x       x       int       12         RCL       r8,i/cl       8       8       x       x       int       11         RCR RCL       r16/32/64,i/cl       6       6       x       x       x       int       11         RCR RCL       m,1       4       3       x       x       1       1       int       7         RCR       m8,i/cl       12       9       x       x       x       1       1       int       14		·											
ROR ROL       m,i/cl       3       2       x       x       1       1       int       6       1         RCR RCL       r,1       2       2       x       x       x       int       2       2         RCR       r8,i/cl       9       9       x       x       x       int       12         RCL       r8,i/cl       8       8       x       x       x       int       11         RCR RCL       r16/32/64,i/cl       6       6       x       x       x       int       11         RCR RCL       m,1       4       3       x       x       x       1       1       int       7         RCR       m8,i/cl       12       9       x       x       x       1       1       int       14	l l	1						1	1	1			
RCR RCL       r,1       2       2       x       x       x       int       2       2         RCR       r8,i/cl       9       9       x       x       x       int       12         RCL       r8,i/cl       8       8       x       x       x       int       11         RCR RCL       r16/32/64,i/cl       6       6       x       x       x       int       11         RCR RCL       m,1       4       3       x       x       x       1       1       int       7         RCR       m8,i/cl       12       9       x       x       x       1       1       int       14	l l	1											
RCR       r8,i/cl       9       9       x       x       x       int       12         RCL       r8,i/cl       8       8       x       x       x       int       11         RCR RCL       r16/32/64,i/cl       6       6       x       x       x       int       11         RCR RCL       m,1       4       3       x       x       x       1       1       int       7         RCR       m8,i/cl       12       9       x       x       x       1       1       int       14	l l	1						1	1	1			
RCL       r8,i/cl       8       8       x       x       x       int       11         RCR RCL       r16/32/64,i/cl       6       6       x       x       x       int       11         RCR RCL       m,1       4       3       x       x       x       1       1       int       7         RCR       m8,i/cl       12       9       x       x       x       1       1       int       14	l l	1											2
RCR RCL     r16/32/64,i/cl     6     6     x     x     x     x     int     11       RCR RCL     m,1     4     3     x     x     x     1     1     int     7       RCR     m8,i/cl     12     9     x     x     x     1     1     int     14	l l	1 '											
RCR RCL m,1 4 3 x x x 1 1 1 int 7 RCR m8,i/cl 12 9 x x x 1 1 1 int 14		1 '											
RCR m8,i/cl 12 9 x x x 1 1 1 int 14	l l					X							
	l l				X	X	Х	-	1	'			
	l l	1 1							-				
RCL	RCL	m8,i/cl	11	8	X	X	X	1	1	1	int	13	

RCR RCL SHLD SHRD SHLD SHRD BT BT BT BT BT BTR BTS BTC BTR BTS BTC BTR BTS BTC BTR BTS BTC CC BSF BSR SETCC SETCC CLC STC CMC CLD STD	m16/32/64,i/cl r,r,i/cl m,r,i/cl r,r/i m,r m,i r,r/i m,r r,r m,i r,r m,i r,r 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	13 2 7 1 1 5 6 2	1 5 1 2 1 0.33 4 14
Control transfer instruction JMP JMP i) JMP JMP JMP Conditional jump Fused compare/test and b J(E/R)CXZ LOOP LOOP(N)E CALL CALL CALL CALL CALL RETN RETN RETF RETF BOUND i) INTO i)	short/near far r m(near) m(far) short/near	1 30 1 1 31 1 2 11 3 43 3 4 44 1 3 32 32 15 5	1 30 1 1 29 1 1 2 43 2 3 42 1 x 30 30 13 5	x x x x	x x x x	1 1 1 1 1 1 X X X X 1 1 1	1 2 1 1 2 2 2 2	1 1 1	1 1 1	int	0 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 REP(N)E CMPS		3 4+7n - 7 4 8+5n - 2 8 1 7+7n - 7 4 7+8n - 7 7+10n -	20+1 5 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	0.55n 0.63n 1 123+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	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												
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	4	4	١.,	١.,	١.,	I	İ	I	int	l 4	0.22
MOVQ	(x)mm, (x)mm	1	1	Х	Х	Х	1			int	1 2	0.33
MOVQ	(x)mm,m64	1					1	1	1	int	3	1
· ·	m64, (x)mm	1	4		١.,	.,		I	'	:4		1
MOVDQA	xmm, xmm	1	1	Х	Х	Х	4			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	Х	1	2	2		3-8	4
MOVDQU	xmm, m128	4	2	X		Х	2			int	2-8	2
LDDQU g)	xmm, m128	4	2	X		Х	2			int	2-8	2
MOVDQ2Q	mm, xmm	1	1	Х	X	Х				int	1	0.33
MOVQ2DQ	xmm,mm	1	1	Х	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
PSHUFB h)	mm,mm	1	1			1				int	1	1
PSHUFB h)	mm,m64	2	1			1	1			int		1
PSHUFB h)	xmm,xmm	4	4							int	3	2
PSHUFB h)	xmm,m128	5	4				1			int		2
PSHUFW	mm,mm,i	1	1			1				int	1	1
PSHUFW	mm,m64,i	2	1			1	1			int		1
PSHUFD	xmm,xmm,i	2	2	X	Х	1				flt→int	3	1
PSHUFD	xmm,m128,i	3	2	X	Х	1	1			int		1
PSHUFL/HW	xmm,xmm,i	1	1			1				int	1	1
PSHUFL/HW	xmm, m128,i	2	1			1	1			int		1
PALIGNR h)	mm,mm,i	2	2	X	Х	Х				int	2	1
PALIGNR h)	mm,m64,i	2	2	X	Х	Х	1			int		1
PALIGNR h)	xmm,xmm,i	2	2	X	Х	Х				int	2	1
PALIGNR h)	xmm,m128,i	2	2	x	х	Х	1			int		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
PEXTRW	r32,xmm,i	3	3							int	5	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				int	6	1.5
PINSRW	xmm,m16,i	4	3	Х	х	х	1			int		1.5
Arithmetic instructions												
PADD/SUB(U)(S)B/W/D	(x)mm, (x)mm	1	1	Х		х				int	1	0.5
PADD/SUB(U)(S)B/W/D	(x)mm,m	1	1	Х		х	1			int		1
PADDQ PSUBQ	(x)mm, (x)mm	2	2	Х		Х				int	2	1
PADDQ PSUBQ	(x)mm,m	2	2	X		Х	1			int		1

DUADD(C)M	1 1		1	ı	I		1		1	I		l I
PHADD(S)W PHSUB(S)W h)		_	_							in.	5	_
, , ,	mm,mm	5	5							int	5	4
PHADD(S)W		0	_									_
PHSUB(S)W h)	mm,m64	6	5				1			int		4
PHADD(S)W		_	_									
PHSUB(S)W h)	xmm,xmm	7	7							int	6	4
PHADD(S)W			_									
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		2
PHADDD PHSUBD h)	xmm,xmm	5	5							int	5	3
PHADDD PHSUBD h)	xmm,m128	6	5				1			int		3
PCMPEQ/GTB/W/D	(x)mm,(x)mm	1	1	X		х				int	1	0.5
PCMPEQ/GTB/W/D	(x)mm,m	1	1	х		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 1
PMULHRSW h)	(x)mm,(x)mm	1	1		1					int	3	1 1
PMULHRSW h)	(x)mm,m	1	1		1		1			int		1 1
PMULUDQ	(x)mm,(x)mm	1	1		1					int	3	1 1
PMULUDQ	(x)mm,m	1	1		1		1			int		1 1
PMADDWD	(x)mm,(x)mm	1	1		1		•			int	3	1 1
PMADDWD	(x)mm,m	1	1		1		1			int	U	1
PMADDUBSW h)	(x)mm,(x)mm	1			1		'			int	3	1
PMADDUBSW h)	(x)mm,m	1			1		1			int	3	1
PAVGB/W	1 ' '	1		\ ,	<b>'</b>	\ \	1			int	1	0.5
PAVGB/W	(x)mm,(x)mm	1	1	X		X	1				ı	
	(x)mm,m	•		X		X	ı			int	4	1
PMIN/MAXUB/SW	(x)mm,(x)mm	1	1	Х		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 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
Logic instructions												
PAND(N) POR PXOR	(x)mm,(x)mm	1	1	Х	Х	Х				int	1	0.33
PAND(N) POR PXOR	(x)mm,m	1	1	Х	х	Х	1			int		1
PSLL/RL/RAW/D/Q	mm,mm/i	1	1	1						int	1	1
PSLL/RL/RAW/D/Q	mm,m64	1	1	1			1			int		1
PSLL/RL/RAW/D/Q	xmm,i	1	1	1						int	1	1
PSLL/RL/RAW/D/Q	xmm,xmm	2	2	х	х					int	2	1
PSLL/RL/RAW/D/Q	xmm,m128	3	2	x	х		1			int		1 1
PSLL/RLDQ	xmm,i	2	2	x	х					int	2	1 1
		_									_	'
Other												
EMMS	1	11	11	x	х	х				float		6
		- •						L L				

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

Instruction	Operands	µops fused	μops	un	fuse	ed d	oma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	<b>p5</b>	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
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			1			int	1	1
MOVDDUP g)	xmm,m64	2	1	1			1			int		1
MOVSH/LDUP g)	xmm,xmm	1	1			1	١.			int	1	1
MOVSH/LDUP g)	xmm,m128	2	1			1	1			int		1
UNPCKH/LPS	xmm,xmm	3	3		3					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	
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
CVT(1)33231 CVTSI2SD	1 '	2	2	1	_	'			float	4	3
	xmm,r32	2			1					4	3
CVTSI2SD	xmm,m32		1		1	1			float		
CVT(T)SD2SI	r32,xmm	1	1		1				float	3	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
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	<u>'</u>		1			float		1
MULSD	xmm,xmm	1	1	1		'			float	5	1
MULSD	xmm,m64	1	1	1		1			float		1
MULPS	1 '	1	1	1		'			float	4	1
	xmm,xmm									4	-
MULPS	xmm,m128	1	1	1		1			float	_	1
MULPD	xmm,xmm	1	1	1					float	5	1
MULPD	xmm,m128	1	1	1		1			float		1
DIVSS	xmm,xmm	1	1	1					float	6-18 d)	5-17 d)
DIVSS	xmm,m32	1	1	1		1			float		5-17 d)
DIVSD	xmm,xmm	1	1	1					float	6-32 d)	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	1 '	1	1		1	'			float	3	1
INIWVOO'N INIINOO'N	xmm,xmm	1	ı						แบลเ	ا ا	1

MAXSS/D MINSS/D	xmm,m32/64	1	1		1		1			float		1	
MAXPS/D MINPS/D	xmm,xmm	1	1		1					float	3	1	
MAXPS/D MINPS/D	xmm,m128	1	1		1		1			float		1	
Math													
SQRTSS/PS	xmm,xmm	1	1	1						float	6-29	6-29	
SQRTSS/PS	xmm,m	2	1	1			1			float		6-29	
SQRTSD/PD	xmm,xmm	1	1	1						float	6-58	6-58	
SQRTSD/PD	xmm,m	2	1	1			1			float		6-58	
RSQRTSS/PS	xmm,xmm	1	1		1					float	3	2	
RSQRTSS/PS	xmm,m	1	1		1		1			float		2	
Logic													
AND/ANDN/OR/XORPS/D	xmm,xmm	1	1	х	Х	Х				int	1	0.33	
AND/ANDN/OR/XORPS/D	xmm,m128	1	1	Х	Х	Х	1			int		1	
Other													
LDMXCSR	m32	14	13				1					42	
STMXCSR	m32	6	4					1	1			19	
FXSAVE	m4096	141									145	145	
FXRSTOR	m4096	119									164	164	

Notes:

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.

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  $\mu$ op fusion if the sum of the numbers listed under p015 + p2 + p3 + p4 exceeds the number listed under  $\mu$ ops fused domain. An x under p0, p1 or p5 means that at least one of the  $\mu$ 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  $\mu$ 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	μορs fused	μops	un	fuse	ed d	oma	ain		Unit	- 3	procal
		do- main	p015	p0	p1	р5	<b>p2</b>	р3	p4			through- put
Move instructions												

MOV	r,r/i	1	1	Х	Х	Х					1	0.33
MOV a)	r,m	1					1				2	1
MOV a)	m,r	1						1	1		3	1
MOV	m,i	1						1	1		3	1
MOV	r,sr	1					1					1
MOV	m,sr	2					1	1	1			1
MOV	sr,r	8	4	Х	х	х	4					16
MOV	sr,m	8	3	х		х	5					16
MOVNTI	m,r	2						1	1			2
MOVSX MOVZX	, ,											
MOVSXD	r,r	1	1	х	X	x					1	0.33
MOVSX MOVZX	r16/32,m	1					1					1
MOVSX MOVSXD	r64,m	2	1	Х	x	x	1					1
CMOVcc	r,r	2	2	X	X	X					2	1
CMOVcc	r,m	2	2	X	X	X	1				_	•
XCHG	r,r	3	3	X	X	x	'				2	2
XCHG	r,m	7	x	^	_ ^	^	1	1	1		high b)	_
XLAT	1,111	2	1				1		'		4	1
PUSH	r	1	'				'	1	1		3	1
PUSH	i	1						1	1		J 3	1
PUSH		2					1					1
PUSH	m	2	4				'	1	1			
	sr	2 17	1 15	.,	.,			1	1			1
PUSHF(D/Q)				Х	Х	X		1	1			7
PUSHA(D) i)		18	9					1	8			8
POP	r	1					1				2	1
POP	(E/R)SP	4	3				1		١,			4 =
POP	m	2	_				1	1	1			1.5
POP	sr	10	9				1					17
POPF(D/Q)		24	23	Х	Х	Х	1				20	
POPA(D) i)		10	2				8					7
LAHF SAHF		1	1	Х	Х	X					1	0.33
SALC i)		2	2	Х	Х	х					4	1
LEA a)	r,m	1	1	1							1	1
BSWAP	r	2	2	1		1					4	1
LDS LES LFS LGS LSS	m	11	11				1					17
PREFETCHNTA	m	1					1					1
PREFETCHT0/1/2	m	1					1					1
LFENCE		2						1	1			8
MFENCE		2						1	1			6
SFENCE		2						1	1			9
CLFLUSH	m8	4	2	1		1		1	1		120	90
IN												
OUT												
Arithmetic instructions												
ADD SUB	r,r/i	1	1	Х	х	х					1	0.33
ADD SUB	r,m	1	1	Х	х	х	1					1
ADD SUB	m,r/i	2	1	Х	Х	X	1	1	1		6	1
ADC SBB	r,r/i	2	2	Х	X	X					2	2
ADC SBB	r,m	2	2	X	X	X	1				2	2
ADC SBB	m,r/i	4	3	X	X	X	1	1	1		7	-
CMP	r,r/i	1	1	X	X	X					1	0.33
CMP	m,r/i	1	1	X	x	X	1				1	1
1	,	•		1 1	1 -	1		I	I	I		

INC DEC NEG NOT	r	1	1	x	x	x					1 1	0.33
INC DEC NEG NOT	m	3		x	x	x	1	1	1		6	1
AAA AAS DAA DAS i)	'''	1	1	^	1	^	'	'	'		0	1
AAD i)		3	3	v	X	\ ,						1
AAM i)		5	5	X		X					17	ı
,	r0	1	1	Х	1 X	Х					3	4
MUL IMUL	r8	3		.,								1
MUL IMUL	r16		3	X	X	X					5	1.5
MUL IMUL	r32	3	3	Х	X	Х					5	1.5
MUL IMUL	r64	3	3	Х	X	X					7	4
IMUL	r16,r16	1	1		1						3	1
IMUL	r32,r32	1	1		1						3	1
IMUL	r64,r64	1	1	1							5	2
IMUL	r16,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	X	1				5	1.5
MUL IMUL	m32	3	3	Х	Х	x	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	x	x					14-22 c)	
DIV IDIV	r32	7	7	2	3	2					14-23 c)	
DIV	r64	32-38	32-38	9	10	13					18-57 c)	
IDIV	r64	56-62	56-62	X	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	x	X	_ x	1				14-23 c)	
DIV	m64	32	31	X	X	X	1				34-88 c)	
IDIV	m64	56	55	X	X	X	1				39-72 c)	
CBW CWDE CDQE		1	1	X	X	X	•				1	
CWD CDQ CQO		1	1	x	^	X						
OWD ODG OGO		'	'	^		^					'	
Logic instructions												
AND OR XOR	r,r/i	1	1	x	X	х					1	0.33
AND OR XOR	r,m	1	1	X	X	X	1				.	1
AND OR XOR	m,r/i	2	1	X	X	X	1	1	1		6	1
TEST	r,r/i	1	1	X	X	X	•	•	•		1	0.33
TEST	m,r/i	1	1	x	X	X	1				'	1
SHR SHL SAR	r,i/cl	1	1	x	^	x	'				1 1	0.5
SHR SHL SAR	m,i/cl	3	2	x		x	1	1	1		6	1
ROR ROL	r,i/cl	1	1	x		х Х	'		'		1 1	1
	· ·	3	2				1	1	1		6	1
ROR ROL	m,i/cl	2		X	,,	X	'	ı				2
RCR RCL	r,1		2	X	X	X					2	2
RCR	r8,i/cl	9	9	X	X	X					12	
RCL	r8,i/cl	8	8	X	X	X					11	
RCR RCL	r,i/cl	6	6	X	X	X	_		,		11	
RCR RCL	m,1	4	3	X	X	X	1	1	1		7	

RCR RCL m8,i/cl m1,i m1,i m1,i m1,i m1,i m1,i m1,i m1,i													
RCR   Mild   10			1	1									
SHLD SHRD			1	1	x		×						
SHLD SHRD			1	l				١.	'	ľ			1 1
BT			1	l				1	1	1			'
BT								'	'	'			4
BTR BTS BTC			1					1				1	
BTR BTS BTC BTR BTS BTC BTR BTS BTC BTR BTS BTC BSF BSR C,r,r C, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,				l									
BTR BTS BTC								'				1	'
BTR BTS BTC			1					1	1	1			
BSF BSR			1							_			
BSF BSR			1					'	'	'			4
SETCC			1	l				1				2	
SETCC								'				1	
CLC STC CMC			1						1	1		1	-
CLD		111	1						'	'		4	-
STD			1									1	
Control transfer instructions			1	l									
JMP   JMP   Far   30   30   Jmp   Far   30   30   Jmp   Jmp   Far   30   30   Jmp   Jmp	SID		0	0	X	X	X						14
JMP   JMP   Far   30   30   Jmp   Far   30   30   Jmp   Jmp   Far   30   30   Jmp   Jmp	Control transfer instructi	ons .											
JMP i)			1	1			1					0	1-2
JMP         r         1         2         2         2         1         1         1         1         2         2         1         1         1         2         2         2         1         1         1         2         2         2         2         1         1         1         1         1         1         1			30	30									
JMP	1 '		1				1					0	
JMP		m(near)	1	1				1					
Conditional jump		, , ,	1					2					
Fused compare/test and branch e,i)			1				1	_				0	
J(E/R)CXZ		1					l						
Short   11   11   x   x   x   x			2	l	x	x	l						
LOOP(N)E         short         11         11         x         x         x         1         1         2         2         2         2         75         2         75         2         75         2         75         2         75         2         75         2         75         2         2         75         2         2         75         2         2         75         2         75         2         2         75         2         75         2         2         75         2         78         2         78         2         78         2         78         2         78         2         78         3         2         75         3         3         2         4         78         3													
CALL (CALL (C			1										
CALL i)         far         43         43         75           CALL         r         3         2         1         1         2           CALL         m(near)         4         3         1         1         1         1         2           CALL         m(far)         44         42         2         75         75           RETN         1         1         1         1         1         2         75           RETN         i         3         1         1         1         1         2         75           RETF         i         32         30         2         78			1	l					1	1			
CALL         r         3         2         1         1         1         2         2         2         2         2         2         2         2         2         75         2         2         75         2         75         2         75         2         75         2         75         2         75         2         75         2         75         2         2         75         2         2         75         2         2         75         2         2         75         2         2         75         2         2         75         2         2         78         3         2         78         3         2         78         3         2         78         3         3         2         78         3         3         2         78         3         3         3         2         78         3         3         3         2         78         3         3         3         2         1<										-			
CALL CALL CALL CALL (CALL CALL (CALL CALL	· · · · · · · · · · · · · · · · · · ·		1	l					1	1			
CALL         m(far)         44         42         2         75           RETN         1         1         1         1         1         2           RETN         i         3         1         1         1         2         78           RETF         i         32         30         2         78         78           BOUND i)         r,m         15         13         2         78         8           INTO i)         5         5         5         3         2         78         8           String instructions         3         2         1 <td>                                     </td> <td>m(near)</td> <td>1</td> <td>1</td> <td></td> <td></td> <td></td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td>		m(near)	1	1				1					
RETN         i         1         1         1         1         1         2           RETR         i         32         30         2         78           RETF         i         32         30         2         78           BOUND i)         r,m         15         13         2         8           INTO i)         5         5         3         2         1         8           String instructions           LODS         3         2         1													
RETN       i       3       1       1       1       2         RETF       i       32       30       2       78         BOUND i)       r,m       15       13       2       8         INTO i)       5       5       3       2       1       1         String instructions       3       2       1       1       1       1       1         LODS       3       2       1 </td <td></td> <td>( - ,</td> <td></td> <td></td> <td></td> <td></td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>		( - ,					1						
RETF RETF BOUND i) INTO i)  String instructions LODS REP LODS STOS REP STOS MOVS  REP MOVS REP MOVS REP MOVS SCAS REP(N)E SCAS  I A 32 30		i	3	1			1	1					
RETF	RETF		32	30				2					
BOUND i)   r,m		i	32	l				2					
String instructions		r,m	15	13				2					8
STOS	•		5	5									
STOS													
REP LODS       4+7n-14+6n       1+5n-21+3n         STOS       4   2         1   1       1         REP STOS       8+5n-20+1.2n       7+2n-0.55n         MOVS       8   5   REP MOVS       7+7n-13+n       1+3n-0.63n         SCAS       4   3     1                   1         REP(N)E SCAS       7+8n-17+7n       3+8n-23+6n													
STOS     4   2         1   1     1       REP STOS     8+5n-20+1.2n     7+2n-0.55n       MOVS     8   5			1	I				1					
REP STOS       8+5n-20+1.2n       7+2n-0.55n         MOVS       8   5			1									1+5n-21	1+3n
MOVS     8   5			1	l					1	1			
1   1   1   5			8+5n-2	0+1.2	'n							7+2n-0.	55n
REP MOVS     7+7n-13+n     1+3n-0.63n       SCAS     4   3       1         1   1   1         REP(N)E SCAS     7+8n-17+7n     3+8n-23+6n	MOVS												
SCAS     4   3       1   1   1   1   1   1   1   1			1 -		1		5						
REP(N)E SCAS 7+8n-17+7n 3+8n-23+6n			1									1+3n-0.	
			1	I				1					
CMPS	. ,		1		ı	ı	ı	ı	1	1		3+8n-23	
	CMPS		7	5				2					3

REP(N)E CMPS		7+10n-	7+9n	I	I	I	I	I	l	2+7n-22	2+5n
Other NOP (90) Long NOP (0F 1F)		1 1	1	x x	x x	X X					0.33 1
PAUSE		3	3	х	х	x					8
ENTER	i,0	12	10					1	1		8
ENTER	a,b										
LEAVE		3	2				1				
CPUID		53-117									53-211
RDTSC		13									32
RDPMC		23									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	μops	un	fuse	ed d	oma	ain		Unit	Laten- cy	Reci- procal
		do- main	p015	p0	p1	р5	p2	р3	p4			through- put
Move instructions												
FLD	r	1	1	1						float	1	1
FLD	m32/64	1	1				1			float	3	1
FLD	m80	4	2	2			2			float	4	3
FBLD	m80	40	38	Х	Х	Х	2			float	45	20
FST(P)	r	1	1	1						float	1	1
FST(P)	m32/m64	1						1	1	float	3	1
FSTP	m80	7	3	Х	Х	Х		2	2	float	4	5
FBSTP	m80	171	167	Х	Х	х		2	2	float	164	166
FXCH	r	1	0 f)							float	0	1
FILD	m	1	1		1		1			float	6	1
FIST	m	2	1		1			1	1	float	6	1
FISTP	m	3	1		1			1	1	float	6	1
FISTTP g)	m	3	1		1			1	1	float	6	1
FLDZ		1	1	1						float		1
FLD1		2	2	1	1					float		2
FLDPI FLDL2E etc.		2	2		2					float		2
FCMOVcc	r	2	2	2						float	2	2
FNSTSW	AX	1	1	1						float		1
FNSTSW	m16	2	1	1				1	1	float		2
FLDCW	m16	2	1				1			float		10
FNSTCW	m16	3	1			1		1	1	float		8
FINCSTP FDECSTP		1	1	1						float	1	1
FFREE(P)	r	2	2	Х	Х	Х				float		2
FNSAVE	m	141	95	Х	Х	Х	7	23	23	float		142

Arithmetic instructions   FADD(P) FSUB(R)(P)   r	FRSTOR	m	78	51	x	x	x	27	float		177
FADD(P) FSUB(R)(P) FMUL(P) FMU	Arithmetic instructions										
FADD(P) FSUB(R)(P) FMUL(P) FMU	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)	1 ' '	m	1	1	1			1	float		
FDIV(R)(P)	, ,	r	1	1	1				float	6-21 d)	5-20 d)
FABS	1 ' ' ' '	m	1	1	1			1	float	6-21 d)	, ,
FCOM(P) FUCOM			1	1	1				float	1	. 1
FCOM(P) FUCOM	FCHS		1	1	1				float	1	1
FCOM(P) FUCOM   TOOM(P)   FUCOM(P)   FUCOM		r	1	1		1			float		1
FCOMPP FUCOMIPP FUCOMI(P)   FUCOMI(P)   FUCOMI(P) FUCOMI(P)   FU	1 ' '	m	1	1		1		1			
FCOMI(P) FUCOMI(P) FIADD FISUB(R) FIMUL FIADD FISUB(R) FIDIV(R) FIDIV(R) FICOM(P) FI	1 ' '			2	1	1					
FIADD FISUB(R)    m		r			-						1
FIMUL         m         2         2         1         1         1         float         5         2           FIDIV(R)         m         2         2         1	, , , , , , , , , , , , , , , , , , , ,					2		1		3	
FIDIV(R)         m         2         2         1         1         1         float	` '				1			1			
FICOM(P)         m         2         2         2         2         1         float	_				-	-		•			
FTST         1	, ,		1		'			•		0 -	,
FXAM         1	, ,							•			
FPREM         26-29         x			1								
FPREM1         28-35         x				'	Y		¥			13-40	'
T-19											
Math         28         28         x         x         x         float         43           FXTRACT         53-84         x         x         x         x         float         ~170           FSQRT         1         1         1         float         6-20           FSIN         18-85         x         x         x         float         32-85           FCOS         76-100         x         x         x         x         float         70-100           FSINCOS         105         x         x         x         x         float         70-100           F2XM1         19         19         x         x         x         float         45           FYL2X FYL2XP1         57-65         x         x         x         float         50-100           FPTAN         19-100         x         x         x         float         40-130           FPATAN         23-87         x         x         x         float         55-130           Other           FNOP         1         1         1         float         1           WAIT         2         2         x											
FSCALE         28         28         x<											
FXTRACT         53-84         x <td< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></td<>											
FSQRT         1 <td></td> <td></td> <td>1</td> <td>28</td> <td>Х</td> <td>Х</td> <td>X</td> <td></td> <td></td> <td></td> <td></td>			1	28	Х	Х	X				
FSIN	FXTRACT		53-84		Х	Х	X		float	~170	
FCOS         76-100         x	FSQRT		1	1	1				float	6-20	
FSINCOS F2XM1 FYL2X FYL2XP1 FPTAN FPATAN  Cother FNOP WAIT FNCLEX FIGURE 105  18- 105	FSIN		18-85		Х	Х	X		float	32-85	
FSINCOS         105         x	FCOS		76-100		Х	Х	Х		float	70-100	
F2XM1       19       19       x </td <td></td> <td></td> <td>18-</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>			18-								
FYL2X FYL2XP1         57-65         x x x x         float 50-100 float 40-130 float 55-130           FPATAN         23-87         x x x x         float 55-130           Other         1 1 1 1 float WAIT         float 1 f	FSINCOS		105		Х	Х	X		float	38-107	
FPTAN         19-100         x			1	19	Х	Х	Х				
PPATAN         23-87         x	FYL2X FYL2XP1		57-65		Х	Х	X		float	50-100	
Other         1         1         1         float         1           WAIT         2         2         x         x         x         float         1           FNCLEX         4         4         x         x         x         float         15	FPTAN		19-100		Х	Х	х		float	40-130	
FNOP         1	FPATAN		23-87	ı	x	х	x		float	55-130	
FNOP         1	Other										
WAIT         2         2         x         x         x         float         1           FNCLEX         4         4         x         x         x         float         15			1	1	1				float		1
FNCLEX 4 4 x x   float   15			1		-	x	x				
			1		^						
	FNINIT		15	15	X	x	X		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	<b>p5</b>	p2	рЗ	p4			through- put
Move instructions												
MOVD k)	r,(x)mm	1	1	Х	X	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		^	2			int	2-8	2
MOVDQ2Q		1	1				~			int	1	0.33
	mm, xmm	1		X	X	X				_	1	0.33
MOVQ2DQ	xmm,mm		'	X	X	X		1	1	int	I	
MOVNTQ	m64,mm	1						1	1			2
MOVNTDQ	m128,xmm	1						1	1			2
MOVNTDQA j)	xmm, m128	1					1				2	1
PACKSSWB/DW				١.								
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 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) 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	mm,mm,i	1	1			1				int	1	1
PSHUFW	mm,m64,i	2	1			1	1			int		1
PSHUFD	xmm,xmm,i	1	1			1				int	1	1
PSHUFD	xmm,m128,i	2	1			1	1			int		1
PSHUFL/HW	xmm,xmm,i	1	1			1				int	1	1
PSHUFL/HW	x, m128,i	2	1			1	1			int		1
PALIGNR h)	mm,mm,i	2	2			2				int	2	1
PALIGNR h)	mm,m64,i	3	3			3	1			int		1 1
PALIGNR h)	xmm,xmm,i	1	1			1				int	1	1 1
PALIGNR h)	xmm,m128,i	1	1			1	1			int		1 1
PBLENDVB j)	x,x,xmm0	2	2			2				int	2	2
PBLENDVB j)	x,m,xmm0	2	2			2	1			int	_	2
PBLENDW j)	xmm,xmm,i	1	1			1	'			int	1	1
PBLENDW j)	xmm,m,i	1	1			1	1			int		1 1
MASKMOVQ	mm,mm	4		1		'	1	1	1	int		2-5
MASKMOVDQU	xmm,xmm	10	4			3	2	2	3	int		6-10
PMOVMSKB	· ·	10	1	1		3	_	_	3		2	1
	r32,(x)mm	-	1 -	-	.,					int	2	
PEXTRB j)	r32,xmm,i	2	2	X	X	X				int	3	1
PEXTRB j)	m8,xmm,i	2	2	X	X	X				int	3	
PEXTRW	r32,(x)mm,i	2	2	X	X	X	1			int	3	1
PEXTRW j)	m16,(x)mm,i	2	2	?	?	1		1	1	int	_	1
PEXTRD j)	r32,xmm,i	2	2	X	Х	X				int	3	1
PEXTRD j)	m32,xmm,i	2	1			1		1	1	int		1
PEXTRQ j,m)	r64,xmm,i	2	2	Х	Х	Х				int	3	1
PEXTRQ j,m)	m64,xmm,i	2	1			1		1	1	int		1
PINSRB j)	xmm,r32,i	1	1			1				int	1	1
PINSRB j)	xmm,m8,i	2	1			1	1			int		1
PINSRW	(x)mm,r32,i	1	1			1				int	2	1
PINSRW	(x)mm,m16,i	2	1			1	1			int		1
PINSRD j)	xmm,r32,i	1	1			1				int	1	1 1
PINSRD j)	xmm,m32,i	2	1			1	1			int		1 1
PINSRQ j,m)	xmm,r64,i	1	1			1				int	1	1
PINSRQ j,m)	xmm,m64,i	2	1			1	1			int		1
,,,	7,	_	•				•					.
Arithmetic instructions												
PADD/SUB(U)(S)B/W/D		1	1	x		x				int	1	0.5
PADD/SUB(U)(S)B/W/D	(x)mm,m	1		x		x	1			int	'	1
PADDQ PSUBQ	V,V	2	2	x		x	'			int	2	1 1
PADDQ PSUBQ		2	2				1			int		
	(x)mm,m	2	4	X		X				шц		I
PHADD(S)W PHSUB(S)W h)		2		1						:4		
` ' '	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		2
PCMPEQ/GTB/W/D	V,V	1	1	Х		Х				int	1	0.5
PCMPEQ/GTB/W/D	(x)mm,m	1	1	Х		Х	1			int		1
PCMPEQQ j)	xmm,xmm	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					int	3	1
PMULHRSW h)	(x)mm,m	1	1		1		1			int		1
,				1		'	'	'	1 1		1	' '

<b> </b>				ı		۱ ـ		1	1		_	
PMULLD j)	xmm,xmm	4	4		2	2				int	5	2
PMULLD j)	xmm,m128	6	5	1	2	2	1			int	5	4
PMULDQ j)	xmm,xmm	1	1		1					int	3	1
PMULDQ j)	xmm,m128	1	1		1		1			int		1
PMULUDQ	V,V	1	1		1					int	3	1
PMULUDQ	(x)mm,m	1	1		1		1			int		1
PMADDWD	V,V	1	1		1					int	3	1
PMADDWD	(x)mm,m	1	1		1		1			int		1
PMADDUBSW h)	V,V	1	1		1					int	3	1
PMADDUBSW h)	(x)mm,m	1	1		1		1			int		1
PAVGB/W	V,V	1	1	x	-	x				int	1	0.5
PAVGB/W	(x)mm,m	1	1	X		X	1			int	•	1
PMIN/MAXSB j)	xmm,xmm	1	1	1		^	'			int	1	1
PMIN/MAXSB j)	xmm,m128	1		1			1			int	'	1
PMIN/MAXUB	·	1		1			'			int	1	0.5
	V,V	1		X		X	1				ı	
PMIN/MAXUB	(x)mm,m	•		X		X	'			int	4	1
PMIN/MAXSW	V,V	1	1	X		Х				int	1	0.5
PMIN/MAXSW	(x)mm,m	1	1	X		Х	1			int		1
PMIN/MAXUW j)	xmm,xmm	1	1	1						int	1	1
PMIN/MAXUW j)	xmm,m	1	1				1			int		1
PMIN/MAXSD j)	xmm,xmm	1	1	1						int	1	1
PMIN/MAXSD j)	xmm,m128	1	1	1			1			int		1
PMIN/MAXUD j)	xmm,xmm	1	1	1						int	1	1
PMIN/MAXUD j)	xmm,m128	1	1	1			1			int		1
PHMINPOSUW j)	xmm,xmm	4	4			4				int	4	4
PHMINPOSUW j)	xmm,m128	4	4			4	1			int		4
PABSB PABSW PABSD h		1	1	x		x				int	1	0.5
PABSB PABSW PABSD	,											
h)	(x)mm,m	1	1	x		x	1			int		1
PSIGNB PSIGNW												
PSIGND h)	V,V	1	1	X		х				int	1	0.5
PSIGNB PSIGNW		•									•	0.0
PSIGND h)	(x)mm,m	1	1	X		x	1			int		1
PSADBW	V,V	1	1	^	1	^	·			int	3	1
PSADBW	(x)mm,m	1	1		1		1			int	Ü	1
MPSADBW j)	xmm,xmm,i	3	3		1	2	'			int	5	2
MPSADBW j)	xmm,m,i	4	3			2	1			int	3	2
WFSADBW J)	XIIIIII,III,I	4	3		'	~	'			Ш		2
Logio inotructions												
Logic instructions		4	4			,				:4	4	0.33
PAND(N) POR PXOR	V,V	1	1	X	X	X				int	1	0.33
PAND(N) POR PXOR	(x)mm,m	1	1	X	X	X	1			int	4	1
PTEST j)	xmm,xmm	2	2	1	X	X				int	1	1
PTEST j)	xmm,m128	2	2	1	Х	Х	1			int	_	1
PSLL/RL/RAW/D/Q	mm,mm/i	1	1	1						int	1	1
PSLL/RL/RAW/D/Q	mm,m64	1	1	1			1			int		1
PSLL/RL/RAW/D/Q	xmm,i	1	1	1						int	1	1
PSLL/RL/RAW/D/Q	xmm,xmm	2	2	Х		Х				int	2	1
PSLL/RL/RAW/D/Q	xmm,m128	3	2	Х		х	1			int		1
PSLL/RLDQ	xmm,i	1	1	Х		х				int	1	1
Other												
EMMS		11	11	Х	Х	Х				float		6
										-		

Notes:

- g) SSE3 instruction set.
- h) Supplementary SSE3 instruction set.
- j) SSE4.1 instruction set
- k) MASM uses the name MOVD rather than MOVQ for this instruction even
  - when moving 64 bits
- m) Only available in 64 bit mode

# Floating point XMM instructions

Instruction	Operands	μορs fused	μops	un	fuse	ed d	oma	ain		Unit		Reci- procal
		do-	p015	p0	p1	р5	p2	р3	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				i	i				
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	'	'		'		'			noat		'	
Arithmetic													
ADDSS/D SUBSS/D	xmm,xmm	1	1		1					float	3	1	
ADDSS/D SUBSS/D	x,m32/64	1	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		1			float		1	
ADDSUBPS/D g)	xmm,xmm	1	1		1		'			float	3	1	
ADDSUBPS/D g)	xmm,m128	1	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		1	2	1			float	'	3	
HADDPD HSUBPD g)	xmm,xmm	3	3		X		'			float	6	1.5	
	xmm,m128	4	3	X		X	1					1.5	
HADDPD HSUBPD g) MULSS	· · · · · · · · · · · · · · · · · · ·		) 1	X	X	X	'			float float	4		
	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		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					float	3	1
MAXSS/D MINSS/D	x,m32/64	1	1		1		1			float		1
MAXPS/D MINPS/D	xmm,xmm	1	1		1					float	3	1
MAXPS/D MINPS/D	xmm,m128	1	1		1		1			float		1
ROUNDSS/D j)	xmm,xmm,i	1	1		1					float	3	1
ROUNDSS/D j)	xmm,m128,i	1	1		1		1			float		1
ROUNDPS/D j)	xmm,xmm,i	1	1		1					float	3	1
ROUNDPS/D j)	xmm,m128,i	1	1		1		1			float		1
DPPS j)	xmm,xmm,i	4	4	2	2					float	11	3
DPPS j)	xmm,m128,i	4	4	2	2		1			float		3
DPPD j)	xmm,xmm,i	4	4	Х	х	х				float	9	3
DPPD j)	xmm,m128,i	4	4	х	х	х	1			float		3
Math												
SQRTSS/PS	xmm,xmm	1	1	1						float	6-13	5-12
SQRTSS/PS	xmm,m	2	1	1			1			float		5-12
SQRTSD/PD	xmm,xmm	1	1	1						float	6-20	5-19
SQRTSD/PD	xmm,m	2	1	1			1			float		5-19
RSQRTSS/PS	xmm,xmm	1	1		1					float	3	2
RSQRTSS/PS	xmm,m	1	1		1		1			float		2
Logic												
AND/ANDN/OR/XORPS/D	xmm,xmm	1	1	Х	х	Х				int	1	0.33
AND/ANDN/OR/XORPS/D	xmm,m128	1	1	х	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:

Round divisors give low values.

d) 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  $\mu$ op fusion if the sum of the numbers listed under p015 + p2 + p3 + p4 exceeds the number listed under  $\mu$ ops fused domain. An x under p0, p1 or p5 means that at least one of the  $\mu$ 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  $\mu$ 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  $\mu$ 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  $\mu$ op in one domain is read by a  $\mu$ 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	<b>p5</b>	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	X	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	X	Х	6			int		15
PREFETCHNTA	m	1					1			int		1

PREFETCHT0/1/2	m	1				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	x	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	X	Х				int	15	2
AAM i)		5	5	X	Х	X				int	20	7
MUL IMUL	r8	1	1		1					int	3	1
MUL IMUL	r16	3	3	X	X	Х				int	5	2
MUL IMUL	r32	3	3	X	X	X				int	5	2
MUL IMUL	r64	3	3	X	X	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	ľ			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	X				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	7-11 7-12
IDIV c)	r32	7	7	X	3	X				int	17-28	7-12 7-17
IDIV c)	r64	~60	X	X	X	X				int	37-100	26-86
CBW CWDE CDQE	107	1	1	X	X	X				int	1	1
CWD CDQ CQO		1	1	X	^	X				int		1
POPCNT ()	r,r	1	1	^	1	^				int	3	1
POPCNT ()	r,m	1	1		1		1			int	3	1
CRC32 ()		1	1		1		'			int	3	1
CRC32 ()	r,r r m	1	1		1		1			int	3	1
UNU32 1)	r,m	1	ı							HIL	1	ı

						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	х	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	х	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	х	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	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	х		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	i	1 3 39	1 2 39			1	1			int int int		2 2 120
RETF	i	40	40							int		124
BOUND i)	r,m	15	13				2			int		7
INTO i)	,,,,,	4	4				_			int		5
,												
String instructions												
LODS		2	1	х	Х	Х	1			int		1
REP LODS		11+4n							.	int	40+12n	
STOS		3	1	Х	Х	Х		1	1	int		1
REP STOS	small n	60+n					!	'	.	int	12+n	
REP STOS	large n	2.5/16	bytes							int	1 clk / 1	6 bytes
MOVS		5	2	Х	Х	Х	1	1	1	int		4
REP MOVS	small n	13+6n					1	'	'	int	12+n	
REP MOVS	large n	2/16 by	tes							int	1 clk / 1	6 bytes
SCAS		3	2	Х	Х	Х	1			int		1
REP SCAS		37+6n					ı	•	.	int	40+2n	
CMPS		5	3	х	Х	х	2			int		4
REP CMPS		65+8n		1			ı	ı	'	int	42+2n	
Other												
NOP (90)		1	1	х	Χ	Х				int		0.33
Long NOP (0F 1F)		1	1	Х	Χ	Х				int		1
PAUSE		5	5	Х	Χ	Х				int		9
ENTER	a,0	11	9	Х	Χ	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	un	fuse	ed d	oma	ain		Do- main	Laten- cy	procal
		do-	p015	p0	<b>p1</b>	р5	p2	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

l		i.	1 .	ı	ı	ı	1	1 -		_	1 1	ı
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	3	1		1			1	1	float	7	1
FISTTP g)		3	1		1			1			7	
· · · · · · · · · · · · · · · · · · ·	m m		1						'	float	'	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	2
FNSTSW	AX	2	2							float		1
FNSTSW	m16	3	2					1	1	float		2
FLDCW	m16	2					1	'	'		7	31
			1	_			1			float	7	
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	143	89	х	х	х	8	23	23	float	178	178
FRSTOR	m	79	52	х	х	x	27			float	156	156
		. •	-	^								
Arithmetic instructions												
	_		1		4					£1 4	_	4
FADD(P) FSUB(R)(P)	r	1	1		1					float	3	1
FADD(P) FSUB(R)(P)	m m	1	1		1		1			float		1
FMUL(P)	r	1	1	1						float	5	1
FMUL(P)	m	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							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	2	2		2		1			float	3	2
FIMUL	m	2	2	1	1		1			float	5	2
FIDIV(R)	m	2	2	1	1		1			float	7-27 d)	
, ,											1-21 u)	. 1
FICOM(P)	m	2	2		2		1			float		1
FTST		1	1		1					float		1
FXAM		1	1		1					float		1
FPREM		25	25	Х	Х	Х				float	14	
FPREM1		35	35	х	х	х				float	19	
FRNDINT		17	17	х	х	х				float	22	
Math												
FSCALE		24	24	х	Х	Х				float	12	
FXTRACT		17	17	X	X	X				float	13	
FSQRT		17	1	1	^	^				float	~27	
				-								
FSIN		~100	~100	Х	Х	Х				float	40-100	
FCOS		~100	~100	Х	Х	X				float	40-100	
FSINCOS		~100	~100	Х	Х	Х				float	~110	
F2XM1		19	19	Х	Х	Х				float	58	
FYL2X FYL2XP1		~55	~55	Х	х	х				float	~80	
FPTAN		~100	~100	Х	Х	Х				float	~115	
FPATAN		~82	~82							float	~120	
LEVIVI		-02	-02	Х	X	X				IIUal	120	

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

Notes:

d) Round divisors or low precision give low values.

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

g) SSE3 instruction set.

# Integer MMX and XMM instructions

Instruction	Operands	μοps fused	μops	un	fus	ed d	oma	ain		Do- main	Laten- cy	Reci- procal
		do-	p015	<b>p0</b>	<b>p1</b>	р5	p2	р3	p4	1		through-
Move instructions		main	Ī	1	1		•	1	1			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	Х		Х				ivec	1	0.5
PSHUFB h)	(x)mm,m	2	1	Х		Х	1					1
PSHUFW	mm,mm,i	1	1	х		х				ivec	1	0.5
PSHUFW	mm,m64,i	2	1	х		х	1					1
PSHUFD	xmm,xmm,i	1	1	х		Х				ivec	1	0.5
PSHUFD	xmm,m128,i	2	1	х		х	1					1
PSHUFL/HW	xmm,xmm,i	1	1	x		х				ivec	1	0.5
PSHUFL/HW	xmm, m128,i	2	1	X		Х	1				-	1
PALIGNR h)	(x)mm,(x)mm,i	1	1	X		Х				ivec	1	1
PALIGNR h)	(x)mm,m,i	2	1	X		Х	1				•	1
PBLENDVB j)	x,x,xmm0	2	2	1		1	•			ivec	2	1
PBLENDVB j)	xmm,m,xmm0	3	2	1		1	1			1700	_	1
PBLENDW j)	xmm,xmm,i	1	1	X		X	'			ivec	1	0.5
PBLENDW j)	xmm,m,i	2	1	x		X	1			1000	'	1
MASKMOVQ	mm,mm	4	1	1		^	1	1	1	ivec		2
MASKMOVDQU	xmm,xmm	10	4	X	х	х	2	2	'	ivec		7
PMOVMSKB	r32,(x)mm	10	1	1	۸	^	_	~	^	float	2+2	1
PEXTRB j)	r32,xmm,i	2	2	-	v	v				ivec	2+2 2+1	1
PEXTRB j)	m8,xmm,i	2	2	X	Х	X				IVEC	2+1	1
PEXTRW		2	2	X	V	X				ivoo	2+1	1
	r32,(x)mm,i	2		X	Х	X		4	1	ivec	271	1
PEXTRW j)	m16,(x)mm,i	2	2 2	X	.,	X		1	'	ivoo	2+1	1
PEXTRD j)	r32,xmm,i	2	1	X	Х	X		4	1	ivec	2+1	1
PEXTRD j)	m32,xmm,i			X	.,	X		1	'		2.4	1
PEXTRQ j,m)	r64,xmm,i	2	2	X	Х	X		_	,	ivec	2+1	1
PEXTRQ j,m)	m64,xmm,i	2	1	X		X		1	1		4.4	1
PINSRB j)	xmm,r32,i	1	1	Х		Х	_			ivec	1+1	1
PINSRB j)	xmm,m8,i	2	1	X		Х	1				4.4	1
PINSRW	(x)mm,r32,i	1	1	Х		Х				ivec	1+1	1
PINSRW	(x)mm,m16,i	2	1	X		Х	1				4.4	1
PINSRD j)	xmm,r32,i	1	1	X		Х	_			ivec	1+1	1
PINSRD j)	xmm,m32,i	2	1	X		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)		_									_	0.5
(S)B/W/D/Q	(x)mm, (x)mm	1	1	X		Х				ivec	1	0.5
PADD/SUB(U)	()						4					•
(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	X		Х	1					3
PCMPEQ/GTB/W/D	(x)mm,(x)mm	1	1	X		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	X		Х	1					2

PCMPGTQ ()	xmm,xmm	1	1	ı	1		l	ivec	3	1 1
PCMPGTQ ()	xmm,m128	1	1		1		1	1060	3	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	IVEC	3	1
PMULHRSW h)	(x)mm,(x)mm	1	1		1		'	ivec	3	1
PMULHRSW h)	(x)mm,m	1	1		1		1	1060	3	1
PMULLD j)	xmm,xmm	2	2		2		'	ivec	6	2
PMULLD j)	xmm,m128	3	2		2		1	1060	U	
PMULDQ j)	xmm,xmm	1	1		1		'	ivec	3	1
PMULDQ j)	xmm,m128	1	1		1		1	ivec	3	1
PMULUDQ J)	(x)mm,(x)mm	1	1		1		1	ivoo	3	1
PMULUDQ	` ' ' ' '	1	1		1		1	ivec	3	
PMADDWD	(x)mm,m	1	1		1		1	ivoo	3	1
PMADDWD	(x)mm,(x)mm	1	1		1		1	ivec	3	1
PMADDUBSW h)	(x)mm,m	1	1		1		1	ivoo	3	1
PMADDUBSW h)	(x)mm,(x)mm	1	1		1		1	ivec	3	1
PAVGB/W	(x)mm,m	1	1		1	.,	1	ivoo	1	0.5
	(x)mm,(x)mm	-	-	X		X	1	ivec	I	
PAVGB/W PMIN/MAXSB j)	(x)mm,m	1 1	1	X		X	I	ivoo	1	1
PMIN/MAXSB j)	xmm,xmm xmm,m128	1	1	X		X	1	ivec	1	2
PMIN/MAXUB	(x)mm,(x)mm	1	1	X			1	ivoo	1	0.5
PMIN/MAXUB	(x)mm,m	1	1	X		X	1	ivec	ı	2
PMIN/MAXSW	(x)mm,(x)mm	1	1	X		X	ı	ivec	1	0.5
PMIN/MAXSW	(x)mm,m	1	1	^		x	1	1060		2
PMIN/MAXUW j)	xmm,xmm	1	1	X		X	ı	ivec	1	1
PMIN/MAXUW j)	xmm,m	1	1	^		X	1	1060		2
PMIN/MAXU/SD j)	xmm,xmm	1	1	^		X	'	ivec	1	1
PMIN/MAXU/SD j)	xmm,m128	1	1	^		X	1	1060		2
PHMINPOSUW j)	xmm,xmm	1	1	^	1	^	'	ivec	3	1
PHMINPOSUW j)	xmm,m128	1	1		1		1	IVEC	3	3
PABSB PABSW PABSD	XIIIII,III 120	•	'		'		'			
h)	(x)mm,(x)mm	1	1	X		х		ivec	1	0.5
PABSB PABSW PABSD	(**)***********************************	•	'	^		^		1,000	•	0.0
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	(**)***********************************	·							•	0.0
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	X	х	х		ivec	5	1
MPSADBW j)	xmm,m,i	4	3	X	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										0.00
PAND(N) POR PXOR	(x)mm,(x)mm	1	1	Х	Х	Х	_	ivec	1	0.33
PAND(N) POR PXOR	(x)mm,m	1	1	X	Х	Χ	1			1

PTEST j)	xmm,xmm	2	2	х	Х	Х		ivec	3	1
PTEST j)	xmm,m128	2	2	Х	Χ	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		Χ		ivec	1	1
String instructions										
PCMPESTRI ()	xmm,xmm,i	8	8	Х	Χ	Χ		ivec	14	5
PCMPESTRI ()	xmm,m128,i	9	8	Х	Χ	Χ	1	ivec	14	6
PCMPESTRM ()	xmm,xmm,i	9	9	Х	Χ	Χ		ivec	7	6
PCMPESTRM ℓ)	xmm,m128,i	10	10	Х	Χ	Χ	1	ivec	7	6
PCMPISTRI ()	xmm,xmm,i	3	3	Х	Χ	Χ		ivec	8	2
PCMPISTRI ()	xmm,m128,i	4	4	Х	Χ	Χ	1	ivec	8	2
PCMPISTRM ℓ)	xmm,xmm,i	4	4	Х	Χ	Χ		ivec	7	2
PCMPISTRM ()	xmm,m128,i	6	5	Х	Χ	Χ	1	ivec	7	5
Other										
EMMS		11	11	Х	Χ	Χ		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	μops	un	fus	ed d	oma	ain		Do- main	Laten- cy	Reci- procal
		do-	p015	<b>p0</b>	<b>p1</b>	р5	<b>p2</b>	р3	p4			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	1			1	1			float		1	
BLENDVPS/PD j)	x,x,xmm0	2	2			2				float	2	2	
BLENDVPS/PD j)	xmm,m,xmm0	3	2			2	1			float		2	
MOVDDUP g)	xmm,xmm	1	1			1				float	1	1	
MOVDDUP g)	xmm,m64	1					1				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	
	r32,xmm,i	1	1			1	'			float	1+2	1	
EXTRACTPS j)		2	1			1		1	1	lioat	172	1	
EXTRACTPS j)	m32,xmm,i		-			-		'	1	414	4		
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	0.2	1	
CVTDQ2PD	xmm,xmm	2	2		1	1	'			float	4+2	1	
CVTDQ2PD	xmm,m64	2	2		1	1	1			float	7.2	1	
CVTDQ2FD CVT(T)PD2DQ		2	2		1	1	'			float	4+2	1	
, ,	xmm,xmm	2	2		1	1	4				4 <b>T</b> Z		
CVT(T)PD2DQ	xmm,m128				-	'	1			float	0.0	1	
CVTPI2PS	xmm,mm	1	1		1		_			float	3+2	3	
CVTPI2PS	xmm,m64	1	1		1		1			float		3	
CVT(T)PS2PI	mm,xmm	1	1		1					float	3+2	1	
CVT(T)PS2PI	mm,m128	1	1		1		1			float		1	
CVTPI2PD	xmm,mm	2	2		1	1				ivec/float	6	1	
CVTPI2PD	xmm,m64	2	2		1	1	1					1	
CVT(T) PD2PI	mm,xmm	2	2	Х	1	Х				float/ivec	6	1	
CVT(T) PD2PI	mm,m128	2	2	Х	1	Х	1					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	

1	I	1 .		I		1	1	ı		<b>.</b> .		
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	, 22	7-22
RCPSS/PS	xmm,xmm	1	1	'	1		l '			float	3	2
RCPSS/PS	· ·	1	1		1		1			float		2
	xmm,m	1	'		1		'			iloat		2
CMPccSS/D CMPccPS/D	vmm vmm	1	1		1					floot	3	1
CMDCC/D CMDDC/D	xmm,xmm	I	1		ı					float	3	ı
CMPccSS/D CMPccPS/D		_					,			fl 4		4
	xmm,m	2	1		1		1			float	4.0	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	Х	Х	х	1			float		3
,												
Math												
SQRTSS/PS	xmm,xmm	1	1	1						float	7-18	7-18
SQRTSS/PS	xmm,m	2	1	1			1			float		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
KOGK 100/10	AIIIII,III	'	'		'		'			iloat		2
Logic												
Logic AND/ANDN/OR/XORPS/D	vmm vmm	1	1			1				float	1	1
	xmm,xmm	1	1			1	1				'	1
AND/ANDN/OR/XORPS/D	xmm,m128	I	1			'	'			float		ı
Other	22		_				_					-
LDMXCSR	m32	6	6	Х	Х	X	1					5
STMXCSR	m32	2	1			1		1	1			1
FXSAVE	m4096	141	141	Х	Χ	Х	5	38	38		90	90
FXRSTOR	m4096	112	90	Χ	Χ	Χ	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 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  $\mu$ op fusion if the sum of the numbers listed under p015 + p23 + p4 exceeds the number listed under  $\mu$ 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  $\mu$ op in the second clock cycle, but a read port can receive an address-calculation  $\mu$ op in the second clock cycle. An x under p0, p1 or p5 means that at least one of the  $\mu$ 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  $\mu$ 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  $\mu$ 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								Latency	Reci-	Com-
		fused do- main	p015	p0	p1	<b>p</b> 5	p23	p4		procal through- put	ments
Move instructions											
MOV	r,r/i	1	1	х	х	х			1		

MOV				- ,	- 3 -							
MOV         m,r         1 <td>MOV</td> <td>r,m</td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td>1</td> <td></td> <td>2</td> <td>0.5</td> <td>dressing</td>	MOV	r,m	1					1		2	0.5	dressing
MOV												modes
MOVNTI										3		
MOVSX MOVZX   NOVSXD		m,i										
MOVSXD		m,r						1	1	~350	1	
MOVSXD   CMOVcc		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	x	х	x			2		
XLAT					' '			2	1		-	implicit
PUSH		.,						_				
PUSH			3	2				1				
PUSHF(D/Q)			1					1	1	3		
PUSHA(D)		i						1	1		1	
PUSHA(D)   POP	PUSH	m	2					2	1		1	
POP POP         r         1         0         1         1         2         0.5         0.5           POP POP POPF (D/Q)         m         2         2         1	PUSHF(D/Q)		3	2	х	Х	х	1	1		1	
POP	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	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	v						
BSWAP BSWAP R64 RFEFETCHNTA RM RFENCE RFENCE RFENCE RAITHMETIC INSTRUCTIONS ADD SUB AD					^							
BSWAP BSWAP RF64 RF64 RFEFETCHNTA RFEFETCHT0/1/2 RFFETCHT0/1/2 RFFENCE RFFFENCE RFFFENCE RFFFENCE RFFFENCE RFFFFENCE RFFFFENCE RFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF		1,111	'			'				0	'	
BSWAP												
BSWAP	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           Arithmetic instructions         7,r/i         1         1         1         1         1         1         1         6           ADD SUB         r,m         1         1         1         1         1         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.5         0.25         0.5         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0.25         0	BSWAP			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         r,same         1         0         0.5         0.5         0.5         0.25				_				1		_		
LFENCE												
MFENCE         3         1         1         1         1         1         1         33         6           Arithmetic instructions         r,r/i         1         1         x         x         x         1         1         0         0         0.5         0         0.5         0         0.5         0         0.5         0         0.25         0         0.25         0         0.25         0         0.25         0         0.25         0         0.25         0         0.25         0         0.25         0         0.25         0         0.25         0         0         0.25         0         0.25         0         0         0.25         0         0         0.25         0         0         0.25         0         0         0.25         0         0         0.25         0         0         0.25         0         0         0.25         0         0         0.25         0         0         0.25         0									1			
Arithmetic instructions         r,r/i         1         2         2         1         2         2         1         3         2         2         1         3         2         2         1         3         3         3         3         3         3         3         3         3         4         3         3         4         3         4         3         4         3         4         3         4         3         4         3         4<				1								
ADD SUB         r,r/i         1         1         x         x         x         1         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         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           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         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         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         m,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,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												
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		
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			-	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	X	2	1		I -	
ADC SBB											0.25	
ADC SBB					Х	Х	Х					
CMP         r,r/i         1         1         x         x         x         1         1         CMP         m,r/i         1         1         x         x         x         1         1         0.5         1         0.5         1         1         x         x         x         1         1         0.5         1         0.5         1         0.5         1         0.5		r,m	2		Х	Х	Х					
CMP	ADC SBB	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	

		-	,								
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	
MUL IMUL	r16	4	4						4	2	
MUL IMUL	r32	3	3						4	2	
MUL IMUL	r64	2	2						3	1	
IMUL	r,r	1	1		1				3	1	
IMUL	r16,r16,i	2	2						4	1	
IMUL	r32,r32,i	1	1		1				3	1	
IMUL	r64,r64,i	1	1		1				3	1	
MUL IMUL	m8	1	1		1		1		3	1	
MUL IMUL	m16	4	3				1			2	
MUL IMUL	m32	3	2				1			2	
MUL IMUL	m64	2	1				1			2	
IMUL	r,m	1	1		1		1			1	
IMUL	r16,m16,i	2	2				1			1	
IMUL	r32,m32,i	1	1		1		1			1	
IMUL	r64,m64,i	1	1		1		1		00.04	1	
DIV	r8	10	10						20-24	11-14	
DIV	r16	11	11						21-25	11-14	
DIV DIV	r32 r64	10 34-56	10						20-28 30-94	11-18 22-76	
IDIV	r8	10	10						30-94 21-24	11-14	
IDIV	r16	10	10						21-24	11-14	
IDIV	r32	9	9						20-27	11-14	
IDIV	r64	59-	X						40-103	25-84	
	104	138	^						40 100	20 04	
CBW		1	1						1	0.5	
CWDE		1	1			1			1	1	
CDQE		1	1						1	0.5	
CWD		2	2						1	1	
CDQ		1	1						1	1	
CQO		1	1						1	0.5	
POPCNT	r,r	1	1		1				3	1	SSE4.2
POPCNT	r,m	1	1		1		1			1	SSE4.2
CRC32	r,r	1	1		1				3	1	SSE4.2
CRC32	r,m	1	1		1		1			1	SSE4.2
Logic instructions											
AND OR XOR	r,r/i	1	1	х	х	х			1		
AND OR XOR	r,m	1	1	х	х	х	1			0.5	
AND OR XOR	m,r/i	2	1	Х	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	

			,	J							
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	0				_	'	high	high	
		_	2						high		
RCR	r16/32/64,1	3	3						2	2	
RCR	r,i	8	8						5	5	
RCR	m,i	11	7				Х	Х		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			7				.,	.,	U	6	
I	m,cl	11					Х	X			
SHRD SHLD	r,r,i	1	1				_			0.5	
SHRD SHLD	m,r,i	3					2	1		2	
SHRD SHLD	r,r,cl	4	4						2	2	
SHRD SHLD	m,r,cl	5	3				2	1		4	
ВТ	r,r/i	1	1						1	0.5	
ВТ	m,r	10	8				х			5	
BT	m,i	2	1				1			0.5	
BTR BTS BTC	r,r/i	1	1						1	0.5	
BTR BTS BTC		11	7				.,	\ ,	ı		
I	m,r						X	X		5	
BTR BTS BTC	m,i	3	1				2	1		2	
BSF BSR	r,r	1	1						3	1	
BSF BSR	r,m	1	1		1		1			1	
SETcc	r	1	1	Х		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		
CLD STD		3	3	^	^	\ \ \			•	4	
OLD O1D			0								
Control transfer instructi	 										
Control transfer instructi			4			4			^		
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								
CALL		3	2			4	4	4		5 2	
	near					1	1	1		2	
CALL	r	2	1			1	1	1		2	
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	
2000	I	3	_			1	'			1 '	

REP LODS STOS REP STOS		5n+12 3 2n	1		1	1	~2n n	1	worst
REP STOS		1.5/16E	3				1/16B		case best case
MOVS REP MOVS		5 2n					1.5 n	4	worst case
REP MOVS		3/16B					1/16B		best case
SCAS		3					045	1	
REP SCAS CMPS		6n+47 5					2n+45	4	
REP CMPS		8n+80					2n+80		
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	a,0	12	10		2	1		8	
ENTER	a,b	49+6b					84+3b		
LEAVE		3	3		1			7	
CPUID		31-75					100-250		
RDTSC		21						28	
RDTSCP		23						36	
RDPMC		35						42	

Floating point x87 instructions

nstruction	Operands	μops	μops	un	fuse	ed d	Ioma	in	Latency	Reci-	Com-
		fused do- main	p015	p0	p1	р5	p23	p4		procal through- put	ments
Move instructions											
FLD	r	1	1	1					1	1	
FLD	m32/64	1	1				1		3	1	
FLD	m80	4	2	1	1		2		4	2	
FBLD	m80	43	40				3		45	21	
FST(P)	r	1	1	1					1	1	
FST(P)	m32/m64	1					1	1	4	1	
FSTP	m80	7	3				2	2	5	5	
FBSTP	m80	246								252	
FXCH	r	1	0						0	0.5	
FILD	m	1	1		1		1		6	1	
FIST(P)	m	3	1		1		1	1	7	2	
FISTTP	m	3	1		1		1	1	7	2	SSE3
FLDZ		1	1	1						2	
FLD1		2	2	1	1					2	
FLDPI FLDL2E etc.		2	2		2					2	
FCMOVcc	r	3	3						3	2	

FNSTSW 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	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 53-118 102 28-91	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						Latency	Reci-	Com-
		fused do- main	p015	p0	p1	р5	p23	p4		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	X	X	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	x	x	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	SSE
MOVDQ2Q	mm, x	2	2				'		1	1	JOL.
MOVQ2DQ	x,mm	1	1						1	ļ !	
MOVNTQ	m64,mm	1	'				1	1	~300	1	
MOVNTDQ	m128,x	1					1	1	~300	l	
MOVNTDQ	1	1					1	'	~300	0.5	SSE4
	x, m128	'					'			0.5	33E4
PACKSSWB/DW PACKUSWB		1	4	,					4	4	
	mm,mm	'	1	1					1	1	
PACKSSWB/DW PACKUSWB	ma ma C 4	1	1	1			1				
	mm,m64	'	1	'			'				
PACKSSWB/DW PACKUSWB	V V	1	1			V			1	0.5	
	x,x	'	1		Х	Х			'	0.5	
PACKSSWB/DW PACKUSWB	x,m128	1	1		x	v	1			0.5	
PACKUSDW		1	1			X	'		1	0.5	SSE4
PACKUSDW	X,X	1	1		X	X	1		'	0.5	SSE4
PUNPCKH/LBW/WD/DQ	X,M					X	'		1	0.5	33E4
	(x)mm,(x)mm	1	1		X	X	4		ı		
PUNPCKH/LBW/WD/DQ	(x)mm,m	1			X	X	1		4	0.5	
PUNPCKH/LQDQ	X,X	1	1		X	X	4		1	0.5	
PUNPCKH/LQDQ	x, m128	2	1		X	X	1		4	0.5	SSE4
PMOVSX/ZXBW	X,X	1	1		X	X	1		1	0.5	
PMOVSX/ZXBW	x,m64	1	1		X	X	1		_	0.5	SSE4
PMOVSX/ZXBD	X,X	1	1		Х	Х			1	0.5	SSE4
PMOVSX/ZXBD	x,m32	1	1		Х	Х	1			0.5	SSE4
PMOVSX/ZXBQ	X,X	1	1		Х	Х			1	0.5	SSE4
PMOVSX/ZXBQ	x,m16	1	1		Х	Х	1			0.5	SSE4
PMOVSX/ZXWD	X,X	1	1		Х	Х			1	0.5	SSE4
PMOVSX/ZXWD	x,m64	1	1		Х	Х	1			0.5	SSE4
PMOVSX/ZXWQ	X,X	1	1		Х	Х			1	0.5	SSE4
PMOVSX/ZXWQ	x,m32	1	1		Х	Х	1			0.5	SSE4
PMOVSX/ZXDQ	X,X	1	1		Х	Х			1	0.5	SSE4
PMOVSX/ZXDQ	x,m64	1	1		Х	Х	1			0.5	SSE4
PSHUFB	(x)mm,(x)mm	1	1		Х	Х			1	0.5	SSSE
PSHUFB	(x)mm,m	2	1		Х	Х	1			0.5	SSSE
PSHUFW	mm,mm,i	1	1		Х	Х			1	0.5	
PSHUFW	mm,m64,i	2	1		Х	Х	1			0.5	

PSHUFD	vvi	1	1	1	V	V	1	I	1	0.5	
PSHUFD	x,x,i x,m128,i	1 2	1		X	X	1		1	0.5	
PSHUFL/HW	1 '	1	1		X	X	ı		1	0.5	
PSHUFL/HW	x,x,i x, m128,i	2	1		X	X	1		ı	0.5	
PALIGNR	(x)mm,(x)mm,i	1	1		X	X	1		1	0.5	SSSE3
PALIGNR	' ' ' '	2	1		X	X	1		ı	0.5	SSSE3
PBLENDVB	(x)mm,m,i	2	2		1	1	1		2	1	SSE4.1
PBLENDVB	x,x,xmm0	3	2		1	1	1			1	SSE4.1
PBLENDW	x,m,xmm0	1	1		1	-	1		1	0.5	SSE4.1
PBLENDW	x,x,i	2	1		X	X	1		ı	0.5	SSE4.1
MASKMOVQ	x,m,i	4		1	Х	Х	2	1			33⊑4.1
MASKMOVDQU	mm,mm	10	1 4	'			4	-		1 6	
· ·	X,X			4			4	X	2		
PMOVMSKB	r32,(x)mm	1	1	1					2 2	1	00544
PEXTRB	r32,x,i	2	2	1	X	X	_	4	2	1	SSE4.1
PEXTRB	m8,x,i	2	1	4	X	X	1	1	_	1	SSE4.1
PEXTRW	r32,(x)mm,i	2	2	1	Х	Х		,	2	1	00544
PEXTRW	m16,(x)mm,i	2	1		X	X	1	1	_	2	SSE4.1
PEXTRD	r32,x,i	2	2	1	X	X	_		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		х	Х			1	0.5	
PADD/SUB(U,S)B/W/D/Q	(x)mm,m	1	1		X	X	1		'	0.5	
PHADD/SUB(S)W/D	(x)mm, (x)mm	3	3		X	X	'		2	1.5	SSSE3
PHADD/SUB(S)W/D	(x)mm,m64	4	3		X	X	1			1.5	SSSE3
PCMPEQ/GTB/W/D	(x)mm,(x)mm	1	1		X	X	'		1	0.5	OCOLO
PCMPEQ/GTB/W/D	(x)mm,m	1	1		X	X	1		•	0.5	
PCMPEQQ	X,X	1	1		X	X	'		1	0.5	SSE4.1
PCMPEQQ	x,m128	1	1		X	X	1		•	0.5	SSE4.1
PCMPGTQ	X,11120	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	00L4.2
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,111126	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	JUL4.1
PMULUDQ	(x)mm,m	1	1	1			1			1	
ו ואוטבטטע	(^ <i>)</i> !!!!!,!!!	1	1 '	1			'		I	ı .	

DAMADDIMO	l , , , , , ,				ı	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	4	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	х	х	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	х	х	1		1	
PSLL/RLDQ	x,i	1	1		Х	Х		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,x,i	8	8					11-12	4	SSE4.2
PCMPESTRM	x,m128,i	8	7				1		4	SSE4.2
PCMPISTRI	x,x,i	3	3					3	3	SSE4.2
PCMPISTRI	x,m128,i	4	3				1		3	SSE4.2
PCMPISTRM	x,x,i	3	3				•	11	3	SSE4.2
PCMPISTRM	x,m128,i	4	3				1		3	SSE4.2
									•	
<b>Encryption instructions</b>										
PCLMULQDQ	x,x,i	18	18					14	8	CLMUL

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

Floating point XMM and YMM instructions

Instruction	Operands	μops	μops				oma	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	X		X	1			0.5	SSE4.
VBLENDPS/PD	y,y,i	1	1	X		X			1	1	AVX
VBLENDPS/PD	y,m256,i	2	1	X		X	1+			1	AVX
BLENDVPS/PD	x,x,xmm0	2	2	x		X	'		2	1	SSE4.
BLENDVPS/PD	x,m,xmm0	3	2	X		X	1		_	1	SSE4.
VBLENDVPS/PD	y,y,y,y	2	2	x		X	'		2	1	AVX

			,								
VBLENDVPS/PD	y,y,m,y	3	2	х		Х	1+			1	AVX
MOVDDUP	X,X	1	1			1			1	1	SSE3
MOVDDUP	x,m64	1					1		3	0.5	SSE3
VMOVDDUP	у,у	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	ľ		1	1	SSE3
MOVSH/LDUP	x,m128	1				'	1		3	0.5	SSE3
VMOVSH/LDUP	y,y	1	1			1	ľ		1	1	AVX
VMOVSH/LDUP	y,m256	1	'			ļ .	1+		4	1	AVX
UNPCKH/LPS/D	X,X	1	1			1			1	1	SSE3
UNPCKH/LPS/D	x,m128	1	1			1	1		•	1	SSE3
VUNPCKH/LPS/D	y,y,y	1	1			1	'		1	1	AVX
VUNPCKH/LPS/D	y,y,y y,y,m256	1	1			1	1+		'	1	AVX
EXTRACTPS	r32,x,i	2	2	1		1	١.		2	1	SSE4.1
EXTRACTPS	m32,x,i	3	2	'		1	1	1	2	1	SSE4.1
VEXTRACTF128	x,y,i	1	1			1	'	'	2	1	AVX
VEXTRACTF 128	m128,y,i	2	1			'	1	1	2	1	AVX
INSERTPS	_	1	1			1	ı	'	1	1	SSE4.1
INSERTPS	X,X,İ	2	1			1	1		I	1	SSE4.1
VINSERTF128	x,m32,i	1	1			1	'		2	1	AVX
	y,y,x,i		1			1	4		2		
VINSERTF128	y,y,m128,i	2	1			1	1			1	AVX
VMASKMOVPS/D	x,x,m128	3	2				1			1	AVX
VMASKMOVPS/D	y,y,m256	_	2				1+	,		1	AVX
VMASKMOVPS/D	m128,x,x	4	2				1	1		1	AVX
VMASKMOVPS/D	m256,y,y	4	2				1	1+		2	AVX
Conversion											
CVTPD2PS		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	1+		4	_	AVX
CVTSD2SS	·	2	2 2		1	1	ĮΤ		3	1 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 -		ļ '	1		3		
I	x,m64	2	1	1		4	ı		4	1	A \ / \
VCVTPS2PD	y,x		2	1		1	4		4	1	AVX
VCVTPS2PD	y,m128	3 2	2	1		1	1		2	1	AVX
CVTSS2SD	X,X		2	1		1			3	1	
CVTSS2SD	x,m32	2	1	1			1		0	1	
CVTDQ2PS	X,X	1	1		1		,		3	1	
CVTDQ2PS	x,m128	1	1		1		1		0	1	A) () (
VCVTDQ2PS	у,у	1	1		1		_		3	1	AVX
VCVTDQ2PS		1	1		1		1+			1	AVX
O) (T(T) DOOD O	y,m256	-								4	1
CVT(T) PS2DQ	x,x	1	1		1				3	1	
CVT(T) PS2DQ	x,x x,m128	1	1		1		1			1	
CVT(T) PS2DQ VCVT(T) PS2DQ	x,x x,m128 y,y	1 1 1	1		1				3	1	AVX
CVT(T) PS2DQ VCVT(T) PS2DQ VCVT(T) PS2DQ	x,x x,m128 y,y y,m256	1 1 1	1 1 1		1 1 1		1+		3	1 1 1	AVX AVX
CVT(T) PS2DQ VCVT(T) PS2DQ	x,x x,m128 y,y	1 1 1	1		1	1 1				1	

VCVTDQ2PD		2	2		1	1		5	1	AVX
VCVTDQ2PD VCVTDQ2PD	y,x y,m128	3	2		1	1	1	5	1	AVX
CVT(T)PD2DQ	X,X	2	2		1	1	'	4	1	7//
CVT(T)PD2DQ	x,m128	2	2		1	1	1	7	1	
VCVT(T)PD2DQ	1	2	2		1	1	ı	5	1	AVX
VCVT(T)PD2DQ VCVT(T)PD2DQ	x,y x,m256	2	2		1	1	1+	5	1	AVX
CVTPI2PS		1	1		1 -	ı	17	4	2	AVA
	x,mm				1			4		
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	١.	1		1	_	1.5	
CVT(T)SS2SI	r32,x	2	2	1	1			4	1	
CVT(T)SS2SI	r32,m32	2	2	١.	1		1	_	1	
CVTSI2SD	x,r32	2	2	1	1			4	1.5	
CVTSI2SD	x,m32	1	1	١.	1		1	_	1.5	
CVT(T)SD2SI	r32,x	2	2	1	1			4	1	
CVT(T)SD2SI	r32,m64	2	2	1	1		1		1	
Arithmetic										
ADDSS/D SUBSS/D	X,X	1	1		1			3	1	
ADDSS/D SUBSS/D	x,m32/64	1	1		1		1	3	1	
ADDSS/D SUBPS/D	X,11132704 X,X	1	1		1		'	3	1	
ADDPS/D SUBPS/D	x,m128	1	1		1		1	3	1	
VADDPS/D VSUBPS/D	· ·	1	1		1		'	3	1	AVX
VADDPS/D VSUBPS/D	y,y,y y,y,m256	1	1		1		1+	3	1	AVX
ADDSUBPS/D	• •	1	1		1		1 +	3	1	SSE3
ADDSUBPS/D	x,x x,m128	1	1		1		1	3	1	SSE3
VADDSUBPS/D	· ·	1	1				ı	3	1	AVX
VADDSUBPS/D	y,y,y				1		1+	3		
	y,y,m256	1	1		1	2	1+	_	1 2	AVX SSE3
HADDPS/D HSUBPS/D	X,X	3 4	3		1	2	1	5	2	SSE3
HADDPS/D HSUBPS/D VHADDPS/D	x,m128	4	3		'		ı			ರಿರ⊏ರಿ
VHSUBPS/D	y,y,y	3	3		1	2		5	2	AVX
VHADDPS/D	y,y,y				'	_		9		AVA
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	Ü	1	
VMULPS	y,y,y	1	1	1			'	5	1	AVX
VMULPS	y,y,m256	1	1	1			1+	O	1	AVX
MULSD MULPD	X,X	1	1	1			' '	5	1	7,,,,
MULSD MULPD	x,m	1	1	1			1	9	1	
VMULPD	y,y,y	1	1	1			'	5	1	AVX
VMULPD	y,y,y y,y,m256	1	1	1			1+	3	1	AVX
DIVSS DIVPS		1	1	1			' '	10-14	10-14	717
DIVSS DIVPS	X,X	1	1	1			1	10-14	10-14	
VDIVPS	x,m	3	3	2		1	ı	21-29	20-28	AVX
VDIVPS	y,y,y	4	3	2		1	1+	21-29	20-28	AVX
DIVSD DIVPD	y,y,m256	1	1			1		10-22	10-22	AVA
טועסט טוערט	X,X	I		1	1			10-22	10-22	

			- ,	- 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	J	1	
VRCPPS		3	3	2		1	'	7	2	AVX
	y,y v ==256	4	3	~		'	4.	,	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		1	
MAXPS/D MINPS/D	x,me2/01	1	1		1		'	3	1	
MAXPS/D MINPS/D	x,m128	1	1		1		1	3	1	
VMAXPS/D VMINPS/D		1	1		1		'	3	1	AVX
	y,y,y		-				١,, ١	3	=	
VMAXPS/D VMINPS/D	y,y,m256	1	1		1		1+		1	AVX
ROUNDSS/SD/PS/PD	x,x,i	1	1		1			3	1	SSE4.1
ROUNDSS/SD/PS/PD	x,m128,i	2	1		1		1		1	SSE4.1
VROUNDSS/SD/PS/PD	y,y,i	1	1		1			3	1	AVX
VROUNDSS/SD/PS/PD	y,m256,i	2	1		1		1+		1	AVX
DPPS	x,x,i	4	4	1	2	1		12	2	SSE4.1
DPPS	x,m128,i	6	5				1		4	SSE4.1
VDPPS	y,y,y,i	4	4	1	2	1		12	2	AVX
VDPPS	y,m256,i	6	5				1+		4	AVX
DPPD	x,x,i	3	3	1	1	1	-	9	2	SSE4.1
DPPD	x,m128,i	4	3	'	ļ .		1	0	2	SSE4.1
B1 1 B	X,111120,1		0				'		_	00L4.1
Math										
SQRTSS/PS	x,x	1	1	1				10-14	10-14	
SQRTSS/PS			1	1			1	10-14	10-14	
	x,m128	1		'			'			A \ /\/
VSQRTPS	у,у	3	3						21-28	AVX
VSQRTPS	y,m256	4	3				1+	40.04	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		1	
VRSQRTPS	y,y	3	3					7	2	AVX
VRSQRTPS	y,m256	4	3				1+		2	AVX
	<b>y</b> ,									
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/	۸,111120	'	'			'			ı	
PD	V V V	1	1			1		1	1	AVX
	y,y,y	'	l					ı	I	ΑVΛ
VAND/ANDN/OR/XORPS/ PD	V. V 0.F.C	4	4			4	4.		4	A \ /\/
F D	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
								2	-	AVX,
VZEROALL		12							11	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

register, (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.

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  $\mu$ op fusion if the sum of the numbers listed under p015 + p23 + p4 exceeds the number listed under  $\mu$ 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  $\mu$ op in the second clock cycle, but a read port can receive an address-calculation  $\mu$ op in the second clock cycle. An x under p0, p1 or p5 means that at least one of the  $\mu$ 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  $\mu$ 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  $\mu$ 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	р1	р5	p23	p4		through- put	

Move instructions											
MOV	r,i	1	1	Х	Х	Х			1	0.33	
MOV	r8/16,r8/16	1	1	Х	Х	Х			1	0.33	
MOV	r32/64,r32/64	1	1	Х	Х	Х			0-1	0.25	may be elimin.
MOV	r8/16,m8/16	1	1	X	х	х	1		2	0.5	Cilitiiii.
MOV	r32/64,m32/64	1					1		2	0.5	
MOV	r,m	1					1		2	1	64 b abs
MOV	m,r	1					1	1	3	1	address
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	x	Х	х			1	0.33	
MOVZX	r32/64,r8	1	1	X	X	х			0-1	0.25	may be
MOVZX	r32/64,r16	1	1	x	х	х			1	0.33	elimin.
MOVSX MOVZX	r16,m8	2	1	X	X	X	1		3	0.5	
MOVSX MOVZX MOVSXD	r32/64,m	1	•		^		1		2	0.5	
CMOVcc	r,r	2	2	X	х	Х			2	0.67	
CMOVcc	r,m	2	2	X	X	X	1			~0.8	
XCHG	r,r	3	3	x	x	X	'		2	1	
XCHG		3 7	X	^	^	^	2	3	25	I	implicit
ACTIO	r,m	,	^					3	25		lock
XLAT		3	2				1		7	1	
PUSH	r	1					1	1	3	1	
PUSH	i	1					1	1		1	
PUSH	m	2					2	1		1	
PUSH	(E/R)SP	2	1	Х	Х	Х	1	1	3	1	
PUSHF(D/Q)		3	2	Х	Х	Х	1	1		1	
PUSHA(D)		19	3	Х	Х	Х	8	8		8	not 64 bit
POP	r	1					1		2	0.5	
POP	(E/R)SP	3	2	Х	Х	Х	1			0.5	
POP	m	2					2	1		1	
POPF(D/Q)		9	8	х	Х	Х	1			18	
POPA(D)		18	10	х	Х	Х	8			9	not 64 bit
LAHF SAHF		1	1	х		х			1	1	
SALC		3	3	х	Х	х			1	1	not 64 bit
LEA	r16,m	2	2	Х	1	х			2-4	1	
LEA	r32/64,m	1	1	x	х				1	0.5	1-2 com-
	*22/64 ***	1	4		4				3		ponents
LEA	r32/64,m	ı	1		1				3	1	3 com- ponents
BSWAP	r32	1	1		1				1	1	or RIP
BSWAP	r64	2	2	V	1	v			2	1 1	
				X	ı	Х	4				
PREFETCHNTA	m 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
Arithmetic instructions											
ADD SUB	r,r/i	1	1	X	Х	Х			1	0.33	
ADD SUB	r,m	1	1	X	Х	Х	1			0.5	
ADD SUB	m,r/i	2	1	X	Х	Х	2	1	6	1	
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	Х	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	Х	x	2	1	6	1	
AAA AAS		2	2	X	1	Х		-	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	
MUL IMUL	r16	4	4						4	2	
MUL IMUL	r32	3	3						4	2	
MUL IMUL	r64	2	2						3	1	
IMUL	r,r	1	1		1				3	1	
IMUL	r16,r16,i	2	2		•				4	1	
IMUL	r32,r32,i	1	1		1				3	1	
IMUL	r64,r64,i	1	1		1				3	1	
MUL IMUL	m8	1	1		1		1		3	1	
MUL IMUL	m16	4	3				1			2	
MUL IMUL	m32	3	2				1			2	
MUL IMUL	m64	2	1				1			2	
IMUL	r,m	1	1		1		1			1	
IMUL	r16,m16,i	2	2				1			1	
IMUL	r32,m32,i	1	1		1		1			1	
IMUL	r64,m64,i	1	1		1		1			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	Χ						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	X		х	_	'	1	1	
ROR ROL	m,cl	5	3	^		^	2	1		4	
RCL RCR	r,1	3	3	x	х	х	_	'	2	2	
RCL RCR	r,i	8	8	x	X	X			5	5	
RCL RCR	m,i	11	8	x	X	X	2	1	3	6	
RCL RCR	r,cl	8	8	x	X	X	_	'	5	5	
RCL RCR		11	8				2	1	5	6	
SHRD SHLD	m,cl r,r,i	1	1	X	Х	X		'	1	0.5	
SHRD SHLD		3	3		v	X	2	1	'	2	
SHRD SHLD	m,r,i	4	4	X	1	X		'	2	2	
SHRD SHLD	r,r,cl		4	X		X	2	4		4	
	m,r,cl	5		X	1	X	2	1	4		
BT	r,r/i	1	1	X		X	4		1	0.5	
BT	m,r	10	9	X	Х	X	1			5	
BT DTO DTO	m,i	2	1	Х		Х	1		_	0.5	
BTR BTS BTC	r,r/i	1	1	Х		Х			1	0.5	
BTR BTS BTC	m,r	11	8	X	Х	Х	2	1		5	
BTR BTS BTC	m,i	3	2	X		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 instruction									_	_	
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
Fused arithmetic and		1	1			1			0	1-2	jump fast if no
branch		'	'			'				1-2	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	m	3	1			1	2	1		2	
RET		2	1			1	1	•		2	
RET	i	3	2	x	х	1	1			2	
1	l .		1	1 -		1	1	1	i .	I .	1

BOUND INTO	r,m	15 4	13 4	X	х	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
REP MOVS		4/100							1/100		best case
SCAS		3	2	х	х	х	1			1	
REP SCAS		~6n							~2n		
CMPS		5	3	х	Х	Х	2			4	
REP CMPS		~8n							~2n		
Synchronization instruct	ions										
XADD	m,r	4	3	Х	Х	Х	1	1	7		
LOCK XADD	m,r	8	5	Х	Х	Х	2	1	22		
LOCK ADD	m,r	7	5	Х	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	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	μops	s unfused domain					_	procal	procal	Com- ments
		do- main	p015	p0	p1	р5	p23	p4		through- put		
Move instructions												

		•	, Dila	90							
FLD	r	1	1			1			1	1	
FLD	m32/64	1					1		3	1	
FLD	m80	4	2		1	1	2		5	2	
FBLD	m80	43	40		'	'	3		45	21	
							3				
FST(P)	r	1	1			1			1	1	
FST(P)	m32/m64	1					1	1	4	1	
FSTP	m80	7	3				2	2	5	5	
FBSTP	m80	243								252	
FXCH	r	1	0						0	0.5	
FILD	m	1	1		1		1		6	1	
FIST(P)	m	3	1		1		1	1	7	1	
FISTTP	m	3	1		1		1	1	7	2	SSE3
FLDZ		1	1			1					
FLD1		2	2		1	1					
FLDPI FLDL2E etc.		2	2	1	1	ľ				2	
FCMOVcc	r	3	3	1	'	2			2	2	
FNSTSW	AX	2	2	1	v				4	1	
				'	Х	Х		_	4		
FNSTSW	m16	2	1			_	1	1		1	
FLDCW	m16	3	2			2	1			3	
FNSTCW	m16	2	1			1	1	1		1	
FINCSTP FDECSTP		1	1			1			1	1	
FFREE(P)	r	1	1			1				1	
FNSAVE	m	143								167	
FRSTOR	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)		2	1	1			1		10-24	8-18	
	m			'		4	1		4		
FABS		1	1			1			1	1	
FCHS		1	1			1			1	1	
FCOM(P) FUCOM	r	1	1		1				3	1	
FCOM(P) FUCOM	m	1	1		1		1			1	
FCOMPP FUCOMPP		2	2		1	1			4	1	
FCOMI(P) FUCOMI(P)	r	3	3	1	1	1			5	1	
FIADD FISUB(R)	m	2	2		2		1			2	
FIMUL	m	2	2	1	1		1			2	
FIDIV(R)	m	2	2	1	1		1				
FICOM(P)	m	2	2		2		1			2	
FTST		1	1		1					1	
FXAM		2	2		2					2	
FPREM		28	28						21-26	12	
FPREM1		41							27-50	19	
FRNDINT		17	17						22	11	
Math											
FSCALE		25	25	x	х	х			49	49	
FXTRACT		17	17	X	X	X			10	10	
FSQRT		1	1	1		^			10-23	8-17	
FSIN		21-78	'	1	v	v			47-106	47-106	
I OIIN		21-10		X	Х	X	I		47-100	77-100	

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	Χ	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	Χ	1		1	
FNCLEX	5	5	Х	Χ	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	procal	Com- ments
		do- main	p015	p0	p1	<b>p5</b>	p23	p4		through- put	
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	Х			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		X	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	<b>'</b>	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	l	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	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
· ·								-		-	. '

DOMDEOO	1 400			ı	ı	I	   a	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
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	1 .	х	х	.		1	0.5	
PAVGB/W	(x)mm,m	1	1		X	X	1		•	0.5	
PMIN/MAXSB	x,x	1	1		X	X	'		1	0.5	SSE4.1
PMIN/MAXSB	x,m128	1	1		X	X	1		!	0.5	SSE4.1
PMIN/MAXUB	·	1	1				'		1	0.5	33E4.1
_	(x)mm,(x)mm	1			X	X	,		I		
PMIN/MAXUB	(x)mm,m	•	1		Х	Х	1		4	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
5/ 2 / .	7.,,.	•		'		•	-			·	552
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	0.5	SSE4.1
PTEST	x,m128	3	2	1	X		1		ı	1	SSE4.1
	· ·				Х	Х	'		4		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											

PCMPESTRI	DOMDECTDI		_		_			1	1			00540
PCMPESTRM	PCMPESTRI	x,x,i	8	8	3	1	4			4	4	SSE4.2
PCMPESTRM		x,m128,i	_		_	1	3	1			4	
Name	PCMPESTRM	x,x,i	8	8	3	1	4			12	4	SSE4.2
PCMPISTRI	PCMPESTRM	x,m128,i	8	7	3	1	3	1			4	SSE4.2
PCMPISTRM PCMPISTRM         x,x,i x,m128,i         3 3 3 3 1 1         11 3 SSE4.2           Encryption instructions PCLMULQDQ PCLMULQDQ AESDEC, AESDECLAST, AESENC, AESENCLAST AESENC, AESENCLAST AESENC, AESENCLAST X,x         18 18 x x x x 1 1 8 14 8 CLMUL AESONC, AESENCLAST X,x         14 8 CLMUL AESONCLAST AESONC, AESENCLAST X,x         1 AESONC AESONC, A	PCMPISTRI	x,x,i	3	3	3					3		SSE4.2
Encryption instructions         x,m128,i         4         3         3         1         3         SSE4.2           Encryption instructions         PCLMULQDQ         x,x,i         18         18         x x x x x x x x x x x x x x x x x x x	PCMPISTRI	x,m128,i	4	3	3			1			3	SSE4.2
Encryption instructions	PCMPISTRM	x,x,i	3	3	3					11		SSE4.2
PCLMULQDQ         x,x,i         18         18         x x x         14         8         CLMUL           PCLMULQDQ         x,m,i         18         17         x x x x         1         4         8         CLMUL           AESDEC, AESDECLAST, AESENC, AESENCLAST         x,x         2         2 x x 1         4         1         AES           AESIMC         x,x         2         2 x x 1         1         1         AES           AESIMC         x,m         3         2 x x 1         1         1         AES           AESKEYGENASSIST         x,x,i         11         11 x x x x         1         10         8         AES           Other         Other         AES         AES         AES         AES         AES         AES	PCMPISTRM	x,m128,i	4	3	3			1			3	SSE4.2
PCLMULQDQ         x,x,i         18         18         x x x         14         8         CLMUL           PCLMULQDQ         x,m,i         18         17         x x x x         1         4         8         CLMUL           AESDEC, AESDECLAST, AESENC, AESENCLAST         x,x         2         2 x x 1         4         1         AES           AESIMC         x,x         2         2 x x 1         1         1         AES           AESIMC         x,m         3         2 x x 1         1         1         AES           AESKEYGENASSIST         x,x,i         11         11 x x x x         1         10         8         AES           Other         Other         AES         AES         AES         AES         AES         AES												
PCLMULQDQ         x,m,i         18         17         x         x         1         8         CLMUL           AESDEC, AESDECLAST, AESENC, AESENCLAST         x,x         2         2         x         x         1         4         1         AES           AESIMC         x,x         2         2         x         x         1         1         AES           AESIMC         x,m         3         2         x         x         1         2         AES           AESKEYGENASSIST         x,x,i         11         11         x         x         x         1         7         AES           Other         Other         0         <												
AESDEC, AESDECLAST, AESENC, AESENCLAST  AESDEC, AESDECLAST, AESENC, AESENCLAST  X,X  2  2  X,X  4  1  AES  AESIMC  AESIMC  AESIMC  AESIMC  AESIMC  AESIMC  AESKEYGENASSIST  AESKEYGENASSIST  AESKEYGENASSIST  X,m,i  11  11  11  11  12  14  24  11  14  24  2	PCLMULQDQ	x,x,i	18	18	Х	Х	Х			14	8	CLMUL
AESENC, AESENCLAST  AESDEC, AESDECLAST, AESENC, AESENCLAST  X,X  2  2  2  X  X  4  1  AES  AESIMC  AESIMC  AESIMC  AESIMC  AESIMC  AESKEYGENASSIST  AESKEYGENASSIST  AESKEYGENASSIST  AESKEYGENASSIST  ACS  Other	PCLMULQDQ	x,m,i	18	17	Х	Х	Х	1			8	CLMUL
AESDEC, AESDECLAST, AESENC, AESENCLAST  x,m 3 2 x 1 1 1 1 AES  AESIMC  AESIMC  AESKEYGENASSIST  AESKEYGENASSIST  AESKEYGENASSIST  AESKEYGENASSIST  ACSTALL  x,x 2 2 2 2 1 1 2 AES  x,x,i 11 11 x x x x 1 10 8 AES  AESKEYGENASSIST  x,m,i 11 10 x x x 1 1 7 AES  Other	AESDEC, AESDECLAST,											
AESDEC, AESDECLAST, AESENC, AESENCLAST  x,m 3 2 x x 1 1 1 1 AES  AESIMC  AESIMC  AESKEYGENASSIST  AESKEYGENASSIST  AESKEYGENASSIST  X,m,i 11 11 x x x x 1 10 8 AES  AESKEYGENASSIST  X,m,i 11 10 x x x 1 1 7 AES  Other	AESENC, AESENCLAST											
AESENC, AESENCLAST  x,m 3 2 x x 1 1 1  AES  AESIMC  AESIMC  AESIMC  X,m 3 2 2 2 1 14 2 AES  AESKEYGENASSIST  AESKEYGENASSIST  X,m,i 11 11 x x x x 1 10 8 AES  AESKEYGENASSIST  X,m,i 11 10 x x x 1 7 AES  Other		x,x	2	2	х	Х	1			4	1	AES
AESIMC       x,x       2       2       2       1       1       AES         AESIMC       x,m       3       2       2       1       14       2       AES         AESKEYGENASSIST       x,x,i       11       11       x       x       x       10       8       AES         AESKEYGENASSIST       x,m,i       11       10       x       x       x       1       7       AES             Other	AESDEC, AESDECLAST,											
AESIMC       x,x       2       2       2       14       2       AES         AESIMC       x,m       3       2       2       1       2       AES         AESKEYGENASSIST       x,x,i       11       11       x       x       x       x       1       10       8       AES         AESKEYGENASSIST       x,m,i       11       10       x       x       x       1       7       AES             Other	AESENC, AESENCLAST											
AESIMC         x,m         3         2         2         1         2         AES           AESKEYGENASSIST         x,x,i         11         11         x         x         x         x         1         10         8         AES           AESKEYGENASSIST         x,m,i         11         10         x         x         x         1         7         AES		x,m	3	2	Х	Х	1	1			1	AES
AESKEYGENASSIST         x,x,i         11         11         x x x         11         10         8         AES           AESKEYGENASSIST         x,m,i         11         10         x x x         1         7         AES	AESIMC	x,x	2	2			2			14	2	AES
AESKEYGENASSIST         x,m,i         11         10         x         x         1         7         AES           Other	AESIMC	x,m	3	2			2	1			2	AES
Other	AESKEYGENASSIST	x,x,i	11	11	Х	Х	Х			10	8	AES
	AESKEYGENASSIST	x,m,i	11	10	x	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	Com- ments
		do- main	p015	p0	p1	р5	p23	<b>p4</b>		through- put	
Move instructions											
MOVAPS/D	X,X	1	1			1			0-1	≤1	elimin.
VMOVAPS/D	y,y	1	1			1			0-1	≤1	elimin.
MOVAPS/D MOVUPS/D VMOVAPS/D	x,m128	1					1		3	0.5	
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	

INSERTPS				,								
VSHUEPS/D VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X/I VPERMILPS/PD VX.X,I VPERMILPS/PD VX.X,I VPERMILPS/PD VY.M,I VPERMILPS/PD VX.X VP		x,m128,i	2	1			1	1			1	
VPERMILPS/PD  VPURMILPS/PD  VP	VSHUFPS/D	y,y,y,i	1	1			1			1	1	AVX
VPERMILPS/PD         Y,y,y/i         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         AVX           VPERMILPS/PD         X,x,mi         2         1         1         1         1         1         AVX           VPERMILPS/PD         X,mi         2         1         1         1         1         AVX           VPERMILPS/PD         X,mi         2         1         1         1         1         AVX           VPERMILPS/PD         Y,y,mi         2         1         1         1         1         AVX           VPERMILPS/PD         Y,y,mi         2         1         1         1         1         AVX           VPERMILPS/PD         X,xi         1         1         1         1         4         AVX           VPERMILPS/PD         X,xi         1         1         1         1         AVX           VPERMILPS/PD         X,xi         1         1         1         1         AVX           VBLENDS/PS/PD         X,xi         X,xi         1         1         1         AVX	VSHUFPS/D	y, y,m256,i	2	1			1	1+			1	AVX
VPERMILPS/PD         X.X.m         2         1         1         1         1         AVX           VPERMILPS/PD         Y.y.m.i         2         1         1         1         1         AVX           VPERMILPS/PD         y.m.i         2         1         1         1         1         AVX           VPERMILPS/PD         y.m.i         2         1         1         1         1         AVX           VPERMILPS/PD         y.m.i         2         1         1         1         1         AVX           VPERMILPS/PD         y.y.m.i         2         1         1         1         1         AVX           VPERMILPS/PD         x.m.128,1         2         1         x         x         1         0.5         SSE4.1           VBLENDPS/PD         y.m.256,1         2         1         x         x         1         1         AVX           VBLENDVPS/PD         y.y.m.xmm0         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y.y.m.y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y.y.m.y	VPERMILPS/PD	x,x,x/i	1	1			1			1	1	AVX
VPERMILPS/PD	VPERMILPS/PD	y,y,y/i	1	1			1			1	1	AVX
VPERMILPS/PD         y,y,m         2         1         1         1         1         AVX           VPERMILPS/PD         y,m,i         2         1         1         1         1         AVX           VPERMILPS/PD         y,m,i         2         1         1         1         1         1         AVX           VPERMZF128         y,y,m,i         2         1         1         1         1         AVX           VPERMZF128         y,y,m,i         2         1         1         1         1         AVX           VPERMZF128         y,y,m,i         2         1         1         1         1         AVX           BLENDPS/PD         x,mid2,i         2         1         x         x         1         0.5         SSE4.1           BLENDPS/PD         y,m256,i         2         1         x         x         1+         1         AVX           VBLENDVPS/PD         y,y,y,y         2         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,	VPERMILPS/PD	1	2	1			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           VPERMIZP128         y,y,m,i         2         1         X         X         1         0.5         SSE4.1           BLENDPS/PD         y,m,256,i         2         1         X         X         1         0.5         SSE4.1           BLENDVPS/PD         y,m,20         2         2         X         X         1         AVX           VBLENDVPS/PD         y,y,m,y         2         2         X         X         1         AVX           VBLENDVPS/PD         y,m,20         2         X         X         1         1         AVX           VBLENDVPS/PD         y,y,m,4         2         1	VPERMILPS/PD		2	1			1	1+			1	AVX
VPERMILPS/PD         y,m,i         2         1         1         1         1         1         AVX           VPERMZF128         y,y,m,i         2         1         1         1         1         1         AVX           VPERMZF128         y,y,m,i         2         1         1         1         1         1         AVX           BLENDPS/PD         X,x,i         1         1         1         X         1         0.5         SSE4.1           VBLENDPS/PD         X,mid28,i         2         1         X         X         1         0.5         SSE4.1           VBLENDVPS/PD         y,m256,i         2         1         X         X         1         1         AVX           VBLENDVPS/PD         y,m256,i         2         1         X         X         1         1         AVX           VBLENDVPS/PD         y,y,y,y         2         2         X         X         1         1         AVX           VBLENDVPS/PD         y,y,y,y         3         2         X         X         1         1         AVX           VBLENDVPS/PD         y,y,m3         3         2         X         X         1				1			1	1			1	
VPERMZF128         Y.y.y.i         1         0.5         SSE4.1         1         0.5         SSE4.1         1         0.5         SSE4.1         1         1         0.5         SSE4.1         1         1         3         SSE4.1         1         1         3         SSE4.1         1         4         VX         1         1         3         SSE4.1         1         1         3         SSE4.	VPERMILPS/PD			1			1	1+			1	
VPERMZF128	_	-					1			2		
BLENDPS/PD								1+		_		
BLENDPS/PD		1			×		×	-		1		
VBLENDPS/PD         y,y,i         1         1         x         x         1         1         0.5         AVX           VBLENDPS/PD         y,m256,i         2         1         x         x         1         1         AVX           BLENDVPS/PD         y,x,mm0         2         2         x         x         1         1         SSE4.1           VBLENDVPS/PD         y,y,y,y         2         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           MOVDDUP         x,m64         1         1         1         1         1         1         1         1         1         1         AVX         VBROADCASTSS         y,m32         1								1		•		
VBLENDPS/PD         y,m256,i         2         1         x         x         1+         2         1         AVX           BLENDVPS/PD         x,x,xmm0         2         2         x         x         1         SSE4.1           VBLENDVPS/PD         y,y,y,y,y         2         2         x         x         1         1         SSE4.1           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLENDVPS/PD         y,y,m,y         3         2         x         x         1         1         AVX           VBLODDUP         x,m64         1         1         1         1         1         AVX           VBROADCASTSS         y,m64         2         1         1         1         1         AVX				_						1		
BLENDVPS/PD BLENDVPS/PD BLENDVPS/PD X,m,xmm0 3 2 x x 1 1 1 SSE4.1 SLENDVPS/PD X,m,xmm0 3 2 x x 1 1 1 SSE4.1				_				1+		•		
BLENDVPS/PD		·		1 -				' '		2		
VBLENDVPS/PD Vy.y.m.y VBLENDVPS/PD Vy.y.m.y VBLENDVPS/PD Vy.y.m.y VBLENDVPS/PD Vy.y.m.y VBLENDVPS/PD VB.Y.M.Y VBLENDVPS/PD VB.Y.M.Y VB.Y.Y VBROADCASTSS V.M.Y VBROADCASTSS V.M.Y VBROADCASTSD V.M.Y VBROADCASTSS V.M.Y VALY VALY VBROADCASTSS V.M.Y VALY VALY VALY VBROADCASTSS V.M.Y VALY VALY VALY VALY VALY VALY VALY VAL								1		2		
VBLENDVPS/PD         y,y,m,y         3         2         x         1+         1         AVX           MOVDDUP         x,x         1         1         1         1         1         1         SSE3           VMOVDDUP         y,m64         1         1         1         1         1         1         AVX           VMOVDDUP         y,m256         1         1         1         1         1         AVX           VBROADCASTSS         y,m32         2         1         1         1         4         0.5         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1								'		2	1	
MOVDDUP         X,X         1         1         1         1         1         3         0.5         SSE3           MOVDDUP         y,M64         1         1         1         1         1         1         1         1         AVX           VMOVDDUP         y,m256         1         1+         3         1         AVX           VBROADCASTSS         y,m32         2         1         1         4         0.5         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1		1						1_		2		
MOVDDUP         x,m64         1         d         1         1         1         3         0.5         SSE3           VMOVDDUP         y,y         1         1         1         1         1         1         AVX           VBROADCASTSS         x,m32         1         1         4         0.5         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1		''	_		<b>*</b>			17		4	•	
VMOVDDUP         y,y         1         1         1         1         1         1         1         AVX           VMOVDDUP         y,m256         1         1         1         1         1         4         0.5         AVX           VBROADCASTSS         y,m32         2         1         1         1         5         1         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         1         1         SSE3           VBROADCASTF128         y,m128         2         1         1         1         1         1         SSE3           VMOVSH/LDUP         x,m128         1         1         1         1         1         1         1         1         AVX           UNPCKH/LPS/D         x,m2         1         1		· ·	-	'			'	4			-	
VMOVDDUP         y,m256         1         1         1         1         4         0.5         AVX           VBROADCASTSS         y,m32         2         1         1         1         4         0.5         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTSD         y,m128         1         1         1         1         1         1         AVX           VBROADCASTSP         y,m128         1         1         1         1         1         1         AVX           WONSH/LDUP         x,m128         1         1         1         1         4         AVX           UNPCKH/LPS/D         x,m128         1         1         1         1         1         3		· ·		4			_	ı				
VBROADCASTSS         x,m32         1         1         1         4         0.5         AVX           VBROADCASTSS         y,m32         2         1         1         1         5         1         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           MOVSH/LDUP         x,x         1         1         1         1         1         1         1         1         1         1         1         1         1         AVX           VMOVSH/LDUP         y,m256         1         1         1         1         1         1         AVX         AVX           VMOVSH/LDUP         y,m256         1         1         1         1         1         1         AVX			-	1			1	۱.		-		
VBROADCASTSS         y,m32         2         1         1         1         5         1         AVX           VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         1         1         1         1         AVX           VBROADCASTF128         y,m128         2         1		l • • ·										
VBROADCASTSD         y,m64         2         1         1         1         5         1         AVX           VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           MOVSH/LDUP         x,x         1         1         1         1         1         1         1         SSE3           MOVSH/LDUP         y,m256         1         1         1         1         1         AVX           VMOVSH/LDUP         y,m256         1         1         1         1         AVX           UNPCKH/LPS/D         x,m128         1         1         1         1         1         3         SSE3           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         4         AVX           EXTRACTPS         m32,xi         2         2         x								-				
VBROADCASTF128         y,m128         2         1         1         1         5         1         AVX           MOVSH/LDUP         x,x         1         <		1 -					-					
MOVSH/LDUP         x,x         1         1         1         1         1         1         3         0.5         SSE3           MOVSH/LDUP         y,y         1         1         1         1         1         1         1         1         1         AVX           VMOVSH/LDUP         y,y         1         1         1         1         1         1         1         AVX           VMOVSH/LDUP         y,y         1         1         1         1         1         1         AVX           VMOVSH/LDUP         y,y         1         1         1         1         1         1         AVX           UNPCKH/LPS/D         x,x         1         1         1         1         1         SSE3           VUNPCKH/LPS/D         y,y,m         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m         25         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m         25         1         1         1         1         1         1         AVX           EXTRACTPS         m32,x,i <td></td> <td>J .</td> <td></td>		J .										
MOVSH/LDUP         x,m128         1         1         1         3         0.5         SSE3           VMOVSH/LDUP         y,y         1         1         1         1         1         1         AVX           VMOVSH/LDUP         y,m256         1         1         1         1         1         1         AVX           UNPCKH/LPS/D         x,m128         1         1         1         1         1         1         SSE3           UNPCKH/LPS/D         y,y,m256         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         AVX           EXTRACTPS         m32,xi         3         2         x         x         1         1         AVX           VEXTRACTF128         m128,yi				_			-	1				
VMOVSH/LDUP         y,y         1         1         1         1         1         1         1         1         AVX           VMOVSH/LDUP         y,m256         1         1         1         1         1         1         AVX           UNPCKH/LPS/D         x,m128         1         1         1         1         1         1         1         1         SSE3           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         AVX           VWEXTRACTPS         r32,xi         2         2         x         x         1         1         1         1         AVX           VEXTRACTF128         m128,yi         2		· ·		1			1					
VMOVSH/LDUP         y,m256         1         1         1+         1         AVX           UNPCKH/LPS/D         x,x         1         1         1         1         1         1         1         1         1         SSE3           UNPCKH/LPS/D         x,m128         1         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         AVX           VEXTRACTPS         r32,x,i         2         2         x         x         1         1         AVX           VEXTRACTF128         m128,y,i         2         0         1         1         4         1         AVX           VEXTRACTF128         m128,y,i         2         0         1         1         4		· ·	-					1				
UNPCKH/LPS/D UNPCKH/LPS/D UNPCKH/LPS/D VUNPCKH/LPS/D VINPCKH/LPS/D VINPC				1			1			1	-	
UNPCKH/LPS/D		y,m256	-					1+			-	
VUNPCKH/LPS/D         y,y,y         1         1         1         1         1         1         AVX           VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         1         AVX           EXTRACTPS         r32,x,i         2         2         x         x         2         1         SSE4.1           EXTRACTPS         m32,x,i         3         2         x         x         1         1         1         1         SSE4.1           VEXTRACTF128         m32,x,i         3         2         x         x         1         1         4         1         AVX           VEXTRACTF128         m128,y,i         2         0         1         1         4         1         AVX           VEXTRACTF128         m128,y,i         2         0         1         1         4         1         AVX           VINSERTPS         x,m32,i         2         1         1         1         1         1         3         SSE4.1           VINSERTF128         y,y,x,i         1         1         1         1         4         1         AVX           VMASKMOVPS/D         y,y,m256		· ·	1				1			1	1	
VUNPCKH/LPS/D         y,y,m256         1         1         1         1         1         1         4VX           EXTRACTPS         r32,x,i         2         2         x         x         2         1         SSE4.1           EXTRACTPS         m32,x,i         3         2         x         x         1         1         1         1         SSE4.1           VEXTRACTF128         m128,y,i         1         1         1         1         4         1         AVX           VEXTRACTF128         m128,y,i         2         0         1         1         4         1         AVX           VEXTRACTF128         m128,y,i         2         0         1         1         4         1         AVX           INSERTPS         x,x,i         1         1         1         1         1         1         SSE4.1           VINSERTF128         y,y,x,i         1         1         1         1         2         1         AVX           VMASKMOVPS/D         x,x,m128         2         2         1         x         x         1         4         1         AVX           VMASKMOVPS/D         m256,y,y         4		x,m128					1	1			1	
EXTRACTPS    F32,x,i   2   2   x   x			1	1			1			1	1	
EXTRACTPS  m32,x,i  x,y,i  m128,y,i  linsertps  x,x,i  linsertps  x,m32,i  x,m12,i  ,m12,i x,m12,i x,m12,i x,m12,i x,m12,i x,m12,i x,m12,i		y,y,m256		1 -			1	1+			1	
VEXTRACTF128         x,y,i         1         1         1         2         1         AVX           VEXTRACTF128         m128,y,i         2         0         1         1         4         1         AVX           INSERTPS         x,x,i         1         1         1         1         1         1         1         SSE4.1           INSERTPS         x,m32,i         2         1         1         1         1         1         SSE4.1           VINSERTF128         y,y,x,i         1         1         1         2         1         AVX           VINSERTF128         y,y,m128,i         2         1         x         x         1         4         1         AVX           VMASKMOVPS/D         x,x,m128         3         2         x         x         1         4         1         AVX           VMASKMOVPS/D         m128,x,x         4         2         x         x         1         1         1         AVX           VMASKMOVPS/D         m256,y,y         4         2         x         x         1         1         1         AVX           COTPD2PS         x,m128         2         2 <t< td=""><td></td><td>r32,x,i</td><td></td><td></td><td>Х</td><td></td><td>Х</td><td></td><td></td><td>2</td><td>1</td><td>SSE4.1</td></t<>		r32,x,i			Х		Х			2	1	SSE4.1
VEXTRACTF128         m128,y,i         2         0         1         1         4         1         AVX           INSERTPS         x,x,i         1         2         1         AVX         VXX         VXX         VXX         2         2         1         1         1         1         1         AVX         XXX         XXX         XXX         2         2         1         1         1         1         1         AVX         XXX         XXX         XXX	EXTRACTPS	m32,x,i	3	2	Х		Х	1	1		1	SSE4.1
INSERTPS		x,y,i					1				1	AVX
INSERTPS	VEXTRACTF128	m128,y,i	2	0				1	1	4	1	AVX
VINSERTF128         y,y,x,i         1         1         2         1         AVX           VINSERTF128         y,y,m128,i         2         1         x         x         1         4         1         AVX           VMASKMOVPS/D         x,x,m128         3         2         x         x         1         4         1         AVX           VMASKMOVPS/D         m128,x,x         4         2         x         x         1         1         1         AVX           VMASKMOVPS/D         m256,y,y         4         2         x         x         1         1+         2         AVX           Conversion         x,x         2         2         1         1         1         4         1           CVTPD2PS         x,m128         2         2         1         1         1         4         1           VCVTPD2PS         x,y         2         2         1         1         1         AVX	INSERTPS	x,x,i	1	1			1			1	1	SSE4.1
VINSERTF128         y,y,m128,i         2         1         x         x         1         4         1         AVX           VMASKMOVPS/D         x,x,m128         3         2         x         x         1         4         1         AVX           VMASKMOVPS/D         y,y,m256         3         2         1+         5         1         AVX           VMASKMOVPS/D         m128,x,x         4         2         x         x         1         1         AVX           VMASKMOVPS/D         m256,y,y         4         2         x         x         1         1+         2         AVX           Conversion         x,x         2         2         1         1         4         1         AVX           CVTPD2PS         x,m128         2         2         1         1         1         4         1         AVX           VCVTPD2PS         x,y         2         2         1         1         1         AVX         1         1         AVX         1         1         AVX         1         1         1         AVX         1         1         1         1         1         1         1         1	INSERTPS	x,m32,i	2	1			1	1			1	SSE4.1
VMASKMOVPS/D         x,x,m128         3         2         x         1         4         1         AVX           VMASKMOVPS/D         y,y,m256         3         2         x         1+         5         1         AVX           VMASKMOVPS/D         m128,x,x         4         2         x         x         1         1         AVX           VMASKMOVPS/D         m256,y,y         4         2         x         x         1         1+         2         AVX           Conversion         CVTPD2PS         x,x         2         2         1         1         4         1           CVTPD2PS         x,m128         2         2         1         1         1         4         1           VCVTPD2PS         x,y         2         2         1         1         4         1         AVX	VINSERTF128	y,y,x,i		1			1			2	1	AVX
VMASKMOVPS/D         y,y,m256         3         2         1+         5         1         AVX           VMASKMOVPS/D         m128,x,x         4         2         x         x         1         1         AVX           VMASKMOVPS/D         m256,y,y         4         2         x         x         1         1+         2         AVX           Conversion         x,x         2         2         1         1         4         1           CVTPD2PS         x,m128         2         2         1         1         4         1           VCVTPD2PS         x,y         2         2         1         1         4         1         AVX	VINSERTF128	y,y,m128,i	2	1	X		Х	1		4	1	AVX
VMASKMOVPS/D       m128,x,x m256,y,y       4       2       x       x       1       1       1       AVX         Conversion       x,x 2       2       2       1       1       4       1         CVTPD2PS       x,m128       2       2       1       1       4       1         VCVTPD2PS       x,y       2       2       1       1       4       1         VCVTPD2PS       x,y       2       2       1       1       4       1       AVX	VMASKMOVPS/D	x,x,m128	3	2	Х		Х	1		4	1	AVX
VMASKMOVPS/D       m128,x,x m256,y,y       4       2       x       x       1       1       1       AVX         Conversion       x,x 2       2       2       1       1       4       1         CVTPD2PS       x,m128       2       2       1       1       4       1         VCVTPD2PS       x,y       2       2       1       1       4       1         VCVTPD2PS       x,y       2       2       1       1       4       1       AVX	VMASKMOVPS/D	y,y,m256	3	2				1+		5	1	AVX
VMASKMOVPS/D         m256,y,y         4         2         x         x         1         1+         2         AVX           Conversion         CVTPD2PS         x,x         2         2         1         1         4         1           CVTPD2PS         x,m128         2         2         1         1         1         1           VCVTPD2PS         x,y         2         2         1         1         4         1         AVX	VMASKMOVPS/D	• •	4	2	X	Х		1	1		1	AVX
CVTPD2PS         x,x         2         2         1         1         4         1           CVTPD2PS         x,m128         2         2         1         1         1         1           VCVTPD2PS         x,y         2         2         1         1         4         1         AVX	VMASKMOVPS/D	m256,y,y	4	2	X	х		1	1+		2	AVX
CVTPD2PS         x,x         2         2         1         1         4         1           CVTPD2PS         x,m128         2         2         1         1         1         1           VCVTPD2PS         x,y         2         2         1         1         4         1         AVX	Conversion											
VCVTPD2PS         x,y         2         2         1         1         4         1         AVX	CVTPD2PS	X,X	2	2		1	1			4	1	
VCVTPD2PS         x,y         2         2         1         1         4         1         AVX	CVTPD2PS	· ·		2		1	1	1			1	
	VCVTPD2PS		2	2		1	1			4	1	AVX
	VCVTPD2PS		2	2		1	1	1+			1	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		7	1	AVX
1	1			1		1	'		2	1	AVA
CVTSS2SD	X,X	2	2	1		1			2	-	
CVTSS2SD	x,m32	2	1	1			1		_	1	
CVTDQ2PS	X,X	1	1		1				3	1	
CVTDQ2PS	x,m128	1	1		1		1			1	
VCVTDQ2PS	y,y	1	1		1				3	1	AVX
VCVTDQ2PS	y,m256	1	1		1		1+			1	AVX
CVT(T) PS2DQ	X,X	1	1		1				3	1	
CVT(T) PS2DQ	x,m128	1	1		1		1			1	
VCVT(T) PS2DQ	y,y	1	1		1				3	1	AVX
VCVT(T) PS2DQ	y,m256	1	1		1		1+			1	AVX
CVTDQ2PD	x,x	2	2		1	1			4	1	/ ( ) / (
CVTDQ2PD	x,m64	2	2		1	1	1		_	1	
VCVTDQ2PD	1	2	2		1	1	'		5	1	AVX
VCVTDQ2PD VCVTDQ2PD	y,x	2	2			1	4		S	1	AVX
1	y,m128				1	-	1				AVX
CVT(T)PD2DQ	X,X	2	2		1	1			4	1	
CVT(T)PD2DQ	x,m128	2	2		1	1	1			1	
VCVT(T)PD2DQ	x,y	2	2		1	1			5	1	AVX
VCVT(T)PD2DQ	x,m256	2	2		1	1	1+			1	AVX
CVTPI2PS	x,mm	1	1		1				4		
CVTPI2PS	x,m64	1	1		1		1			3	
CVT(T)PS2PI	mm,x	2	2		1	1			4	1	
CVT(T)PS2PI	mm,m128	2	1		1		1			1	
CVTPI2PD	x,mm	2	2		1	1			4	1	
CVTPI2PD	x,m64	2	2		1	1	1		-	1	
CVT(T) PD2PI	mm,x	2	2		1	1			4	1	
CVT(T) PD2PI	mm,m128	2	2		1	1	1		_	1	
CVT(1)1 B211	x,r32	2	2		1	1	'		4	3	
CVTSI2SS	· ·	1	1		1		4		7	3	
I	x,m32			,			1		4		
CVT(T)SS2SI	r32,x	2	2	1	1				4	1	
CVT(T)SS2SI	r32,m32	2	2	1	1		1			1	
CVTSI2SD	x,r32	2	2		1	1			4	3	
CVTSI2SD	x,m32	2	1		1		1			3	
CVT(T)SD2SI	r32,x	2	2	1	1				4	1	
CVT(T)SD2SI	r32,m64	2	2	1	1		1			1	
VCVTPS2PH	x,v,i	3	3	1	1	1			10	1	F16C
VCVTPS2PH	m,v,i	3	2	1	1		1	1		1	F16C
VCVTPH2PS	V,X	2	2	1		1			6	1	F16C
VCVTPH2PS	v,m	2	1		1		1			1	F16C
Arithmetic											
ADDSS/D SUBSS/D	x,x	1	1		1				3	1	
ADDSS/D SUBSS/D	x,m32/64	1	1		1		1			1	
ADDOS/D SUBPS/D	X,11102704 X,X	1	1		1		'		3	1	
ADDPS/D SUBPS/D	x,m128	1	1		1		1			1	
VADDPS/D VSUBPS/D		1	1				'		3	1	A\/\
	y,y,y	-	1		1		۱.		ာ		AVX
VADDPS/D VSUBPS/D	y,y,m256	1	1		1		1+			1	AVX
ADDSUBPS/D	X,X	1	1		1				3	1	SSE3

	ı		<b>,</b>	J-	1	1		ı		
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	у,у,у	1	1	1				5	1	AVX
VMULPS	y,y,m256	1	1	1			1+		1	AVX
MULSD MULPD	x,x	1	1	1				5	1	
MULSD MULPD	x,m	1	1	1			1		1	
VMULPD	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			' '	10-13	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		١.	' +	10-20	8-14	AVA
DIVSD DIVPD	X,X	1	1	1			4	10-20	8-14	
	x,m		-			1	1	20.25		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	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
T. Control of the Con	'	1	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  $\mu$ ops at the decode, rename and allocate stages in the pipeline. Fused

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

MOV	m,i	1	2	p237 p4		1	
MOVNTI	m,r	2	2	p23 p4	~400	1	
MOVSX MOVZX	r,r	1	1	p0156	1	0.25	
MOVSXD							
MOVSX MOVZX	r16,m8	1	2	p23 p0156		0.5	
MOVSX MOVZX	r,m	1	1	p23		0.5	all other
MOVSXD							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	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			9	not 64 bit
LAHF SAHF		1	1	p06	1	1	
SALC		3	3	3p0156	1	1	not 64 bit
LEA	r16,m	2	2	p1 p0156	4	1	16 or 32 bit
	·						address size
LEA	r32/64,m	1	1	p15	1	0.5	1 or 2 compo-
							nents in
							address
LEA	r32/64,m	1	1	p1	3	1	3 components
							in address
LEA	r32/64,m	1	1	p1		1	rip relative
							address
BSWAP	r32	1	1	p15	1	0.5	
BSWAP	r64	2	2	p06 p15	2	1	
MOVBE	r16,m16	3	3	2p0156 p23		0.5	MOVBE
MOVBE	r32,m32	2	2	p15 p23		0.5	MOVBE
MOVBE	r64,m64	3	3	2p0156 p23		0.5	MOVBE
MOVBE	m16,r16	2	3	p06 p237 p4		1	MOVBE
MOVBE	m32,r32	2	3	p15 p237 p4		1	MOVBE
MOVBE	m64,r64	3	4	p06 p15 p237 p4		1	MOVBE
PREFETCHNTA/	m	1	1	p23		0.5	
0/1/2	111	'	'			0.0	
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	
			•				

			110				
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	20.05	1	AVX2
DIV	r8	9	9	p0 p1 p5 p6	22-25	9	
DIV DIV	r16 r32	11	11 10	p0 p1 p5 p6	23-26	9 9-11	
DIV	r64	10 36	36	p0 p1 p5 p6 p0 p1 p5 p6	22-29 32-96	21-74	
IDIV	r8	9	9	p0 p1 p5 p6	23-26	8	
IDIV	r16	10	10	p0 p1 p5 p6 p0 p1 p5 p6	23-26	8	
IDIV	r32	9	9	p0 p1 p5 p6 p0 p1 p5 p6	23-20	8-11	
IDIV	r64	59	59	p0 p1 p5 p6	39-103	24-81	
CBW	104	1	1	p0156	1	24-01	
CWDE		1 1	1	p0156	1		
CDQE				p0156	1		
CWD		2	2	p0156	1		
CDQ		1	1	p0130	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
1	1 .,		_	F   P=0	ı		· · <b>-</b>

				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	
BT	m,i	2	2	p06 p23		0.5	
BTR BTS BTC	r,r/i	1	1	p06	1	0.5	
BTR BTS BTC	m,r	10	11			5	
BTR BTS BTC	m,i	3	4	2p06 p23 p4		2	
BSF BSR	r,r	1	1	p1	3	1	
BSF BSR	r,m	1	2	p1 p23		1	
SETcc	r	1	1	p06	1	0.5	
SETcc	m	2	3	p06 p237 p4		1	
CLC		1	0	none		0.25	
STC		1	1	p0156		0.25	
CMC		1	1	p0156	1		
CLD STD		3	3	p15 p6		4	
LZCNT	r,r	1	1	p1	3	1	LZCNT
LZCNT	r,m	1	2	p1 p23		1	LZCNT
TZCNT	r,r	1	1	p1	3	1	BMI1
TZCNT	r,m	1	2	p1 p23		1	BMI1

	i					1	
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	2	p1 p23	· ·	1	BMI2
Control transfer i	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
, ,				·			taken
Conditional jump	short/near	1	1	p06		0.5-1	predicted not taken
Fused arithmetic and branch		1	1	р6		1-2	predicted taken
Fused arithmetic and branch		1	1	p06		0.5-1	predicted not 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 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			5	not 64 bit
String instruc-							
LODSB/W		3	3	2p0156 p23		1	
LODSD/Q		2	2	p0156 p23		1	
REP LODS		5n+12				~2n	
STOS		3	3	p23 p0156 p4		1	
REP STOS		<2n				~0.5n	worst case
REP STOS		2.6/32B				1/32B	best case aligned by 32
MOVS		5	5	2p23 p4 2p0156		4	
REP MOVS		~2n				~1.5 n	worst case
REP MOVS		4/32B				1/32B	best case aligned by 32
SCAS		3	3	p23 2p0156		1	ungried by 32
REP SCAS		≥6n		p20 2p0 100		≥2n	
11 00/10			l			-211	

CMPS		5	5	2p23 3p0156		4	
REP CMPS		≥8n				≥2n	
C. mahmani-ation	 						
Synchronization			_			_	
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	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	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	

ENCTOW	4.0				I		I
FNSTCW	m16	2	3	p237 p4 p6		1	
FINCSTP FDECS		1	1	p01	0	0.5	
FFREE(P)	r	1	1	p01		0.5	
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	_r .	19	13	
FPREM1		41	41		27	17	
FRNDINT		17	17		11	23	
Math							
FSCALE		25-75			49-125		
FXTRACT		17	17		15	11	
FSQRT		1	1	p0	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	p0100		83	
LIMINII				l .		00	

# Integer MMX and XMM instructions

Instruction	Operand	μορs fused	μορs unfused	uono cook vort	Lotonov	Recipro- cal through	Comments
Instruction	Operands	domain	domain	µops each port	Latency	put	Comments
Move instruc- tions							
MOVD	r32/64,(x)mm	1	1	p0	1	1	
MOVD	, ,	1	2			1	
MOVD	m32/64,(x)mm	1		p237 p4	3	1	
	(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	m128,x	1	2	p237 p4	~400	1 1	
VMOVNTDQ	m256,y	1	2	p237 p4	~400	1	AVX2
MOVNTDQA		1	1			0.5	SSE4.1
	x, m128		1	p23	3		
VMOVNTDQA PACKISWB/DW	y,m256	1	1	p23	3	0.5	AVX2
PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW PACKUSWB	mm,m64	3	3	p23 2p5		2	
PACKSSWB/DW PACKUSWB	x,x / y,y,y	1	1	p5	1	1	
PACKSSWB/DW							
PACKUSWB	x,m / y,y,m	1	2	p23 p5		1	
PACKUSDW	x,x / y,y,y	1	1	p5	1	1	SSE4.1
PACKUSDW	x,m / y,y,m	1	2	p23 p5		1	SSE4.1
PUNPCKH/L BW/WD/DQ	v,v / v,v,v	1	1	p5	1	1	
PUNPCKH/L BW/WD/DQ	v,m / v,v,m	1	2	p23 p5		1	
PUNPCKH/L							
QDQ	x,x / y,y,y	1	1	p5	1	1	
PUNPCKH/L							
QDQ	x,m / y,y,m	2	2	p23 p5		1	
PMOVSX/ZX BW BD BQ DW DQ	x,x	1	1	p5	1	1	SSE4.1
PMOVSX/ZX BW 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	1		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	4	1	
PSHUFD	V,V,İ	1	1	p5	1	1	
PSHUFD	v,m,i	2	2	p23 p5	4	1	
PSHUFL/HW	v,v,i	1	1	p5	1	1	
PSHUFL/HW	v,m,i	2	2	p23 p5	4	1	00050
PALIGNR	v,v,i / v,v,v,i	1	1	p5	1	1	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	0/40	0		04005	_	_	A) ()(0
B/W	x,m8/16	3	3	p01 p23 p5	5	1	AVX2
VPBROADCAST	x,m32/64	1	1	n23	4	0.5	AVX2
D/Q	A,11132/0 <del>4</del>	'	'	p23	4	0.5	7,472
VPBROADCAST B/W/D/Q	y,x	1	1	p5	3	1	AVX2
VPBROADCAST	J,**	•		""		•	
B/W	y,m8/16	3	3	p01 p23 p5	7	1	AVX2

I	I	I	I	I	I	I	l I
VPBROADCAST	v m22/64	4	1	22		0.5	AVX2
D/Q	y,m32/64	1 1	1	p23	5 3	0.5	AVX2 AVX2
VBROADCASTI128 VPGATHERDD	y,m128 x,[r+s*x],x	20	20	p23	3	9	AVX2 AVX2
VPGATHERDD		34	34			12	AVX2 AVX2
	y,[r+s*y],y						
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	p0	5	1	SSE4.2
PCMPGTQ	v,m / v,v,m	1	2	p0 p23		1	SSE4.2
PMULL/HW							
PMULHUW	v,v / v,v,v	1	1	p0	5	1	
PMULL/HW							
PMULHUW	v,m / v,v,m	1	2	p0 p23		1	
PMULHRSW	v,v / v,v,v	1	1	p0	5	1	SSSE3
PMULHRSW	v,m / v,v,m	1	2	p0 p23		1	SSSE3
PMULLD	x,x / y,y,y	2	2	2p0	10	2	SSE4.1
PMULLD	x,m / y,y,m	3	3	2p0 p23		2	SSE4.1
PMULDQ	x,x / y,y,y	1	1	p0	5	1	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	p15	1	0.5	000L3
PAVGB/W	v,w / v,v,w v,m / v,v,m	1	2	p15 p23	'	0.5	
	v,111 / v,V,III	'		p 15 p25		0.5	
PMIN/PMAX							
SB/SW/SD UB/UW/UD	V V / V V V	1	1	p15	1	0.5	SSE4.1
	x,x / y,y,y	'	'	μισ	'	0.5	JJL4.1
PMIN/PMAX SB/SW/SD							
UB/UW/UD	x,m / y,y,m	1	2	p15 p23		0.5	SSE4.1
00/01/100	^,111 / y,y,111	ı .		ρ 13 μ23	I	0.5	55L <del>4</del> . I

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 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	р0	1	1		
VPSLLVD/Q VPSRAVD		0		0.0.5			A) ()/O	
VPSRLVD/Q VPSLLVD/Q	V,V,V	3	3	2p0 p5	2	2	AVX2	
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-								
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	uctions						
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
	,	2	2		14	2	
AESIMC	X,X			2p5	14	_	AES
AESIMC	x,m	3	3	2p5 p23		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			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	'	l l	ρ23		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	, , , , , , , , , , , , , , , , , , ,	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,1110-4 y,x		1	p5	3	1	AVX2
VBROADCASTF128	y,^ 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	I I	1	SSE3
EXTRACTPS	r32,x,i	2	2	p0 p5		1	SSE4.1
EXTRACTPS	m32,x,i	3	3	p0 p5 p23	4	1	SSE4.1
VEXTRACTF128	x,y,i	1	1	p5 p5 p25	3	1	AVX
VEXTRACTF 128	m128,y,i	2	2	p23 p4	4	1	AVX
INSERTPS	x,x,i	1	1	p25 p4	1	1	SSE4.1
INSERTPS	x,m32,i	2	2	p23 p5	4	1	SSE4.1
VINSERTF128	y,y,x,i	1	1	p25 p5	3	1	AVX
VINSERTF128	y,y,∧,ı y,y,m128,i	2	2	p015 p23	4	2	AVX
VMASKMOVPS/D	y,y,111120,1 V,V,M	3	3	2p5 p23	4	2	AVX
VMASKMOVPS/D	m128,x,x	4	4	p0 p1 p4 p23	13	1	AVX
VMASKMOVPS/D	m256,y,y	4	4	p0 p1 p4 p23 p0 p1 p4 p23	14	2	AVX
VPGATHERDPS	x,[r+s*x],x	20	20	ρυ ρι μ <del>η</del> μεσ	14	9	AVX AVX2
VPGATHERDPS	x,[r+s x],x y,[r+s*y],y	34	34			12	AVX2 AVX2
VPGATHERDPS	y,[r+s y],y x,[r+s*x],x	15	15			8	AVX2 AVX2
VPGATHERQPS	x,[r+s x],x x,[r+s*y],x	22	22			7	AVX2 AVX2
VPGATHERQPS		12	12			7	AVX2 AVX2
VPGATHERDPD	X,[r+s*x],X	20	20			9	AVX2 AVX2
VPGATHERDPD	y,[r+s*x],y v [r+s*v] v	14	14			7	AVX2 AVX2
VPGATHERQPD	x,[r+s*x],x					9	
VEGATHERQED	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			1	

VCVTPD2PS         x,y         2         2         p1p5         5         1         AVX           VCVTPD2PS         x,m256         2         3         p1p6 p23         1         AVX           CVTSD2SS         x,m64         2         3         p1p6 p23         1         AVX           CVTPS2PD         x,x         2         2         p0 p25         2         1           CVTPS2PD         y,x         2         2         p0 p23         1         AVX           VCVTPS2PD         y,x         2         2         p0 p23         1         AVX           VCVTPS2PD         y,x         2         2         p0 p23         1         AVX           VCVTPS2PD         y,x         2         2         p0 p23         1         AVX           VCVTS2S2D         x,x         2         2         p0 p23         1         AVX           CVTS2S2D         x,x         1         1         p1         3         1         AVX           CVTD02PS         x,x         1         1         p1         3         1         AVX           CVTT(T) PS2DQ         x,m128         1         2         p1 p23 <t< th=""><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></t<>								
CVTSD2SS	VCVTPD2PS	x,y	2	1	p1 p5	5	1	AVX
CVTSD2SS	VCVTPD2PS	x,m256	2	3	p1 p5 p23		1	AVX
CVTPS2PD	CVTSD2SS	x,x	2	2	p1 p5	4	1	
CVTPS2PD	CVTSD2SS	x,m64	2	3	p1 p5 p23		1	
CVTPS2PD	CVTPS2PD	X,X	2	2	p0 p5	2	1	
VCVTPS2PD         y.x         2         2         2         p0 p5         5         1         AVX           VCVTPS2PD         y.m128         2         2         p0 p5         2         1         AVX           CVTSS2SD         x.x         2         2         p0 p23         1         1           CVTDQ2PS         x.x         1         1         p1         3         1           CVTDQ2PS         y.m128         1         2         p1 p23         1         1           VCVTDQ2PS         y.m256         1         2         p1 p23         1         AVX           VCVTIQ1 PS2DQ         x.x         1         1         p1         3         1         AVX           VCVTIQ1 PS2DQ         x.x         1         1         p1         3         1         AVX           VCVTIQ1 PS2DQ         y.x         1         1         p1         3         1         AVX           VCVTIQ2PD         y.m64         2         p1 p23         1         AVX           VCVTDQ2PD         y.m64         2         p1 p5         4         1           VCVTTQ2PD         y.m64         2         p1 p5         6	CVTPS2PD		2	2			1	
VCVTPS2PD         y,m128         2         2         p0 p23         1         AVX           CVTSS2SD         x,m32         2         2         p0 p5         2         1           CVTDQ2PS         x,m32         2         2         p0 p23         1           CVTDQ2PS         x,m128         1         1         p1         3         1           CVTDQ2PS         y,y         1         1         p1         3         1         AVX           VCVTDQ2PS         y,m256         1         2         p1 p23         1         AVX           VCVT(T) PS2DQ         x,m128         1         2         p1 p23         1         AVX           VCVT(T) PS2DQ         y,m256         1         2         p1 p23         1         AVX           VCVT(T) PS2DQ         y,m256         1         2         p1 p5         4         1           CVTDQ2PD         x,m64         2         2         p1 p5         4         1           CVTTQ2PD         x,m64         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         <	VCVTPS2PD	V,X	2	2		5	1	AVX
CVTSS2SD	VCVTPS2PD	-	2	2			1	AVX
CVTSQ2PS		•				2	1	
CVTDQ2PS						_		
CVTDQ2PS		•				3		
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         y,y         1         1         p1 p23         1         AVX           VCVT(T) PS2DQ         y,y         1         1         p1 p23         1         AVX           VCVTTQ2PD         y,m256         1         2         p1 p23         1         AVX           CVTDQ2PD         x,x         2         2         p1 p23         1         AVX           VCVTDQ2PD         x,m64         2         2         p1 p5         4         1           VCVTDQ2PD         y,m128         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x,x         2         2         p1 p5         4         1         CVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX         AVX         AVX         AVX         AVX         AVX         AVX         AVX         AVX					-			
VCVTDQ2PS         y,m256         1         2         p1 p23         1         AVX           CVT(T) PS2DQ         x,x         1         1         p1 p23         1         AVX           VCVT(T) PS2DQ         y,x         1         1         p1 p23         1         AVX           VCVT(T) PS2DQ         y,y         1         1         p1 p23         1         AVX           VCVTDQ2PD         x,x         2         2         p1 p23         1         AVX           CVTDQ2PD         x,m64         2         2         p1 p23         1         AVX           VCVTDQ2PD         y,x         2         2         p1 p5         6         1         AVX           VCVT(DQ2PD         y,x         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x,x         2         2         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           VCVT(T)PS2PI         x,m64         1         2         <			· ·			3		AVX
CVT(T) PS2DQ					-			
CVT(T) PS2DQ         x,m128         1         2         p1 p23         1         AVX           VCVT(T) PS2DQ         y,y         1         1         p1 p23         1         AVX           VCVT(DQ2PD         x,x         2         2         p1 p23         1         AVX           CVTDQ2PD         x,m64         2         2         p1 p23         1         AVX           VCVTDQ2PD         y,m428         2         2         p1 p23         1         AVX           VCVTDQ2PD         y,m128         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x,x         2         2         p1 p5         4         1         AVX           VCVT(T)PD2DQ         x,x         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x,m128         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2PB         x,m64         1         2         p1 p23         3         1         AVX           CVT(T)PS2PI         x,m64		-	.			3		7.47
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 p5         6         1         AVX           VCVTDQ2PD         y,x         2         2         p1 p5         6         1         AVX           VCVTTDQ2PD         y,m128         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x,x         2         2         p1 p5         4         1         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           CVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           CVT(T)PD2PB         x,m64         1         2         p1 p23         3         1         AVX           CVT(T)PS2PI         mm,x         2         2         p1 p23         1         1         1         1<	1 ' '				· ·			
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         AVX           VCVTDQ2PD         y,m128         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x,x         2         2         p1 p5         4         1         AVX           VCVT(T)PD2DQ         x,m128         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m128         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           CVT(T)PS2PI         mm,x         2         2         p1 p23         3         1           CVT(T)PS2PI         mm,x         2         2         p1 p23         1         1           CVT(T)PS2PI         x,m64         2         2	' '					3		<b>۸\/</b> Y
CVTDQ2PD	1 ' '				· ·	3		
CVTDQ2PD         x,m64         2         2         p1 p23         1           VCVTDQ2PD         y.x         2         2         p1 p5         6         1         AVX           VCVTDQ2PD         y.m128         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x.x         2         2         p1 p5         4         1         AVX           VCVT(T)PD2DQ         x.y         2         2         p1 p5         6         1         AVX           VCVT(T)PD2DQ         x.m256         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2PB         x.m64         1         2         p1 p5         4         1         AVX           CVT(T)PS2PI         mm,x         2         2         p1 p5         4         1         CVT(T)PS2PI         mm,m128         2         p1 p5         4         1         CVT(T)PS2PI         mm,m28         2         p1 p5         4         1         CVT(T)PD2PI         mm,m22         2         p1 p5         4         1         CVT(T)PD2PI         mm,m28         2         p1 p5         4         1         CVT(T)PD2PI         m,m32         2	1	•				1		AVA
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         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           CVTT(PPSPS         x,m64         1         2         p1 p23         3         3           CVT(T)PS2PI         mm,x         2         2         p1 p5         4         1           CVT(T)PS2PI         mm,m128         2         2         p1 p5         4         1           CVT(T)PS2PI         x,m64         2         2         p1 p5         4         1           CVT(T)PS2PI         mm,m128         2         2         p1 p5         4         1           CVT(T)PD2PI         mm,x         2         2         p1 p5 <t< td=""><td></td><td></td><td></td><td></td><td></td><td>4</td><td></td><td></td></t<>						4		
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         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m64         1         2         p1 p23         3         1         AVX           CVTPI2PS         x,m64         1         2         p1 p23         3         1         AVX           CVT(T)PS2PI         mm,x         2         2         p1 p5         4         1         1         1         1         1         1         1         1         1         1         1         1         1         1         1         4         4         1         2         1         1         1         4         4         1         2         1         1         1         1         4         1         2         2         1         1         1         2         2         1         1         2         1 <t< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>A) //</td></t<>								A) //
CVT(T)PD2DQ         x,x         2         2         p1 p5         4         1           CVT(T)PD2DQ         x,m128         2         3         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,y         2         2         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           CVTP12PS         x,mm         1         1         p1         4         4           CVTP12PS         x,m64         1         2         p1 p5         4         1           CVT(T)PS2PI         mm,x         2         2         p1 p5         4         1           CVT(T)PS2PI         mm,m128         2         2         p1 p5         4         1           CVTP12PD         x,m64         2         2         p1 p5         4         1           CVT(T)PD2PI         mm,x         2         2         p1 p5         4         1           CVT(T) PD2PI         mm,m128         2         3         p1 p5 p23         1         1           CVT(T) PS12SS         x,r32         2         p1 p5         4         1         1		-				6		
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,y         2         2         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m66         1         2         p1 p23         3         1         AVX           CVT(P12PS         x,m64         1         2         p1 p23         3         3         1         CVT(T)PS2PI         mm,x         2         2         p1 p5         4         1         CVT(T)PS2PI         mm,x128         2         2         p1 p23         1         1         CVT(T)PS2PI         x,mm2         2         2         p1 p23         1         1         1         1         1         1         1         1         1         1         1         1         1         1         2         1         1         1         1         1         1         1         2         1         2         1         2         1         2         1         2         1         2         2         1         1         2         2		-						AVX
VCVT(T)PD2DQ         x,y         2         2         p1 p5 p23         1         AVX           VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           CVT(PI2PS         x,mm         1         1         p1 p23         3         3           CVT(T)PS2PI         mm,x         2         2         p1 p5         4         1         1         CVT(T)PS2PI         mm,m128         2         2         p1 p5         4         3         CVT(T)S2SS         x,r32         2         2         p1 p23	1 ' '					4		
VCVT(T)PD2DQ         x,m256         2         3         p1 p5 p23         1         AVX           CVTPI2PS         x,m64         1         2         p1 p23         3         3           CVT(T)PS2PI         mm,x         2         2         p1 p5         4         1           CVT(T)PS2PI         mm,m128         2         2         p1 p5         4         1           CVT(T)PS2PI         mm,m128         2         2         p1 p5         4         1           CVTPI2PD         x,m64         2         2         p1 p5         4         1           CVT(T)PD2PI         mm,x         2         2         p1 p5         4         1           CVT(T)PD2PI         mm,x         2         2         p1 p5         4         1           CVT(T)PD2PI         mm,m128         2         3         p1 p5 p23         1         1           CVT(T)PD2PI         mm,x         2         2         p1 p5         4         1           CVT(T)SD2SI         x,r32         2         2         p1 p5         4         3           CVT(T)SS2SI         r32,m32         2         2         p1 p5         4         3	1 ' '							
CVTPI2PS         x,mm         1         1         p1         4         4           CVTPI2PS         x,m64         1         2         p1 p23         3           CVT(T)PS2PI         mm,m12         2         2         p1 p5         4         1           CVT(T)PS2PI         mm,m128         2         2         p1 p23         1           CVTPI2PD         x,mm         2         2         p1 p23         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         1         1         1         1         1         1         1         1         1         1         1         2         p1 p5         4         1         1         1         1         2         p1 p5         4         1         1         2         1         2         p1 p5         4         3         3         2         2         p1 p5         4         3         3         2         2         p0 p1         4	1 ' '					6		
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           CVT(P12PD         x,mm         2         2         p1 p5         4         1           CVT(T)PD2PI         x,m64         2         2         p1 p5         4         1           CVT(T) PD2PI         mm,x         2         2         p1 p5         4         1           CVT(T) PD2PI         mm,x128         2         3         p1 p5 p23         1         1           CVT(T) PD2PI         mm,m128         2         3         p1 p5 p23         1         1           CVT(T) PD2PI         mm,x128         2         3         p1 p5 p23         1         1           CVTS12SS         x,r32         2         2         p1 p23         3         1           CVT(T)SS2SI         r32,m32         2         2         p1 p5         4         3           CVT(T)SD2SI         x,r32/64         2         2         p1 p23         1           VCVT(T)SD2SI	1							AVX
CVT(T)PS2PI         mm,x         2         2         p1 p5         4         1           CVT(T)PS2PI         mm,m128         2         2         p1 p23         1           CVT(P12PD         x,mm         2         2         p1 p5         4         1           CVT(T)PD2PI         x,m64         2         2         p1 p5         4         1           CVT(T) PD2PI         mm,x         2         2         p1 p5 p23         1           CVT(T) PD2PI         mm,m128         2         3         p1 p5 p23         1           CVT(T) PD2PI         mm,m128         2         3         p1 p5 p23         1           CVT(S12SS         x,r32         2         2         p1 p5         4         3           CVT(T)SS2SI         r32,x         2         2         p0 p1         4         1           CVT(S12SD         x,r32/64         2         2         p1 p5         4         3           CVT(T)SD2SI         r32/64,x         2         2         p0 p1         4         1           CVT(T)SD2SI         r32,m64         2         3         p1 p5         4         1         F16C           VCVTPS2PH						4		
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         1           CVT(T) PD2PI         mm,x         2         2         p1 p5         4         1           CVT(T) PD2PI         mm,m128         2         3         p1 p5 p23         1         1           CVT(T) PD2PI         mm,m128         2         3         p1 p5 p23         1         1           CVT(T)SU2SS         x,r32         2         2         p1 p5         4         3         3           CVT(T)SS2SI         r32,x         2         2         p0 p1         4         1         1         CVT(T)SS2SI         r32,m32         2         2         p1 p5         4         3         CVTS12SD         x,r32/64         2         2         p1 p5         4         3         CVTS12SD         x,m32         2         2         p1 p23         3         1         CVT(T)SD2SI         r32/64,x         2         2         p0 p1         4         1         F16C         VCVTPS2PH				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         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         p1 p5         4         1           CVT(T)SS2SI         r32,m32         2         3         p0 p1 p23         1         1           CVTSI2SD         x,r32/64         2         2         p1 p5         4         3         3           CVT(T)SD2SI         r32,m64         2         2         p1 p23         3         1           CVT(T)SD2SI         r32,m64         2         3         p0 p1 p23         1         1           VCVTPS2PH         x,v,i         4         p1 p4 p5 p23         1         F16C           VC	1 ' '					4		
CVTPI2PD         x,m64         2         2         p1 p23         1           CVT(T) PD2PI         mm,x         2         2         p1 p5         4         1           CVT(T) PD2PI         mm,m128         2         3         p1 p5 p23         1         1           CVTSI2SS         x,r32         2         2         p1 p5         4         3           CVTSI2SS         x,m32         1         2         p1 p23         3           CVT(T)SS2SI         r32,xx         2         2         p0 p1         4         1           CVTSI2SD         x,r32/64         2         2         p1 p5         4         3           CVTSI2SD         x,m32         2         2         p1 p5         4         3           CVT(T)SD2SI         r32/64,x         2         2         p0 p1         4         1           CVT(T)SD2SI         r32,m64         2         3         p0 p1 p23         1         1           VCVTPS2PH         x,v,i         2         2         p1 p5         4         1         F16C           VCVTPH2PS         v,x         2         2         p1 p5         4         1         F16C <td>1 ' '</td> <td>mm,m128</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>	1 ' '	mm,m128						
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         3           CVT(T)SS2SI         r32,x         2         2         p0 p1         4         1         1         1         2         p1 p23         3         2         2         p1 p23         3         3         2         2         p1 p23         1         P16C         2         2         p1 p23         1         P16C         2         p1 p23		x,mm			p1 p5	4		
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         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         ADDSS/D PS/D       x,x / v,v,v,v       1       1       p1 p23       1       F16C         ADDSS/D PS/D       x,m / v,v,m       1       2       p1 p23       1       1		x,m64		1	p1 p23		1	
CVTSI2SS       x,r32       2       2       p1 p5       4       3         CVTSI2SS       x,m32       1       2       p1 p23       3         CVT(T)SS2SI       r32,x       2       2       p0 p1       4       1         CVT(T)SS2SI       r32,m32       2       3       p0 p1 p23       1         CVTSI2SD       x,r32/64       2       2       p1 p5       4       3         CVT(T)SD2SI       r32/64,x       2       2       p0 p1       4       1         CVT(T)SD2SI       r32,m64       2       3       p0 p1 p23       1         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         ADDSS/D PS/D       x,x / v,v,v       1       1       p1       3       1         ADDSS/D PS/D       x,m / v,v,m       1       2       p1 p23       1		mm,x			p1 p5	4	1	
CVTSI2SS       x,m32       1       2       p1 p23       3         CVT(T)SS2SI       r32,x       2       2       p0 p1       4       1         CVT(T)SS2SI       r32,m32       2       3       p0 p1 p23       1         CVTSI2SD       x,r32/64       2       2       p1 p5       4       3         CVT(T)SD2SI       r32/64,x       2       2       p0 p1       4       1         CVT(T)SD2SI       r32,m64       2       3       p0 p1 p23       1         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         ADDSS/D PS/D       x,x / v,v,v       1       1       p1 p23       1       F16C         SUBSS/D PS/D       x,m / v,v,m       1       2       p1 p23       1       1	CVT(T) PD2PI	mm,m128	2	3	p1 p5 p23		1	
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         7           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         x,m / v,v,w         1         1         p1 p23         1           ADDSS/D PS/D         x,m / v,v,m         1         2	CVTSI2SS	x,r32	2	2	p1 p5	4	3	
CVT(T)SS2SI       r32,m32       2       3       p0 p1 p23       1         CVTSI2SD       x,r32/64       2       2       p1 p5       4       3         CVTSI2SD       x,m32       2       2       p1 p23       3         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       x,x / v,v,v,v       1       1       p1 p23       1       ADDSS/D PS/D         SUBSS/D PS/D       x,m / v,v,m       1       2       p1 p23       1       1	CVTSI2SS	x,m32	1	2	p1 p23		3	
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         x,x / v,v,v         1         1         p1         3         1           ADDSS/D PS/D         x,m / v,v,m         1         2         p1 p23         1         1	CVT(T)SS2SI	r32,x	2	2	p0 p1	4	1	
CVTSI2SD         x,m32         2         2         p1 p23         3           CVT(T)SD2SI         r32/64,x         2         2         p0 p1         4         1           CVT(T)SD2SI         r32,m64         2         3         p0 p1 p23         1           VCVTPS2PH         x,v,i         2         2         p1 p5         4         1         F16C           VCVTPS2PH         m,v,i         4         4         p1 p4 p5 p23         1         F16C           VCVTPH2PS         v,x         2         2         p1 p5         4         1         F16C           VCVTPH2PS         v,m         2         2         p1 p23         1         F16C           Arithmetic         ADDSS/D PS/D         x,x / v,v,v         1         1         p1         3         1           ADDSS/D PS/D         x,m / v,v,m         1         2         p1 p23         1         1	CVT(T)SS2SI	r32,m32	2	3	p0 p1 p23		1	
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         x,x / v,v,v         1         1         p1         3         1           ADDSS/D PS/D         x,m / v,v,m         1         2         p1 p23         1         1	CVTSI2SD	x,r32/64	2	2	p1 p5	4	3	
CVT(T)SD2SI       r32/64,x       2       2       p0 p1       4       1         CVT(T)SD2SI       r32,m64       2       3       p0 p1 p23       1         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       x,x / v,v,v       1       1       p1       3       1         ADDSS/D PS/D       x,m / v,v,m       1       2       p1 p23       1       1	CVTSI2SD	x,m32	2	2	p1 p23		3	
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 SUBSS/D PS/D SUBSS/D PS/D       x,x / v,v,v       1       1       p1 p23       1         ADDSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D       x,m / v,v,m       1       2       p1 p23       1	CVT(T)SD2SI	r32/64,x	2	2		4	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 SUBSS/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 SUBSS/D PS/D         x,m / v,v,m         1         2         p1 p23         1	CVT(T)SD2SI	r32,m64	2	3			1	
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           SUBSS/D PS/D SUBSS/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	1 ' '		2	2		4	1	F16C
VCVTPH2PS         v,x         2         2         p1 p5         4         1         F16C           VCVTPH2PS         v,m         2         2         p1 p23         4         1         F16C           Arithmetic         ADDSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/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 SUBSS/D PS/D         x,m / v,v,m         1         2         p1 p23         1								
VCVTPH2PS         v,m         2         2         p1 p23         1         F16C           Arithmetic         ADDSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D         x,x / v,v,v         1         1         p1         3         1           ADDSS/D PS/D SUBSS/D PS/D SUBSS/D PS/D         x,m / v,v,m         1         2         p1 p23         1						4		
ADDSS/D PS/D SUBSS/D PS/D x,x / v,v,v 1 1 1 p1 3 1  ADDSS/D PS/D x,m / v,v,m 1 2 p1 p23 1						-		
ADDSS/D PS/D SUBSS/D PS/D x,x / v,v,v 1 1 1 p1 3 1  ADDSS/D PS/D x,m / v,v,m 1 2 p1 p23 1	Arithmetic							
SUBSS/D PS/D     x,x / v,v,v     1     1     p1     3     1       ADDSS/D PS/D     x,m / v,v,m     1     2     p1 p23     1								
ADDSS/D PS/D   x,m / v,v,m   1   2   p1 p23   1		x,x / v,v,v	1	1	p1	3	1	
SUBSS/D PS/D   x,m / v,v,m   1   2   p1 p23   1		. , ,						
		x,m / v,v,m	1	2	p1 p23		1	
			1			3	1	SSE3

			110	iowen			
ADDSUBPS/D	x,m / v,v,m	1	2	p1 p23		1	SSE3
HADDPS/D		2		-4.0-F	_		0050
HSUBPS/D HADDPS/D	x,x / v,v,v	3	3	p1 2p5	5	2	SSE3
HSUBPS/D	x,m / v,v,m	4	4	p1 2p5 p23		2	SSE3
MULSS/D PS/D	x,x / v,v,v	1	1	p01	5	0.5	0020
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	у,у	3	3	2p0 p15	7	2	AVX
VRCPPS	y,m256	4	4	2p0 p15 p23		2	AVX
CMPccSS/D		_		4	0	_	
CMPccPS/D	x,x / v,v,v	1	1	p1	3	1	
CMPccSS/D CMPccPS/D	, m / , , , , m	2	2	n1 n22		4	
(U)COMISS/D	x,m / v,v,m	2 1	2	p1 p23		1 1	
(U)COMISS/D	x,x x,m32/64	2	2	p1 p1 p23		1 1	
MAXSS/D PS/D	X,11132/0 <del>4</del>	2		ρι μ23		<b>'</b>	
MINSS/D PS/D	x,x / v,v,v	1	1	p1	3	1	
MAXSS/D PS/D	7,7,7,0,0	•	•	ρ.	Ū	•	
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	y m i	3	3	2p1 p23		2	SSE4.1
DPPS	v,m,i x,x,i / v,v,v,i	4	4	2p1 p23 2p0 p1 p5	14	2	SSE4.1
DPPS	x,m,i / v,v,m,i	6	6	2p0 p1 p3 2p0 p1 p5 p23 p6	14	1	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	Ū	1 1	SSE4.1
VFMADD	х,20,.	•	•	po p. po p20		•	002
(all FMA instr.)	<b>V,V,V</b>	1	1	p01	5	0.5	FMA
VFMADD							
(all FMA instr.)	v,v,m	1	2	p01 p23		0.5	FMA
Math							
SQRTSS/PS	X,X	1	1	p0	11	7	
SQRTSS/PS	x,m128	1	2	p0 p23		7	
VSQRTPS	y,y	3	3	2p0 p15	19	14	AVX
VSQRTPS	y,m256	4	4	2p0 p15 p23		14	AVX
SQRTSD/PD	X,X	1	1	p0	16	8-14	
SQRTSD/PD	x,m128	1	2	p0 p23	00.00	8-14	A > 0.4
VSQRTPD	y,y	3	3	2p0 p15	28-29	16-28	AVX
VSQRTPD	y,m256	4	4	2p0 p15 p23	-	16-28	AVX
RSQRTSS/PS	X,X	1	1	p0	5	1 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				_	_		
RPS/PD	x,x / v,v,v	1	1	p5	1	1	
AND/ANDN/OR/XO RPS/PD		4	2	mE m00		1	
1 3/1 D	x,m / v,v,m	1		p5 p23		l I	
Othor							
Other		_	_				A) 0/
VZEROUPPER		4	4	none		1	AVX
							AVX,
VZEROALL		12	12	none		10	32 bit
							AVX,
VZEROALL		20	20	none		8	64 bit
LDMXCSR	m32	3	3	p0 p6 p23	6	3	
STMXCSR	m32	3	4	p0 p4 p6 p237	7	1	
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.

μορs fused domain:

The number of  $\mu$ 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	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instruc-							
tions							
MOV	r,i	1	1	p0156		0.25	
MOV	r8/16,r8/16	1	1	p0156	1	0.25	
MOV	r32/64,r32/64	1	1	p0156	0-1	0.25	may be elim.
MOV	r8l,m	1	2	p23 p0156		0.5	
MOV	r8h,m	1	1	p23		0.5	
MOV	r16,m	1	2	p23 p0156		0.5	
MOV	r32/64,m	1	1	p23	2	0.5	all addressing modes
MOV	m,r	1	2	p237 p4	3	1	

MOV	m,i	1	2	p237 p4		1	
MOVNTI	m,r	2	2	p23 p4	~400	1	
MOVSX MOVZX MOVSXD	r,r	1	1	p0156	1	0.25	
MOVSX MOVZX	r16,m8	1	2	p23 p0156		0.5	
MOVSX MOVZX MOVSXD	r,m	1	1	p23		0.5	all other combinations
CMOVcc	r,r	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
LEA	r32/64,m	1	1	p15	1	0.5	address size 1 or 2 components in
LEA	r32/64,m	1	1	p1	3	1	address 3 components
LEA	r32/64,m	1	1	p1		1	in address rip relative
BSWAP	r32	1	1	p15	1	0.5	address
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	r64,r64,i	1	1	p1	3	1	
IMUL	r16,m16,i	2	3	p1 p0156 p23		1	
IMUL	r32,m32,i	1	2 2	p1 p23		1	
IMUL MULX	r64,m64,i r32,r32,r32	1 3	3	p1 p23 p1 2p056	4	1	AVX2
MULX	r32,r32,m32	3	4	p1 2p056 p23	4	1	AVX2 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	7,7,7,2
DIV	r16	11	11	p0 p1 p5 p6	23-26	9	
DIV	r32	10	10	p0 p1 p5 p6	22-29	9	
DIV	r64	36	36	p0 p1 p5 p6	32-95	21-73	
IDIV	r8	9	9	p0 p1 p5 p6	23-26	6	
IDIV	r16	10	10	p0 p1 p5 p6	23-26	6	
IDIV	r32	9	9	p0 p1 p5 p6	22-29	6	
IDIV	r64	59	59	p0 p1 p5 p6	39-103	24-81	
CBW		1	1	p0156	1		
CWDE		1	1	p0156	1		
CDQE		1	1	p0156	1		
CWD		2	2	p0156	1		
CDQ		1	1	p06	1		
CQO		1	1	p06	1		
•	. '						, !

POPCNT	r,r	1	1	p1	3	1	SSE4.2
POPCNT	r,m	1	2	p1 p23		1	SSE4.2
CRC32	r,r	1	1	p1	3	1	SSE4.2
CRC32	r,m	1	2	p1 p23		1	SSE4.2
Logic instruc-							
AND OR XOR	r,r/i	1	1	p0156	1	0.25	
AND OR XOR	r,m	1	2	p0156 p23		0.5	
AND OR XOR	m,r/i	2	4	2p0156 2p237 p4	6	1	
TEST	r,r/i	1	1	p0156	1	0.25	
TEST	m,r/i	1	2	p0156 p23	1	0.5	
SHR SHL SAR	r,i	1	1	p06	1	0.5	
SHR SHL SAR	m,i	3	4	2p06 p237 p4		2	
SHR SHL SAR	r,cl	3	3	3p06	2	2	
SHR SHL SAR	m,cl	5	6	3p06 2p23 p4		4	
ROR ROL	r,1	2	2	2p06	1	1	short form
ROR ROL	r,i	1	1	p06	1	0.5	
ROR ROL	m,i	4	5	2p06 2p237 p4		2	
ROR ROL	r,cl	3	3	3p06	2	2	
ROR ROL	m,cl	5	6	3p06 p23 p4		4	
RCR RCL	r,1	3	3	2p06 p0156	2	2	
RCR RCL	m,1	4	6			3	
RCR RCL	r,i	8	8	p0156	6	6	
RCR RCL	m,i	11	11			6	
RCR RCL	r,cl	8	8	p0156	6	6	
RCR RCL	m,cl	11	11			6	
SHRD SHLD	r,r,i	1	1	p1	3	1	
SHRD SHLD	m,r,i	3	5			2	
SHLD	r,r,cl	4	4	p0156	3	2	
SHRD	r,r,cl	4	4	p0156	4	2	
SHRD SHLD	m,r,cl	5	7	·		4	
SHLX SHRX SARX	r,r,r	1	1	p06	1	0.5	BMI2
SHLX SHRX SARX	r,m,r	2	2	p06 p23		0.5	BMI2
RORX	r,r,i	1	1	p06	1	0.5	BMI2
RORX	r,m,i	2	2	p06 p23		0.5	BMI2
ВТ	r,r/i	1	1	p06	1	0.5	
BT	m,r	10	10			5	
BT	m,i	2	2	p06 p23		0.5	
BTR BTS BTC	r,r/i	1	1	p06	1	0.5	
BTR BTS BTC	m,r	10	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

	I	1 .			_	ı .	
TZCNT	r,r	1	1	p1	3	1	BMI1
TZCNT	r,m	1	2	p1 p23		1	BMI1
ANDN	r,r,r	1	1	p15	1	0.5	BMI1
ANDN	r,r,m	1	2	p15 p23	1	0.5	BMI1
BLSI BLSMSK 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	p15	1	0.5	BMI2
BZHI	r,m,r	1	2	p15 p23		0.5	BMI2
PDEP	r,r,r	1	1	p1	3	1	BMI2
PDEP	r,r,m	1	2	p1 p23		1	BMI2
PEXT	r,r,r	1	1	p1	3	1	BMI2
PEXT		1 1	2	p1 p23	3		BMI2
FEXI	r,r,m	'		ρτρ23		'	DIVIIZ
Control transfer	instructions						
JMP	short/near	1	1	p6		1-2	
JMP	r	1	1	p6		2	
JMP	m m	1	2	p23 p6		2	
Conditional jump	short/near	1	1	p6		1-2	predicted
Contaitional jump	oriorerioa.	'		ρο			taken
Conditional jump	short/near	1	1	p06		0.5-1	predicted not taken
Fused arithmetic and branch		1	1	р6		1-2	predicted taken
Fused arithmetic and branch		1	1	p06		0.5-1	predicted not 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	i	3	4	p23 2p6 p015		2	
BOUND	r,m	15	15			8	not 64 bit
INTO	,	4	4			5	not 64 bit
String instruc-							
tions							
LODSB/W		3	3	2p0156 p23		1	
LODSD/Q		2	2	p0156 p23		1	
REP LODS		5n+12		F		~2n	
STOS		3	3	p23 p0156 p4		1	
REP STOS		<2n		p=0 p0 . co p .		~0.5n	worst case
REP STOS		2.6/32B				1/32B	best case
1.21 0100		0,020				1/020	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	

			2.00	advven		
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 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	р1	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 1PP	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 6	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 70-100 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

WAIT	2	2	p01	1	
FNCLEX	5	5	p0156	22	
FNINIT	26	26		84	

## **Integer MMX and XMM instructions**

Instruction	Operands	µops fused domain	μορs unfused	μορs each port	Latonov	Recipro- cal through put	Comments
	Operanus	uomam	uomam	pops each port	Latericy	put	Comments
Move instruc- tions							
MOVD	r22/64 (v)mm	1	1	-0	4	1	
MOVD	r32/64,(x)mm	1	2	p0	1 3	1	
	m32/64,(x)mm	· ·	1	p237 p4	l		
MOVD	(x)mm,r32/64	1	1	p5	1	0.5	
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	0.25	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	1	p207 p4	3	0.5	SSE4.1
VMOVNTDQA	y,m256		1	p23	3	0.5	AVX2
PACKSSWB/DW	y,111230	'	'	ρ23	3	0.5	AVAZ
PACKUSWB	mm mm	3	3	n.E	2	2	
	mm,mm	3	3	p5			
PACKSSWB/DW PACKUSWB				m00 0mE			
	mm,m64	3	3	p23 2p5		2	
PACKSSWB/DW		_			_		
PACKUSWB	x,x / y,y,y	1	1	p5	1	1	
PACKSSWB/DW	,			00.5			
PACKUSWB	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	

1	ı	1	ı	1		1	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		2	2	·		1	AVX2
BD BQ DW DQ PSHUFB	y,m v,v / v,v,v	1	1	p5 p23 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	00020
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	р5	1	1	AVX2
VPBROADCAST B/W	x,m8/16	3	3	p01 p23 p5	5	1	AVX2

VPBROADCAST			_	-00	_	0.5	A) /)/O
D/Q	x,m32/64	1	1	p23	4	0.5	AVX2
VPBROADCAST B/W/D/Q	V V	1	1	p5	3	1	AVX2
VPBROADCAST	y,x	'	'	μS	]	ı	AVAZ
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
VPGATHERQD	x,[r+s*y],x	10	10			6	AVX2
VPGATHERDQ	x,[r+s*x],x	7	7			5	AVX2
VPGATHERDQ	y,[r+s*x],y	9	9			6	AVX2
VPGATHERQQ	x,[r+s*x],x	7	7			5	AVX2
VPGATHERQQ	y,[r+s*y],y	9	9			6	AVX2
Arithmetic in-							
structions							
PADD/SUB(S,US)	,			4.5		0.5	
B/W/D/Q	v,v / v,v,v	1	1	p15	1	0.5	
PADD/SUB(S,US) B/W/D/Q	\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \	1	2	n15 n22		0.5	
	v,m / v,v,m	'		p15 p23		0.5	
PHADD(S)W/D	VV /VVV	3	3	n1 2n5	3	2	SSSE3
PHSUB(S)W/D	v,v / v,v,v	3	3	p1 2p5	3		SSSES
PHADD(S)W/D PHSUB(S)W/D	v,m / v,v,m	4	4	p1 2p5 p23		2	SSSE3
PCMPEQB/W/D	V,111 / V,V,111	7	7	p i zpo pzo			333L3
PCMPGTB/W/D	v,v / v,v,v	1	1	p15	1	0.5	
PCMPEQB/W/D	V, V 7 V, V, V	'	'	pio	'	0.5	
PCMPGTB/W/D	v,m / v,v,m	1	2	p15 p23		0.5	
PCMPEQQ	v,v / v,v,v	1	1	p15 p25	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	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		_	ρο ρ2ο			0021.2
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	
1		1	1		1	1	1

	1 1		I	I	l	I	i i
PMIN/PMAX SB/SW/SD							
UB/UW/UD	x,x / y,y,y	1	1	p15	1	0.5	SSE4.1
PMIN/PMAX	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	•	'	Pio	'	0.0	00L4.1
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,v v,m	1	2	p15 p23	'	0.5	SSSE3
PSIGNB/W/D	v,v / v,v,v	1	1	p15 p25	1	0.5	SSSE3
PSIGNB/W/D	v,w / v,v,w v,m / v,w,m	1	2	p15 p23	! 	0.5	SSSE3
PSADBW	v, iii / v, v, iii v, v / v, v, v	1	1	p15 p25	5	1	000L0
PSADBW	v,w / v,v,w v,m / v,w,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
IVIF SADBVV	X,111,1 / V,V,111,1	4	4	ρυ 2μο μ2ο			33E4.1
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							
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		4	4	-0	4	4	
PSRAW/D/Q	v,i / v,v,i	1	1	р0	1	1	
VPSLLVD/Q							
VPSRAVD VPSRLVD/Q		3	3	200 05	2	2	AVX2
	V,V,V	3	3	2p0 p5			AVA2
VPSLLVD/Q VPSRAVD							
VPSRLVD/Q	v,v,m	4	4	2p0 p5 p23		2	AVX2
PSLLDQ	V, V, III	7	"	2ρο ρο ρ2ο		_	/\V/\Z
PSRLDQ	x,i / v,v,i	1	1	p5	1	1	
. 5	75,17 4,4,1	•			·		
String instruc-							
tions							
PCMPESTRI	x,x,i	8	8	6p05 2p16	4	4	SSE4.2
PCMPESTRI	x,m128,i	8	8	3p0 2p16 2p5 p23		4	SSE4.2
PCMPESTRM	x,x,i	9	9	3p0 2p16 4p5	11	11	SSE4.2
•			•	•	-		

	9	9	6p05 2p16 p23		5	SSE4.2
x,x,i	3	3	3p0	3	3	SSE4.2
x,m128,i	4	4	3p0 p23		3	SSE4.2
x,x,i	3	3	3p0	11	11	SSE4.2
x,m128,i	4	4	3p0 p23		3	SSE4.2
ctions						
x,x,i	1	1		5	1	CLMUL
x,m,i	2	2	p0 p23		1	CLMUL
x,x	1	1	p5	7	1	AES
x m	2	2	·		1.5	AES
,				14		AES
,				• •		AES
x,x,i	10	10	2p0 8p5	10	9	AES
x,m,i	10	10	2p0 p23 7p5		8	AES
	31	31			12	
	x,m128,i x,x,i x,m128,i ctions x,x,i x,m,i x,x  x,m x,x  x,m x,x x,m	x,x,i 3 x,m128,i 4 x,x,i 3 x,m128,i 4  ctions  x,x,i 1 x,m,i 2  x,x 1  x,m 2 x,x 2 x,x 3 x,x,i 10	x,x,i     3       x,m128,i     4       x,x,i     3       x,m128,i     4       4     4       ctions     1       x,x,i     1       x,m,i     2       2     2       x,x     2       x,x     2       x,x     3       x,x,i     10     10       x,m,i     10     10	x,x,i       3       3 p0         x,m128,i       4       4         x,x,i       3 p0         x,m128,i       4       4         3 p0       3p0 p23         3 p0       3p0 p23         2 p0 p23       3p0 p23         x,x,i       1 p0 p0 p23         x,x       1 p5         x,m       2 p5 p23         x,x       2 p5 p23         x,x,x       2 p5 p23         x,x,i       10 10 2p0 8p5         x,m,i       10 10 2p0 p23 7p5	x,x,i x,m128,i x,x,i x,m128,i       3       3 p0 p23 3 p0 p23 3 p0 p23       3         x,x,i x,m128,i       4       4       3p0 p23       11         ctions         x,x,i x,m,i       1       1       p0 p0 p23       5         x,x       1       1       p5 p23       7         x,m       2       2       p5 p23 2p5 p23       14         x,x,i       3       3       2p5 p23       14         x,x,i       10       10       2p0 8p5       10         x,m,i       10       10       2p0 p23 7p5	x,x,i       3       3       3p0 p23       3       3 yp0 p23       11 yp0 p23       11 yp0 p23       11 yp0 p23       1 yp0 p23

Floating point XMM and YMM instructions

Instruction	Operands	µops fused domain	µops unfused domain	μοps 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		ı	1		1	1	
MOVMSKPS/D	r32,x	1	1	p0	3	1	
VMOVMSKPS/D	r32,y	1	1	p0	3	1	
MOVNTPS/D	m128,x	1	2	p4 p237	~400	1	
VMOVNTPS/D	m256,y	1	2	p4 p237	~400	1	AVX
SHUFPS/D	x,x,i / v,v,v,i	1	1	p5	1	1	
SHUFPS/D	x,m,i / v,v,m,i	2	2	p5 p23		1	
VPERMILPS/PD	v,v,i	1	1	p5	1	1	AVX
VPERMILPS/PD	v,m,i	2	2	p5 p23		1	AVX
VPERMILPS/PD	V,V,V	1	1	p5	1	1	AVX
VPERMILPS/PD	v,v,m	2	2	p5 p23		1	AVX
VPERM2F128	y,y,y,i	1	1	p5	3	1	AVX
VPERM2F128	y,y,m,i	2	2	p5 p23		1	AVX
VPERMPS	y,y,y	1	1	p5	3	1	AVX2
VPERMPS	y,y,m	1	2	p5 p23		1	AVX2
VPERMPD	y,y,i	1	1	p5	3	1	AVX2
VPERMPD	y,m,i	2	2	p5 p23		1	AVX2
BLENDPS/PD	x,x,i / v,v,v,i	1	1	p015	1	0.33	SSE4.1
BLENDPS/PD	x,m,i / v,v,m,i	2	2	p015 p23		0.5	SSE4.1
BLENDVPS/PD	x,x,xmm0	2	2	2p5	2	2	SSE4.1
BLENDVPS/PD	x,m,xmm0	3	3	2p5 p23		2	SSE4.1
VBLENDVPS/PD	, , , , , , , , , , , , , , , , , , ,	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	P-P-P-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
VPGATHERQPD	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
i. Similarida D	, 2 ,, ,	,	1	I	I	1	

			Di O	aawen			
VPGATHERQPD	y,[r+s*y],y	9	9			6	AVX2
Conversion							
CVTPD2PS	V V	2	2	n1 n5	4	1	
CVTPD2PS	x,x x,m128	2	3	p1 p5 p1 p5 p23	4	1	
VCVTPD2PS	· ·	2	2		5	1	AVX
VCVTPD2PS	X,Y	2	3	p1 p5	5	-	
	x,m256	2	2	p1 p5 p23	_	1	AVX
CVTSD2SS CVTSD2SS	X,X	2	3	p1 p5	4	1	
	x,m64			p1 p5 p23		1	
CVTPS2PD	X,X	2 2	2	p0 p5	2	1	
CVTPS2PD	x,m64		2	p0 p23	_	1	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	p0 p1 p23	4-6	1	F16C
VCVTPS2PH	m,v,i	3	3	p1 p4 p23	T-0	1	F16C
VCVTPH2PS		2	2		4-6	1	F16C
VCVTPH2PS VCVTPH2PS	V,X	2	2	p1 p5	4-0	1	F16C
VOVIFHZES	v,m	4	4	p1 p23		ļ <b>I</b>	100

	l I						
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	
		•			•		00544
ROUNDSS/D PS/D	v,v,i	2	2	2p1	6	2	SSE4.1
ROUNDSS/D PS/D	y m i	3	3	2p1 p23		2	SSE4.1
DPPS	v,m,i x,x,i / v,v,v,i	4	4		12	2	SSE4.1
DPPS	x,x,i / v,v,v,i x,m,i / v,v,m,i	6	6	2p0 p1 p5	12	4	SSE4.1
DPPD		3	3	2p0 p1 p5 p23 p6	7	1	SSE4.1
DPPD	X,X,İ	3 4	4	p0 p1 p5	,		SSE4.1
	x,m128,i	4	4	p0 p1 p5 p23		1	33E4.1
VFMADD	VVV	1	1	201	5	0.5	FMA
(all FMA instr.)	V,V,V	ı	'	p01	ິບ	0.5	LIVIA
VFMADD	v v m	1	2	n01 n23		0.5	FMA
(all FMA instr.)	v,v,m	I		p01 p23		0.5	FIVIA
Math							
Math SQRTSS		1	1	р0	11	4	
JUNITION	X,X	ı	'	l ho	1.1	, <del>-</del>	I

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							
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	
Other							
Other VZEROUPPER		4	4	200		4	AVX
VZEROUPPER		4	4	none		1	
VZEDOALI		10	12	nono		10	AVX, 32 bit
VZEROALL		12	12	none		10	
VZEROALL		20	20	none		8	AVX, 64 bit
LDMXCSR	m32	3	3	none p0 p6 p23	6	3	04 bit
STMXCSR	m32	3	4		7	1	
FXSAVE	m4096	111	4	p0 p4 p6 p237	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  $\mu$ 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	μορs fused domain	μορs unfused domain	μops each port	Latency	Recipro- cal through put	Comments
Move instruc-							
tions							
MOV	r,i	1	1	p0156		0.25	
MOV	r8/16,r8/16	1	1	p0156	1	0.25	
MOV	r32/64,r32/64	1	1	p0156	0-1	0.25	may be elim.
MOV	r8l,m	1	2	p23 p0156		0.5	
MOV	r8h,m	1	1	p23		0.5	
MOV	r16,m	1	2	p23 p0156		0.5	
MOV	r32/64,m	1	1	p23	2	0.5	all addressing modes
MOV	m,r	1	2	p237 p4	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	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		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
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 SUB	r,r/i	1	1	p0156	1	0.25	

ADD SUB	r,m	1	2	p0156 p23		0.5	
ADD SUB	m,r/i	2	4	2p0156 2p237 p4	5	1	
ADC SBB	r,r/i	1	1	p06	1	1	
ADC SBB	r,m	2	2	p06 p23		1	
ADC SBB	m,r/i	4	6	3p0156 2p237 p4	5	2	
OMP	<i>!</i>	_	_	-0450	4	0.05	
CMP	r,r/i	1	1	p0156	1	0.25 0.5	
CMP INC DEC NEG	m,r/i	1	2	p0156 p23	1		
NOT	r	1	1	p0156	ı	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	111	2	2	p1 p56	4	ı	not 64 bit
AAS		2	2		4		not 64 bit
DAA DAS		3	3	p1 p056	4		not 64 bit
AAD		3	3	p1 2p056			
				p1 2p056	4	7	not 64 bit
AAM	0	11	11	p0 p1 p5 p6	23	7	not 64 bit
MUL IMUL	r8	1	1	p1	3	1	
MUL IMUL	r16	4	4	p1 p0156	4	2	
MUL IMUL	r32	3	3	p1 p0156	4	1	
MUL IMUL	r64	2	2	p1 p6	3	1	
MUL IMUL	m8	1	2	p1 p23		1	
MUL IMUL	m16	4	5	p1 3p0156 p23		2	
MUL IMUL	m32	3	4	p1 2p0156 p23		2	
MUL IMUL	m64	2	3	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 3	1	
IMUL	r64,r64,i	1	1	p1	3	1	
IMUL IMUL	r16,m16,i	2	3	p1 p0156 p23		1	
IMUL	r32,m32,i	-	2 2	p1 p23		1	
MULX	r64,m64,i	1	3	p1 p23 p1 2p056	4	1 1	AVX2
MULX	r32,r32,r32 r32,r32,m32	3 3	4	p1 2p056 p23	4	1	AVX2 AVX2
MULX	r64,r64,r64	2	2	p1 2000 p23	4	1	AVX2 AVX2
MULX	r64,r64,m64	2	3	p1 p6 p23	7	1	AVX2 AVX2
DIV	r8	10	10	p0 p1 p5 p6	23	6	AVAZ
DIV	r16	10	10	p0 p1 p5 p6	23	6	
DIV	r32	10	10	p0 p1 p5 p6	26	6	
DIV	r64	36	36	p0 p1 p5 p6	35-88	21-83	
IDIV	r8	11	11				
				p0 p1 p5 p6	24	6	
IDIV	r16	10	10	p0 p1 p5 p6	23	6	
IDIV	r32	10	10	p0 p1 p5 p6	26	6	
IDIV	r64	57	57	p0 p1 p5 p6	42-95	24-90	
CBW		1	1	p0156	1		
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           POPCNT         r,m         1         2         p1 p23         1           CRC32         r,r         1         1         p1         3         1           CRC32         r,m         1         2         p1 p23         1	SSE4.2 SSE4.2 SSE4.2 SSE4.2
CRC32 r,r 1 1 p1 3 1 CRC32 r,m 1 2 p1 p23	SSE4.2
CRC32 r,m 1 2 p1 p23 1	
	33E4.2
Logic instruc-	
AND OR XOR r,r/i 1 1 p0156 1 0.25	
AND OR XOR r,m 1 2 p0156 p23 0.5	
AND OR XOR m,r/i 2 4 2p0156 2p237 p4 5 1	
TEST r,r/i 1 1 p0156 1 0.25	
TEST m,r/i 1 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	311011 101111
ROR ROL m,cl 5 6 3p06 p23 p4 4	
RCR RCL r,1 3 3 2p06 p0156 2 2	
RCR RCL m,1 4 6 3	
RCR RCL   r,i   8   8   p0156   6   6	
RCR RCL   m,i   11   11   6	
RCR RCL   r,cl   8   8   p0156   6   6	
RCR RCL   m,cl   11   11   6	
SHRD SHLD	
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	
BT m,i 2 2 p06 p23 0.5	
BTR BTS BTC	
BTR BTS BTC   m,r   10   11   5	
BSF BSR r,m 1 2 p1 p23 1	
SETcc r 1 1 p06 1 0.5	
SETcc m 2 3 p06 p237 p4 1	
CLC         1         0         none         0.25	
STC 1 1 p0156 0.25	
CMC 1 1 p0156 1 1	
CLD STD 3 3 p15 p6 4	
LZCNT   r,r   1   1   p1   3   1	LZCNT
LZCNT   r,m   1   2   p1 p23   1	LZCNT

TZCNT	r,r	1	1	p1	3	1	BMI1
TZCNT	r,m	1	2	p1 p23		1	BMI1
ANDN	r,r,r	1	1	p15	1	0.5	BMI1
ANDN	r,r,m	1	2	p15 p23	1	0.5	BMI1
BLSI BLSMSK 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	p15	1	0.5	BMI2
BZHI		1 1	2	p15 p23	ı	0.5	BMI2
PDEP	r,m,r	1	1		3	1	BMI2
	r,r,r		2	p1	3		
PDEP	r,r,m	1		p1 p23	_	1	BMI2
PEXT	r,r,r	1	1	p1	3	1	BMI2
PEXT	r,r,m	1	2	p1 p23		1	BMI2
Control transfer	instructions						
JMP	short/near	1	1	p6		1-2	
JMP	r	1	1	p6		2	
JMP	m	1	2	p23 p6		2	
Conditional jump	short/near	1	1	p6		1-2	predicted taken
Conditional jump	short/near	1	1	p06		0.5-1	predicted not taken
Fused arithmetic and branch		1	1	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	, conton
LOOP	short	7	7	ρο 100 ρο		5	
LOOP(N)E	short	11	11			6	
CALL	near	2	3	p237 p4 p6		3	
CALL	r	2	3	p237 p4 p6		2	
CALL	m '	3	4	2p237 p4 p6		3	
RET	""	1	2	p237 p4 p0		1	
RET	i		2	ρ237 ρ0		2	
BOUND	_	15	15			8	not 64 bit
INTO	r,m	5	5			6	not 64 bit
INTO		5	5			0	HOL 64 DIL
String instruc- tions							
LODSB/W	1	3	3	2p0156 p23		1	
LODSD/Q		2	2	p0156 p23		1	
REP LODS		5n+12				~2n	
STOS		3	3	p23 p0156 p4		1	
REP STOS		<2n				~0.5n	worst case
REP STOS		2.6/32B				1/32B	best case
							aligned by 32
MOVS		5	5	2p23 p4 2p0156		4	
REP MOVS		~2n		' '		< 1n	worst case
REP MOVS		4/32B				1/32B	best case
							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		1	0	none		0.25	
1F)			_	_			
PAUSE	_	4	4	p6		_	
ENTER	a,0	12	12			8	
ENTER	a,b	~14+7b	~45+7b		~87+2b		
LEAVE		3	3	2p0156 p23		5	
XGETBV		15	15			9	XGETBV
RDTSC		20	20			25	
RDTSCP		22	22			32	RDTSCP
RDPMC		35	35			40	
RDRAND	r	16	16	p23 15p0156		~460	RDRAND
RDSEED	r	16	16	p23 15p0156		~460	RDSEED

## Floating point x87 instructions

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

FLDPI FLDL2E etc.       2       2       2p05         FCMOVcc       r       4       4       p0 p1 p56         FNSTSW       AX       2       2       p0 p0156	3 2
	3 2
	6 2
FNSTSW m16 2 3 p0 p4 p237	6 1
FLDCW m16 3 3 p01 p23 p6	7 2
FNSTCW m16 2 3 p237 p4 p6	6 1
FINCSTP FDECSTP 1 1 p05	0 0.5
FFREE(P) r 1 1 p05	0.5
FNSAVE m 133 133	176 176
	175 175
Arithmetic in- structions	
FADD(P)	
FSUB(R)(P) r 1 1 p5	3 1
FADD(P)	1
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
	4-16 4-5
FDIV(R)(P) m 1 2 p0 p23	4-5
FABS 1 1 p0	1 1
FCHS 1 1 p0	1 1
FCOM(P) FUCOM r 1 1 p5	3 1
FCOM(P) FUCOM m 1 2 p5 p23	1
FCOMPP FUCOMPP 2 2 p0 p5	1
FCOMI(P)	'
	1
FIADD FISUB(R) m 3 4 2p5 p23	2
FIMUL m 2 3 p0 p5 p23	1
FIDIV(R) m 2 3 p0 p5 p23	
FICOM(P) m 2 3 2p5 p23	2
FTST 1 1 p5	3 1
FXAM 2 2 2p5	6 2
	6-30 17
	0-57 17
	21 11
Math	
	130   130
FXTRACT 17 17	11 11
FSQRT 1 1 p0 1	4-21 4-7
	0-120
	0-130
	5-150
	5-80
	103
	77
	0-160
FPATAN 30-160 10	0-160
Other	
FNOP 1 1 p05	0.5

WAIT	2	2	p05	2	
FNCLEX	5	5	p156	22	
FNINIT	18	18		78	

**Integer MMX and XMM instructions** 

Integer MMX a	ing Alvilvi ins	truction	1S			Recipro-	
		μοps fused	μορs unfused			cal through	
Instruction	Operands	domain		µops each port	Latency	put	Comments
Move instruc-							
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	p015	0-1	0.25	may eliminate
MOVDQA/U	x, m128		1 1	p23	2	0.23	Illay ellilillate
MOVDQA/U	m128, x	1	2	p237 p4	3	1	
		Ī -	1		0-1	1	
VMOVDQA/U	y,y	1		p015		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							
PACKUSWB	mm,mm	3	3	p5	2	2	
PACKSSWB/DW							
PACKUSWB	mm,m64	3	3	p23 2p5		2	
PACKSSWB/DW	,						
PACKUSWB	x,x / y,y,y	1	1	p5	1	1	
PACKSSWB/DW	7,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		-				
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
	A,111 / y,y,111	1		ρ23 ρ3			JOL4.1
PUNPCKH/L BW/WD/DQ	N. A. L. V. V. V.	1	1	n5	1	4	
	v,v / v,v,v	1	1	p5	'	1	
PUNPCKH/L				m00 = 5			
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	,			05 -			
QDQ	x,m / y,y,m	1	2	p23 p5		1	

Í	I		ı	- I I			l I
PMOVSX/ZX BW BD BQ DW DQ	x,x	1	1	p5	1	1	SSE4.1
PMOVSX/ZX BW BD BQ DW DQ	x,m	1	2	p23 p5		1	SSE4.1
VPMOVSX/ZX BW BD BQ DW DQ	y,x	1	1	p5	3	1	AVX2
VPMOVSX/ZX BW BD BQ DW DQ	y,m	2	2	p5 p23		1	AVX2
PSHUFB	v,v / v,v,v	1	1	p5	1	1	SSSE3
PSHUFB	v,m / v,v,m	2	2	p23 p5		1	SSSE3
PSHUFW	mm,mm,i	1	1	p5	1	1	
PSHUFW	mm,m64,i	2	2	p23 p5		1	
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
VPBROADCAST B/W	x,m8/16	2	2	p23 p5	7	1	AVX2

I	I	ı	I	I	I	1	1 1
VPBROADCAST D/Q	x,m32/64	1	1	p23	4	0.5	AVX2
VPBROADCAST	X,11132/04	!	!	μ25	7	0.5	AVAZ
B/W/D/Q	y,x	1	1	p5	3	1	AVX2
VPBROADCAST	),,,,		·				7.07.
B/W	y,m8/16	2	2	p23 p5	7	1	AVX2
VPBROADCAST	-						
D/Q	y,m32/64	1	1	p23	3	0.5	AVX2
VBROADCASTI128	y,m128	1	1	p23	3	0.5	AVX2
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	v,v / v,v,v	1	1	p015	1	0.33	
PADD/SUB(S,US) B/W/D/Q				-045 -00		0.5	
	v,m / v,v,m	1	2	p015 p23		0.5	
PHADD(S)W/D				· 04 0 · 5	_		00050
PHSUB(S)W/D	v,v / v,v,v	3	3	p01 2p5	3	2	SSSE3
PHADD(S)W/D				-04 0- F -00			00050
PHSUB(S)W/D	v,m / v,v,m	4	4	p01 2p5 p23		2	SSSE3
PCMPEQB/W/D		_	_	-0	_	_	
PCMPGTB/W/D	mm,mm	1	1	p0	1	1	
PCMPEQB/W/D		_	_	m04	_	0.5	
PCMPGTB/W/D	x,x / y,y,y	1	1	p01	1	0.5	
PCMPEQB/W/D		_	_	m01 m02		0.5	
PCMPGTB/W/D	x,m / y,y,m	1	2	p01 p23	_	0.5	00544
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	mm mm	4	4	20	_	4	
PMULHUW	mm,mm	1	1	p0	5	1	
PMULL/HW PMULHUW	X X / X X X	1	1	n01	5	0.5	
	x,x / y,y,y	1	l I	p01	) J	0.5	
PMULL/HW PMULHUW	v m / v v m	4	2	n01 n22		0.5	
PMULHRSW	x,m/y,y,m	1 1	1	p01 p23	_	1	SSSE3
PMULHRSW	mm,mm		1	p0	5 5	0.5	SSSE3
PMULHRSW	x,x / y,y,y		2	p01	) 5	0.5	SSSE3
PMULLD	x,m / y,y,m	1	2	p01 p23	10		
	x,x / y,y,y	2 3		2p01	10	1	SSE4.1 SSE4.1
PMULLD PMULDO	x,m / y,y,m	1	3	2p01 p23	_		1
PMULDQ BMULDQ	x,x / y,y,y		1	p01	5	0.5	SSE4.1
PMULDQ	x,m/y,y,m	1	2	p01 p23	F	0.5	SSE4.1
PMULUDQ	mm,mm	1	1	p0	5	1	
PMULUDQ	x,x / y,y,y	1	1	p01	5	0.5	
PMULUDQ	x,m / y,y,m	1	2	p01 p23		0.5	

				,				
PMADDWD	mm,mm	1	1	p0	5	1		
PMADDWD	x,x / y,y,y	1	1	p01	5	0.5		
PMADDWD	x,m / y,y,m	1	2	p01 p23		0.5		
PMADDUBSW	mm,mm	1	1	p0	5	1	SSSE3	
PMADDUBSW	x,x / y,y,y	1	1	p01	5	0.5	SSSE3	
PMADDUBSW	x,m / y,y,m	1	2	p01 p23		0.5	SSSE3	
PAVGB/W	mm,mm	1	1	p0	1	1	000_0	
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	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	•	_	po 1 p20		0.0		
SB/SW/SD								
UB/UW/UD	mm,mm	1	1	р0	1	1	SSE4.1	
PMIN/PMAX	,	•		<b>P</b> •	•	•	002	
SB/SW/SD								
UB/UW/UD	x,x / y,y,y	1	1	p01	1	0.5	SSE4.1	
PMIN/PMAX	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	•		ρο.	•	0.0	002	
SB/SW/SD								
UB/UW/UD	x,m / y,y,m	1	2	p01 p23		0.5	SSE4.1	
PHMINPOSUW	x,x x,x	1	1	p0 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 1	0.5	SSSE3	
PABSB/W/D	x,m / y,m	1	2	p01 p23	!	0.5	SSSE3	
PSIGNB/W/D	1	1	1		1	1	SSSE3	
PSIGNB/W/D	mm,mm	1	1	p0		0.5	SSSE3	
	x,x / y,y,y		2	p01	Į Į			
PSIGNB/W/D	x,m / y,y,m	1		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	00544	
MPSADBW	x,x,i / v,v,v,i	2	2	2p5	4	2	SSE4.1	
MPSADBW	x,m,i / v,v,m,i	3	3	2p5 p23		2	SSE4.1	
Logic instruc-								
tions	-							
PAND PANDN	mm mm	4	1	n05	1	0.5		
POR PXOR	mm,mm	1	1	p05	1	0.5		
PAND PANDN		4	_	-045		0.00		
POR PXOR	x,x / y,y,y	1	1	p015	1	0.33		
PAND PANDN	., ., ., ., ., ., ., ., ., ., ., ., ., .	4	2	m015 m02		0.5		
POR PXOR	v,m / v,v,m	1	2	p015 p23		0.5	00544	
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		4		0				
PSRAW/D/Q	mm,mm	1	1	р0	1	1		
PSLLW/D/Q								
PSRLW/D/Q		0		- 0 - 00				
PSRAW/D/Q	mm,m64	2	2	p0 p23		1		
PSLLW/D/Q								
PSRLW/D/Q		0		-04 <b>5</b>				
PSRAW/D/Q	x,x / v,v,x	2	2	p01 p5	1	1		
PSLLW/D/Q								
PSRLW/D/Q		0		m04 =00		0.5		
PSRAW/D/Q	x,m / v,v,m	2	2	p01 p23		0.5		

1		ı	I.	I	ı	I	l I
PSLLW/D/Q							
PSRLW/D/Q PSRAW/D/Q	mm,i	1	1	p0	1	1	
	111111,1	l	1	ρυ	'	I	
PSLLW/D/Q PSRLW/D/Q							
PSRAW/D/Q	x,i / y,y,i	1	1	p01	1	0.5	
VPSLLVD/Q	A,17 y,y,1	'	'	poi	'	0.5	
VPSRAVD							
VPSRLVD/Q	V,V,V	1	1	p01	1	0.5	AVX2
VPSLLVD/Q	.,.,.			μ.		0.0	7.17.
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-							
tions							
PCMPESTRI	x,x,i	8	8	6p05 2p16	12	4	SSE4.2
PCMPESTRI	x,m128,i	8	8	3p0 2p16 2p5 p23		4	SSE4.2
PCMPESTRM	x,x,i	9	9	3p0 2p16 4p5	9	5	SSE4.2
PCMPESTRM	x,m128,i	9	9	6p05 2p16 p23		5	SSE4.2
PCMPISTRI	x,x,i	3	3	3p0	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		_					
PCLMULQDQ	x,x,i	1	1	p5	7	1	CLMUL
PCLMULQDQ	x,m,i	2	2	p5 p23		1	CLMUL
AESDEC,							
AESDECLAST, AESENC,							
AESENC, AESENCLAST	V V	1	1	20	4	1	AES
AESDEC,	X,X	1	1	p0	4	I	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	,			_p. p		_	
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

						Recipro-		
		µops	µops			cal		
		fused	unfused			through		
Instruction	Operands	domain	domain	µops each port	Latency	put	Comments	

Move instruc- tions							
MOVAPS/D	x,x	1	1	p015	0-1	0.25	may eliminate
VMOVAPS/D	y,y	1	1	p015	0-1	0.25	may eliminate
MOVAPS/D MOVUPS/D	x,m128	1	1	p23	2	0.5	
VMOVAPS/D VMOVUPS/D	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	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	р0	2	1	
VMOVMSKPS/D	r32,y	1	1	р0	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	1	1	p015	1	1	SSE4.1
BLENDVPS/PD	x,m,xmm0	2	2	p015 p23		1	SSE4.1
VBLENDVPS/PD	V,V,V,V	2	2	2p015	2	1	AVX
VBLENDVPS/PD	v,v,m,v	3	3	2p015 p23		1	AVX
MOVDDUP	V,V	1	1	p5	1	1	SSE3
MOVDDUP	v,m	1	1	p23	3	0.5	SSE3
VBROADCASTSS	x,m32	1	1	p23	2	0.5	AVX
VBROADCASTSS	y,m32	1	1	p23	3	0.5	AVX
VBROADCASTSS	x,x	1	1	p5	1	1	AVX2
VBROADCASTSS	y,x	1	1	p5	3	1	AVX2
VBROADCASTSD	y,m64	1	1	p23	3	0.5	AVX
VBROADCASTSD	y,x	1	1	p5	3	1	AVX2
VBROADCASTF128	y,m128	1	1	p23	3	0.5	AVX
MOVSH/LDUP	V,V	1	1	p5	1	1	SSE3

MOVSHILDUP   V,m	MOVOLUL BUD		l a	۱ ،	- 00		0.5	0050
UNPCKHILPSID	MOVSH/LDUP	v,m	1	1	p23	3	0.5	SSE3
EXTRACTPS   m32,xi   2   2   p0 p5						1		
EXTRACTPS				1			1	
VEXTRACTF128	EXTRACTPS	r32,x,i			p0 p5		1	
NEXTRACTF128	EXTRACTPS	m32,x,i	2	3	p4 p5 p23	5	1	SSE4.1
NEXTRACTF128	VEXTRACTF128	x,y,i	1	1	p5	3	1	AVX
NSERTPS	VEXTRACTF128	-	2	2	p23 p4	6	1	AVX
INSERTPS	INSERTPS	•	1			1	1	SSE4.1
VINSERTF128								
VINSERTF128							-	
VMASKMOVPS/ID         V,V,m         2         2         p015 p23         3         0.5         AVX           VMASKMOVPS/ID         m128,xx         4         4         p0 p4 p23         13         1         AVX           VMASKMOVPS/ID         vp6ATHERDPS         x,[r+s*x],x         4         4         p0 p4 p23         13         1         AVX           VPGATHERDPS         x,[r+s*x],x         4         4         p0 p1 p23 p5         12         4         AVX2           VPGATHERDPS         x,[r+s*x],x         5         5         p0 p1 p23 p5         12         AVX2           VPGATHERDPS         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         4         4         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHEROPD         x,[r+s*x],x         5					i i		· ·	
VMASKMOVPS/D         m128,xx         4         4         p0 p4 p23         13         1         AVX           VMASKMOVPS/D         m256,y,y         4         4         p0 p4 p23         13         1         AVX           VPGATHERDPS         Y,[r+s*y],x         4         4         p0 p1 p23 p5         12         4         AVX2           VPGATHERDPS         X,[r+s*x],x         5         5         p0 p1 p23 p5         13         5         AVX2           VPGATHERDPS         X,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERDPD         X,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERDPD         X,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         X,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         X,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         X,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         X,[r+s*x],x         5         5				1				
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         x,[r+s*x],x         5         5         p0 p1 p23 p5         13         5         AVX2           VPGATHEROPS         x,[r+s*y],x         4         4         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHEROPD         x,[r+s*x],x         4         4         p0 p1 p23 p5         2         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHEROPD         x,[r+s*x],x         5         5         p0 p1 p23 p5<								
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*y],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERDPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERDPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERDPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x,[r+s*x],x         4         4         p0 p1 p23 p5         <								
VPGATHERDPS         y,[r+s*y],y         4         4         p0 p1 p23 p5         13         5         AVX2           VPGATHERQPS         x,[r+s*y],x         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         y,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         y,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         y,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         y,[r+s*x],x         4         4         p0 p1 p23 p5         4 <t< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></t<>								
VPGATHERQPS         X,[F+S*X],X         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPS         X,[F+S*X],X         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERDPD         Y,[F+S*X],X         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         X,[F+S*X],X         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         X,[F+S*X],X         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         X,[F+S*X],X         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         X,[F+S*X],X         5         5         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         X,[F+S*X],X         4         4         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         X,[F+S*X],X         4         4         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         X,[F+S*Y],X         4         4         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         X,[F+S*Y],X         4         4         p0 p1 p23 p5         1         AVX2 <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>								
VPGATHERQPS         x, [r+s*y], x         4         4         4         p0 p1 p23 p5         4         AVX2           VPGATHERDPD         x, [r+s*x], x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERDPD         y, [r+s*x], y         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x, [r+s*x], x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x, [r+s*x], x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x, [r+s*x], y         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x, [r+s*x], y         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x, [r+s*x], y         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x, [r+s*x], y         4         4         p0 p1 p23 p5         2         AVX2           VCVTPD2PS         x, m128         2         2         p01 p5         5         1         AVX2           VCVTPD2PS         x, m128         2         2         p01 p5         5 <td></td> <td></td> <td></td> <td></td> <td></td> <td>13</td> <td></td> <td></td>						13		
VPGATHERDPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],y         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x,[r+s*x],y         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],y         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],y         4         4         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*x],y         4         4         4         p0 p1 p23 p5         4         AVX2           VCVTPD2PS         x,m128         2         2         p01 p5 p23         1         AVX           VCVTD2PS         x,m256         2         3         p01 p5 p23         1         AVX           CVTPS2PD         x,x         2         2         p01 p5 p23         0.5         1	VPGATHERQPS				p0 p1 p23 p5			
VPGATHERDPD         y,[r+s*x],y         4         4         p0 p1 p23 p5         4         AVX2           VPGATHERQPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         x,[r+s*x],x         5         5         p0 p1 p23 p5         4         AVX2           CONVERSION         CVTPD2PS         x,x         2         2         p01 p5 p23         1           CVTPD2PS         x,m128         2         3         p01 p5 p23         1         AVX           VCVTPD2PS         x,m256         2         3         p01 p5 p23         1         AVX           CVTSD2SS         x,x         2         2         p01 p5 p23         1         AVX           CVTPS2PD         x,x         2         2         p01 p5 p23         1         AVX           CVTPS2PD         x,x         2         2         p01 p5 p23         0.5         1           CVTPS2PD         x,x         2         2         p01 p5 p23         0.5         1           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5         AVX           VCVTS2PD         y,x         2         2	VPGATHERQPS	x,[r+s*y],x	4	4	p0 p1 p23 p5			AVX2
VPGATHERQPD         X,[r+s*x],x         5         5         p0 p1 p23 p5         2         AVX2           VPGATHERQPD         y,[r+s*y],y         4         4         p0 p1 p23 p5         2         AVX2           Conversion         CVTPD2PS         x,x         2         2         p01 p5         5         1           CVTPD2PS         x,m128         2         3         p01 p5 p23         1         AVX           VCVTPD2PS         x,m256         2         3         p01 p5 p23         1         AVX           CVTSD2SS         x,m2         2         2         p01 p5         5         1         AVX           CVTSD2SS         x,m64         2         3         p01 p5 p23         1         AVX           CVTPS2PD         x,x         2         2         p01 p5         5         1           CVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           CVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           CVTS2S2D         x,x         2         2         p01 p5 p23         0.5         AVX           CVTS2S2D         x,m128	VPGATHERDPD	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
CONVERSION         X,X         2         2         p01 p5 p23 p5         4         AVX2           COTPD2PS         X,X         2         2         p01 p5 p23         1         AVX           CVTPD2PS         X,M         2         2         p01 p5 p23         1         AVX           VCVTPD2PS         X,M         2         2         p01 p5 p23         1         AVX           VCVTPD2PS         X,m256         2         3         p01 p5 p23         1         AVX           CVTSD2SS         X,x         2         2         p01 p5         5         1           CVTPS2PD         X,x         2         2         p01 p5 p23         1         AVX           VCVTPS2PD         X,m64         2         3         p01 p5 p23         0.5         1           CVTPS2PD         X,m64         1         2         p01 p5 p23         0.5         0.5           VCVTPS2PD         Y,m128         1         2         p01 p5 p23         0.5         AVX           CVTPS2PD         Y,m128         1         2         p01 p5 p23         0.5         AVX           CVTS2SSD         X,xx         2         2         p01 p5 p23	VPGATHERDPD	y,[r+s*x],y	4	4	p0 p1 p23 p5		4	AVX2
CONVERSION         X,X         2         2         p01 p5 p23 p5         4         AVX2           COTPD2PS         X,X         2         2         p01 p5 p23         1         AVX           VCVTPD2PS         X,M128         2         3         p01 p5 p23         1         AVX           VCVTPD2PS         X,M256         2         3         p01 p5 p23         1         AVX           CVTSD2SS         X,M256         2         3         p01 p5 p23         1         AVX           CVTSD2SS         X,M64         2         3         p01 p5 p23         1         AVX           CVTPS2PD         X,M64         2         3         p01 p5 p23         0.5         1           CVTPS2PD         X,M64         1         2         p01 p5 p23         0.5         1           CVTPS2PD         X,M64         1         2         p01 p5 p23         0.5         AVX           VCVTPS2PD         Y,M128         1         2         p01 p5 p23         0.5         AVX           CVTS2SDD         X,X         2         2         p01 p5 p23         0.5         AVX           CVTS2SSD         X,M32         1         2         p01 p5 p2	VPGATHERQPD	x,[r+s*x],x	5	5	p0 p1 p23 p5		2	AVX2
Conversion	VPGATHERQPD		4	4			4	AVX2
CVTPD2PS         x,x         2         2         p01 p5 p23         1           CVTPD2PS         x,m128         2         3         p01 p5 p23         1           VCVTPD2PS         x,y         2         2         p01 p5 p23         1           VCVTPD2PS         x,m256         2         3         p01 p5 p23         1           CVTSD2SS         x,x         2         2         p01 p5 p23         1           CVTSD2SS         x,m64         2         3         p01 p5 p23         1           CVTPS2PD         x,x         2         2         p01 p5 p23         0.5           CVTPS2PD         x,m64         1         2         p01 p5 p23         0.5           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           VCVTS2S2D         x,x         2         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p5 p23         0.		371 3173			' ' ' '			
CVTPD2PS         x,x         2         2         p01 p5 p23         1           CVTPD2PS         x,m128         2         3         p01 p5 p23         1           VCVTPD2PS         x,y         2         2         p01 p5 p23         1           VCVTPD2PS         x,m256         2         3         p01 p5 p23         1           CVTSD2SS         x,x         2         2         p01 p5 p23         1           CVTSD2SS         x,m64         2         3         p01 p5 p23         1           CVTPS2PD         x,x         2         2         p01 p5 p23         0.5           CVTPS2PD         x,m64         1         2         p01 p5 p23         0.5           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           VCVTS2S2D         x,x         2         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p5 p23         0.								
CVTPD2PS         x,x         2         2         p01 p5 p23         1           CVTPD2PS         x,m128         2         3         p01 p5 p23         1           VCVTPD2PS         x,y         2         2         p01 p5 p23         1           VCVTPD2PS         x,m256         2         3         p01 p5 p23         1           CVTSD2SS         x,x         2         2         p01 p5 p23         1           CVTSD2SS         x,m64         2         3         p01 p5 p23         1           CVTPS2PD         x,x         2         2         p01 p5 p23         0.5           CVTPS2PD         x,m64         1         2         p01 p5 p23         0.5           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           VCVTS2S2D         x,x         2         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p5 p23         0.	Conversion							
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 p23         1         CVTSD2SS         1         CVTSD2FD         1         CVTSD2FD         1         CVTSD2PD         2         p01 p5 p23         0.5		<b>Y Y</b>	2	2	n01 n5	5	1	
VCVTPD2PS         x,y         2         2         p1 p5 p23         7         1         AVX           VCVTPD2PS         x,m256         2         3         p01 p5 p23         1         AVX           CVTSD2SS         x,x         2         2         p01 p5 p23         1         AVX           CVTPS2PD         x,x         2         2         p01 p5 p23         1         CCVTPS2PD           CVTPS2PD         x,m64         1         2         p01 p5 p23         0.5         D.5           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5         D.5           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5         D.5           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           VCVTS2S2D         x,x         2         2         p01 p5 p23         2         2           CVTDQ2PS         x,x         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,x         1         1         p01 p5 p23         2         2           CVTDQ2PS         x,x         1         1         p01 p5				1				
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         1           CVTPS2PD         x,x         2         2         p01 p5 p23         0.5         1           CVTPS2PD         x,m64         1         2         p01 p5 p23         0.5         0.5           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5         0.5           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           CVTSS2SD         x,x         2         2         p01 p5 p23         2         2           CVTSS2SD         x,m32         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p23         0.5         AVX           VCVTQ2PS         y,m256         1         2         p01 p23         0.5 <td></td> <td></td> <td></td> <td></td> <td></td> <td>7</td> <td>-</td> <td><b>^</b>\/<i>/</i></td>						7	-	<b>^</b> \/ <i>/</i>
CVTSD2SS         x,x         2         2         p01 p5         5         1           CVTSD2SS         x,m64         2         3         p01 p5 p23         1         1           CVTPS2PD         x,x         2         2         p01 p5 p23         0.5         0.5           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5         AVX           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           CVTSS2SD         x,x         2         2         p01 p5 p23         0.5         AVX           CVTSS2SD         x,x         2         2         p01 p5 p23         2         2           CVTSS2SD         x,x         2         2         p01 p5 p23         2         2           CVTDQ2PS         x,m32         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,x         1         1         p01 p4         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) P82DQ         x,x         1         1         p01 p23         0.5						,	-	
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         VCVTPS2PD           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5         AVX           CVTSS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           CVTSS2SD         x,x         2         2         p01 p5 p23         2         2           CVTDQ2PS         x,m32         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,m2         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p23         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,x         1         1         p01 p23         0.5         AVX           VCVTDQ2PD         x,x         2         2         p01 p23         0.5         A						_		AVA
CVTPS2PD         x,x         2         2         p01 p5 p23         0.5           VCVTPS2PD         y,x         2         2         p01 p5 p23         0.5           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           CVTSS2SD         x,x         2         2         p01 p5 p23         2         2           CVTDQ2PS         x,m32         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p23         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,x         1         1         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX				1		5		
CVTPS2PD         x,m64         1         2         p01 p5 p23         0.5           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 p23         2         2           CVTSS2SD         x,m32         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,x         1         1         p01         4         0.5         4           CVTDQ2PS         x,m128         1         2         p01 p23         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2 <t< td=""><td></td><td></td><td></td><td></td><td></td><td>_</td><td></td><td></td></t<>						_		
VCVTPS2PD         y,x         2         2         p01 p5 p23         7         1         AVX           VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           CVTSS2SD         x,x         2         2         p01 p5 p23         2         2           CVTDQ2PS         x,m32         1         2         p01 p5 p23         2         2           CVTDQ2PS         x,m128         1         2         p01 p23         0.5         0.5           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         x,x         1         1         p01         4         0.5         AVX           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,y         1         1         p01         4         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p5         5         1           CVTDQ2PD         y,x         2						5	· •	
VCVTPS2PD         y,m128         1         2         p01 p5 p23         0.5         AVX           CVTSS2SD         x,x         2         2         p01 p5         5         2           CVTDQ2PS         x,x         1         1         p01 p23         2           CVTDQ2PS         x,m128         1         2         p01 p23         0.5           VCVTDQ2PS         y,y         1         1         p01 p23         0.5           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5           CVT(T) PS2DQ         x,x         1         1         p01 p2         0.5           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5           VCVT(T) PS2DQ         y,y         1         1         p01 p23         0.5           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5           VCVTDQ2PD         x,x         2         2         p01 p2         0.5      <			· -	1				
CVTSS2SD         x,x         2         2         p01 p5         5         2           CVTSS2SD         x,m32         1         2         p01 p5 p23         2           CVTDQ2PS         x,x         1         1         p01         4         0.5           CVTDQ2PS         x,m128         1         2         p01 p23         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         x,x         1         1         p01         4         0.5         AVX           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p5         5         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p5 <td></td> <td>-</td> <td></td> <td></td> <td></td> <td>7</td> <td></td> <td></td>		-				7		
CVTSS2SD         x,m32         1         2         p01 p5 p23         2           CVTDQ2PS         x,x         1         1         p01         4         0.5           CVTDQ2PS         x,m128         1         2         p01 p23         0.5           VCVTDQ2PS         y,y         1         1         p01         4         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,x         1         1         p01         4         0.5         AVX           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p23         0.5         AVX           CVTDQ2PD         x,m64         2         2         p01 p23         0.5         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           VCVT(T)PD2DQ         x,x         2         2         p01 p23         0.5		y,m128						AVX
CVTDQ2PS         x,x         1         1         p01         4         0.5           CVTDQ2PS         x,m128         1         2         p01 p23         0.5           VCVTDQ2PS         y,y         1         1         p01         4         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,x         1         1         p01         4         0.5         AVX           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,y         1         1         p01         4         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           VCVTDQ2PD         x,x         2         2         p01 p5         5         1           CVTDQ2PD         y,x         2         2         p01 p23         0.5         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01		X,X	2		p01 p5	5		
CVTDQ2PS         x,m128         1         2         p01 p23         0.5           VCVTDQ2PS         y,y         1         1         p01         4         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,x         1         1         p01         4         0.5         AVX           CVT(T) PS2DQ         y,y         1         1         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           VCVTDQ2PD         x,x         2         2         p01 p5         5         1           CVTDQ2PD         x,m64         2         2         p01 p23         0.5         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           VCVT(T)PD2DQ         x,x         2         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5 <td>CVTSS2SD</td> <td>x,m32</td> <td>1</td> <td>2</td> <td>p01 p5 p23</td> <td></td> <td>2</td> <td></td>	CVTSS2SD	x,m32	1	2	p01 p5 p23		2	
VCVTDQ2PS         y,y         1         1         p01         4         0.5         AVX           VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,x         1         1         p01         4         0.5           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5           VCVT(T) PS2DQ         y,y         1         1         p01         4         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p5         5         1           CVTDQ2PD         y,m64         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p5         5         1           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1         AVX           VCVT(T)PD2DQ         x,y         2         2         p01	CVTDQ2PS	x,x	1	1	p01	4	0.5	
VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,x         1         1         p01         4         0.5           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p23         0.5         AVX           CVTDQ2PD         x,m64         2         2         p01 p23         0.5         AVX           VCVTDQ2PD         y,x         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p5         5         1           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1         AVX           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX	CVTDQ2PS	x,m128	1	2	p01 p23		0.5	
VCVTDQ2PS         y,m256         1         2         p01 p23         0.5         AVX           CVT(T) PS2DQ         x,x         1         1         p01         4         0.5           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p23         0.5         AVX           CVTDQ2PD         x,m64         2         2         p01 p23         0.5         0.5           VCVTDQ2PD         y,x         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1         AVX           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX	VCVTDQ2PS	<b>У</b> , У	1	1	p01	4	0.5	AVX
CVT(T) PS2DQ         x,x         1         1         p01         4         0.5           CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5           VCVT(T) PS2DQ         y,y         1         1         p01         4         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p23         0.5         AVX           CVTDQ2PD         x,m64         2         2         p01 p23         0.5         AVX           VCVTDQ2PD         y,x         2         2         p01 p23         0.5         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1         1           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX	VCVTDQ2PS		1	2	p01 p23		0.5	AVX
CVT(T) PS2DQ         x,m128         1         2         p01 p23         0.5           VCVT(T) PS2DQ         y,y         1         1         p01         4         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p5         5         1           CVTDQ2PD         x,m64         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,x         2         2         p01 p23         0.5         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1         1           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX		-	1			4		
VCVT(T) PS2DQ         y,y         1         1         p01         4         0.5         AVX           VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p5         5         1           CVTDQ2PD         x,m64         2         2         p01 p23         0.5           VCVTDQ2PD         y,x         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1         AVX           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX	1 ' '		1	2	· ·			
VCVT(T) PS2DQ         y,m256         1         2         p01 p23         0.5         AVX           CVTDQ2PD         x,x         2         2         p01 p5         5         1           CVTDQ2PD         x,m64         2         2         p01 p23         0.5           VCVTDQ2PD         y,x         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX			1			4		AVX
CVTDQ2PD         x,x         2         2         p01 p5         5         1           CVTDQ2PD         x,m64         2         2         p01 p23         0.5           VCVTDQ2PD         y,x         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1         1           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX			1		-	·		
CVTDQ2PD         x,m64         2         2         p01 p23         0.5           VCVTDQ2PD         y,x         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX		=				5		AVA
VCVTDQ2PD         y,x         2         2         p01 p5         7         1         AVX           VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX						J	· ·	
VCVTDQ2PD         y,m128         1         2         p01 p23         0.5         AVX           CVT(T)PD2DQ         x,x         2         2         p01 p5         5         1           CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX						7		A\/V
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		-				/	· ·	
CVT(T)PD2DQ         x,m128         3         3         p01 p23 p5         1           VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX		=				_		AVX
VCVT(T)PD2DQ         x,y         2         2         p01 p5         7         1         AVX						5		
VCVT(T)PD2DQ	` '					7		
	VCVT(T)PD2DQ	x,m256	2	3	p01 p23 p5		1	AVX

1	I		1	i		1 .	ı ı
CVTPI2PS	x,mm	2	2	p0 p1	6	2	
CVTPI2PS	x,m64	1	2	p01 p23		3	
CVT(T)PS2PI	mm,x	2	2	p0 p5	7	1	
CVT(T)PS2PI	mm,m128	2	2	p0 p23		1	
CVTPI2PD	x,mm	2	2	p01 p5	5	1	
CVTPI2PD	x,m64	1	2	p01 p23		0.5	
CVT(T) PD2PI	mm,x	2	2	p01 p5	5	1	
CVT(T) PD2PI	mm,m128	2	3	p01 p23 p5		1	
CVT(1)1 B211	· '	2	2		6	2	
	x,r32			p01 p5			
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		3	3	p01 p4 p23	3-7	1	F16C
	m,v,i	2	2		5-7		
VCVTPH2PS	V,X			p01 p5	5-7	1	F16C
VCVTPH2PS	v,m	1	2	p01 p23		1	F16C
Arithmetic							
ADDSS/D PS/D							
SUBSS/D PS/D	x,x / v,v,v	1	1	p01	4	0.5	
ADDSS/D PS/D							
SUBSS/D PS/D	x,m / v,v,m	1	2	p01 p23		0.5	
ADDSUBPS/D	x,x / v,v,v	1	1	p01	4	0.5	SSE3
ADDSUBPS/D	x,m / v,v,m	1	2	p01 p23		0.5	SSE3
HADDPS/D	, , ,						
HSUBPS/D	x,x / v,v,v	3	3	p01 2p5	6	2	SSE3
HADDPS/D	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,			po. <u>-</u> po		_	0020
HSUBPS/D	x,m / v,v,m	4	4	p1 2p5 p23		2	SSE3
MULSS/D PS/D		1	1		4	0.5	33L3
	x,x / v,v,v			p01	4		
MULSS/D PS/D	x,m / v,v,m	1	2	p01 p23	4.4	0.5	
DIVSS	X,X	1	1	p0	11	3	
DIVPS	X,X	1	1	p0	11	3	
DIVSS DIVPS	x,m	1	2	p0 p23		3-5	
DIVSD	X,X	1	1	p0	13-14	4	
DIVPD	X,X	1	1	p0	13-14	4	
DIVSD DIVPD	x,m	1	2	p0 p23		4	
VDIVPS	y,y,y	1	1	p0	11	5	AVX
VDIVPS	y,y,m256	1	2	p0 p23		5	AVX
VDIVPD	y,y,y	1	1	p0	13-14	8	AVX
VDIVPD	y,y,m256	4	4	p0 p23		8	AVX
RCPSS/PS	V,V	1	1	p0	4	1	
RCPSS/PS			2				
	v,m	1		p0 p23		I	
CMPccSS/D				-04		0.5	
CMPccPS/D	x,x / v,v,v	1	1	p01	4	0.5	
CMPccSS/D		_	_				
CMPccPS/D	x,m / v,v,m	2	2	p01 p23		0.5	
(U)COMISS/D	X,X	1	1	p0		1	

MAXSS/D PS/D MINSS/D PS/D V,v,i         2         2         2 p01 p23         0.5           ROUNDSS/D PS/D PS/D DPS         v,v,i         2         2         2 p01 p23         1         SSE4.1           ROUNDSS/D PS/D DPS         v,m,i         3         3         2p01 p23         1         SSE4.1           DPPS         x,x,i / v,v,v,i         4         4         3p01 p5         13         1.5         SSE4.1           DPPS         x,m,i / v,v,m,i         6         6         3p01 p23 p5 p6         1.5         SSE4.1           DPPD         x,x,i         3         3         2p01 p5         9         1         SSE4.1           DPPD         x,m128,i         4         4         2p01 p23 p5         1         SSE4.1           VFMADD (all FMA instr.)         v,v,v         1         1         p01 p23         0.5         FMA           VFMADD (all FMA instr.)         v,v,m         1         2         p01 p23         0.5         FMA					ylako			
MINSS/D PS/D	(U)COMISS/D	x,m32/64	2	2	p0 p23		1	
MINSS/D PS/D		x,x / v,v,v	1	1	p01	4	0.5	
MINSS/D PS/D	MAXSS/D PS/D							
ROUNDSS/D PS/D DPPS	MINSS/D PS/D	x,m / v,v,m	1	2	p01 p23		0.5	
DPPS         X,x,i / V,v,v,i         4         4         3p01 p5         13         1.5         SSE4.1           DPPD         x,x,i         3         3         3p01 p5         9         1.5         SSE4.1           DPPD         x,xi,i         3         3         2p01 p23 p5         1         SSE4.1           DPPD         x,m128,i         4         4         2p01 p23 p5         1         SSE4.1           VFMADD         (all FMA instr.)         V,v,v         1         1         p01 p23         0.5         FMA           Math         XPMADD         (all FMA instr.)         V,v,m         1         2         p01 p23         0.5         FMA           Math         XPMADD         (all FMA instr.)         V,v,m         1         2         p01 p23         0.5         FMA           Math         XPMADD         1         1         p0         12         3         3         V           VSQRTSS/PS         X,m128         1         2         p0 p23         6         AVX         A	ROUNDSS/D PS/D	v,v,i	2	2	2p01	8	1	SSE4.1
DPPS         x,m,i / v,v,m,i         6         6         3 p01 p23 p5 p6         1.5         SSE4.1           DPPD         x,x,i         3         3 p01 p23 p5         9         1         SSE4.1           DPPD         x,m128,i         4         4         2p01 p23 p5         9         1         SSE4.1           VFMADD (all FMA instr.)         v,v,v         1         1         p01 p23 p5         0.5         FMA           Wemphy Math         xx         1         1         p0 p23         0.5         FMA           Math         xx         1         1         p0 p23         3         AVX           VSQRTPS         x,m128         1         2         p0 p23         3         AVX           VSQRTPS         y,m256         4         4         p0 p23         4-6         AVX           SQRTSD/PD         x,x         1         1         p0         15-16         4-6         AVX           VSQRTPD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         x,m12         1	ROUNDSS/D PS/D	v,m,i	3	3	2p01 p23		1	SSE4.1
DPPS         x,m,i / v,v,m,i         6         6         3 pot p23 p5 p6         1.5         SSE4.1           DPPD         x,x,i         3         3 pot p23 p5         9         1         SSE4.1           DPPD         x,m128,i         4         4         2p01 p23 p5         9         1         SSE4.1           VFMADD (all FMA instr.)         v,v,v         1         1         p01 p23         0.5         FMA           Math         Xx         1         1         p0 p23         0.5         FMA           Math         Xx         1         1         p0 p23         3         AVX           SQRTSS/PS         x,m128         1         2         p0 p23         3         AVX           VSQRTPS         y,m256         4         4         p0 p23         4-6         AVX           SQRTSD/PD         x,x         1         1         p0         15-16         4-6         AVX           VSQRTPD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         x,m128         1         2         p0 p23         4-1         AVX           VSQRTPD         x,m128         1         <	DPPS	x.x.i / v.v.v.i	4	4	3p01 p5	13	1.5	SSE4.1
DPPD								
DPPD						a		
VFMADD (all FMA instr.)         v,v,v         1         1         p01         4         0.5         FMA           Weynadd (all FMA instr.)         v,v,m         1         2         p01 p23         0.5         FMA           Math SQRTSS/PS SQRTSS/PS SQRTSS/PS VSQRTPS VSQRTPS VSQRTPS VSQRTPS VSQRTPD VXDRTPD VXDRTP						9		
(all FMA instr.)         v,v,v         1         1         p01         4         0.5         FMA           VFMADD (all FMA instr.)         v,v,m         1         2         p01 p23         0.5         FMA           Math         SQRTSS/PS         x,x         1         1         p0 p23         3         3           SQRTSS/PS         x,m128         1         2         p0 p23         3         3           VSQRTPS         y,y         1         1         p0 p23         6         AVX           VSQRTPS         y,m256         4         4         p0 p23         6         AVX           SQRTSD/PD         x,x         1         1         p0 p15-16         4-6         4-6         AVX           SQRTSD/PD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         y,m256         4         4         p0 p23         9-12         AVX           VSQRTSS/PS         y,v         1         1         p0         4         1         AVX           RSQRTSS/PS         v,v         1         1         p0 p23         9-12         AVX           VSQRTPD         y,m256		X,111120,1	4	4	2p0 1 p23 p3		ı	33E4.1
Math   SQRTSS/PS   X,X	(all FMA instr.)	V,V,V	1	1	p01	4	0.5	FMA
Math         X,X         1         1         p0         12         3           SQRTSS/PS         x,m128         1         2         p0 p23         3         AVX           VSQRTPS         y,y         1         1         p0         12         6         AVX           VSQRTPS         y,y         1         1         p0         15-16         4-6         AVX           SQRTSD         x,x         1         1         p0         15-16         4-6         AVX           SQRTSD/PD         x,x         1         1         p0         15-16         4-6         AVX           VSQRTPD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         y,w         1         1         p0         15-16         9-12         AVX           VSQRTPD         y,w         1         1         p0 p23         4-6         40         AVX           VSQRTSS/PS         v,v         1         1         p0 p23         1         1         AVX         AVX           RSQRTSS/PS         v,w         1         1         p0 p1 p5 p6         1         1         AVX         AVX				_				
SQRTSS/PS         x,x         1         1         p0 p0 p23         3         AVX           VSQRTPS         y,y         1         1         p0 p0 p23         3         AVX           VSQRTPS         y,m256         4         4         p0 p0 p23         6         AVX           SQRTSD         x,x         1         1         p0         15-16         4-6         AVX           SQRTSD/PD         x,x         1         1         p0         15-16         4-6         AVX           SQRTSD/PD         x,x         1         1         p0         15-16         4-6         AVX           VSQRTPD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         y,m256         4         4         p0 p23         9-12         AVX           VSQRTSS/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         VZE	(all FMA instr.)	v,v,m	1	2	p01 p23		0.5	FMA
SQRTSS/PS         x,m128         1         2         p0 p23         3         AVX           VSQRTPS         y,y         1         1         p0 p23         6         AVX           VSQRTPS         y,m256         4         4         p0 p23         6         AVX           SQRTSD         x,x         1         1         p0         15-16         4-6         AVX           SQRTSD/PD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         y,m1286         4         4         p0 p23         4-6         AVX           VSQRTPD         y,m256         4         4         p0 p23         9-12         AVX           VSQRTPD         y,m256         4         4         p0 p23         1         AVX           VSQRTSS/PS         v,v         1         1         p0 p23         1         1           RSQRTSS/PS         v,m         1         2         p0 p23         1         1           Logic         x,x/y,v,v         1         1         p0 p23         0.5         0.5           Chee         x,x/y,v,v,v         1         1         AVX         AVX <td>Math</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>	Math							
SQRTSS/PS         x,m128         1         2         p0 p23         3         AVX           VSQRTPS         y,y         1         1         p0 p23         6         AVX           VSQRTPS         y,m256         4         4         p0 p23         6         AVX           SQRTSD         x,x         1         1         p0         15-16         4-6         AVX           SQRTSD/PD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         y,m1286         4         4         p0 p23         4-6         AVX           VSQRTPD         y,m256         4         4         p0 p23         9-12         AVX           VSQRTPD         y,m256         4         4         p0 p23         1         AVX           VSQRTSS/PS         v,v         1         1         p0 p23         1         1           RSQRTSS/PS         v,m         1         2         p0 p23         1         1           Logic         x,x/y,v,v         1         1         p0 p23         0.5         0.5           Chee         x,x/y,v,v,v         1         1         AVX         AVX <td>SQRTSS/PS</td> <td>x,x</td> <td>1</td> <td>1</td> <td>0q</td> <td>12</td> <td>3</td> <td></td>	SQRTSS/PS	x,x	1	1	0q	12	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         p1-16         4-6           VSQRTPD         y,m128         1         2         p0 p23         4-6         4-6           VSQRTPD         y,m256         4         4         p0 p23         9-12         AVX           RSQRTSS/PS         v,v         1         1         p0         4         1           RSQRTSS/PS         v,w         1         2         p0 p23         9-12         AVX           RSQRTSS/PS         v,w         1         1         p0         4         1         1           RSQRTSS/PS         v,w         1         1         p0 p23         9-12         AVX           RSQRTSS/PS         v,m         1         2         p0 p23         0.5         0.5           Logic           AND/ANDN/OR/XOR         x,x / v,v,v         1			1					
VSQRTPS         y,m256         4         4         p0 p23         6         AVX           SQRTSD         x,x         1         1         p0         15-16         4-6         AVX           SQRTSD/PD         x,x         1         1         p0         15-16         4-6         AVX           SQRTSD/PD         x,m128         1         2         p0 p23         4-6         AVX           VSQRTPD         y,ym256         4         4         p0 p23         9-12         AVX           VSQRTSS/PS         v,v         1         1         p0         4         1         AVX           RSQRTSS/PS         v,m         1         2         p0 p23         9-12         AVX           RSQRTSS/PS         v,w         1         1         p0         4         1         AVX           RSQRTSS/PS         v,m         1         2         p0 p23         1         1           Logic         AND/ANDNOR/XOR         x,x / v,v,w         1         1         p0 p15 p23         0.5         0.5           Other         VZEROUPER         4         4         none         1         AVX           VZEROALL <td< td=""><td></td><td></td><td>=</td><td></td><td></td><td>12</td><td></td><td>Δ\/X</td></td<>			=			12		Δ\/X
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         4         p0 p23         9-12         AVX           RSQRTSS/PS         v,v         1         1         p0 p4         1         1           RSQRTSS/PS         v,m         1         2         p0 p23         1         1           Logic         AND/ANDNOR/XOR         x,x / v,v,v         1         1         p015         1         0.33           AND/ANDNOR/XOR         x,m / v,v,m         1         2         p015 p23         0.5         0.5           Other           VZEROUPPER         4         4         none         1         AVX           VZEROALL         34         34         p0 p1 p5 p6         12         64 bit           LDMXCSR         m32         3         4         p0 p5 p6								
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,m         1         2         p0 p23         1         1           Logic           AND/ANDN/OR/XORPS/PD         x,x / v,v,v         1         1         p015         1         0.33           AND/ANDN/OR/XORPS/PD         x,m / v,v,m         1         2         p015 p23         0.5         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         3         5           VZEROALL         33         4         p0 p5 p6 p23         5         3         3         5 <td< td=""><td></td><td>•</td><td></td><td></td><td></td><td>15 16</td><td></td><td>AVA</td></td<>		•				15 16		AVA
SQRTSD/PD         x,m128         1         2         p0 p23         4-6         4-7         4-6         AVX           VSQRTPD         y,y         1         1         p0 p23         9-12         AVX           VSQRTSS/PS         v,v         1         1         p0 p23         9-12         AVX           RSQRTSS/PS         v,v         1         1         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         34         p0 p1 p5 p6         12         64 bit           VZEROALL         34         4         p0 p5 p6 p23         5         3           STMXCSR         m32         4         4         p0 p4 p6 p237         5         2           FXSAVE         m4096         106         78			•	1				
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 p0 p23         4         1           RSQRTSS/PS         v,m         1         2         p0 p23         4         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         34         p0 p1 p5 p6         12         64 bit           VZEROALL         33         4         p0 p5 p6 p23         5         3           STMXCSR         m32         3         4         p0 p5 p6 p23         5         3           FXSAVE         m4096         106         78         78         7			•			15-16		
VSQRTPD         y,m256         4         4         p0 p23         9-12         AVX           RSQRTSS/PS         v,v         1         1         p0 p23         4         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           STMXCSR         m32         4         4         p0 p5 p6 p23         5         3           FXSAVE         m4096         106         78         78         78         32 bit mode           FXRSTOR         m4096         105         76         76         76         32 bit mode           FXRSTOR         m4096         105         77         77         77         64 bit mode           XSAVE         XSAVE         304         107         107         107 </td <td></td> <td>· ·</td> <td>•</td> <td></td> <td></td> <td></td> <td></td> <td></td>		· ·	•					
RSQRTSS/PS RSQRTSS/PS RSQRTSS/PS RSQRTSS/PS RSQRTSS/PS RSQRTSS/PS V,m 1 2 p0 p23 4 1   Logic AND/ANDN/OR/XO RPS/PD X,x / v,v,v 1 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 VZEROALL 25 25 p0 p1 p5 p6 12 32 bit AVX, AVX, 32 bit AVX, VZEROALL LDMXCSR M32 3 STMXCSR M32 3 STMXCSR M32 3 STMXCSR M32 3 STMXCSR M4096 106 FXSAVE FXSAVE M4096 106 FXSSAVE M4096 107 FXRSTOR M4096 105 FXRSTOR M4096 105 FXRSTOR M4096 105 FXRSTOR M4096 105 FXRSTOR M4096 105 FXRSTOR M4096 106 FXRSTOR M4096 107 107 107 32 bit mode 64 bit mode 76 76 76 32 bit mode 64 64 64 64 bit mode 77 77 77 64 bit mode 77 77 77 64 bit mode 78 XSAVE 107 107 107 107 32 bit mode 64 bit mode 78 XSAVE 78 XSAVE 79 107 107 107 107 107 107 107 107 107 107			-		·	15-16		
RSQRTSS/PS		y,m256	4	4			9-12	AVX
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	RSQRTSS/PS	V,V	1		p0	4	1	
AND/ANDN/OR/XO   RPS/PD	RSQRTSS/PS	v,m	1	2	p0 p23		1	
AND/ANDN/OR/XO   RPS/PD	Lasia							
RPS/PD								
AND/ANDN/OR/XO RPS/PD			4	_	~01F	4	0.22	
Other         X,m / v,v,m         1         2         p015 p23         0.5           Other         VZEROUPPER         4         4         4         none         1         AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A		X,X / V,V,V	1	1	pu15	1	0.33	
Other         VZEROUPPER         4         4         4         none         1         AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A		v m / v v m	1	2	2015 222		0.5	
VZEROUPPER         4         4         none         1         AVX AVX, AVX, AVX, AVX, AVX, AVX, AVX, A	IN S/I B	X,111 / V,V,111	I		ρυ 15 μ25		0.5	
VZEROALL         25         25         p0 p1 p5 p6         12         AVX, 32 bit AVX, 64 bit AVX, 64 bit           VZEROALL 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           FXRSTOR         m4096         105         64         64         64 bit mode           FXRSTOR         m4096         121         77         77         64 bit mode           XSAVE         247         107         107         32 bit mode           XSAVE         304         107         107         64 bit mode           XRSTOR         257         257         122         122         32 bit mode								
VZEROALL         25         25         p0 p1 p5 p6         12         32 bit AVX, 64 bit           VZEROALL LDMXCSR         m32         4         4         p0 p5 p6 p23 p6 p23 p0 p4 p6 p237         5         3           STMXCSR         m32         3         4         p0 p4 p6 p237         5         2           FXSAVE         m4096         106 p7 p5 p6 p23 p6 p23 p6 p23 p0 p4 p6 p237         5         2         78 p0 p4 p6 p237         5         2           FXSAVE         m4096         136 p6 p6 p237         64 p6 p6 p237         5         2         78 p0 p4 p6 p237         78 p7 p8 p6 p23 p	VZEROUPPER		4	4	none		1	AVX
VZEROALL LDMXCSR         m32 m32         4 4 4 4 4 90 p5 p6 p23         5 5 3 5 78         3 3 5 5 78         3 3 2 5 78         3 3 2 5 78         3 3 2 5 78         3 3 2 78         3 3 2 3 3 4 78         3 3 2 3 3 4 6 4 6 4 6 4 6 4 6 4 6 4 6 4 6 4 6								AVX,
VZEROALL         34         34         90 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           FXRSTOR         m4096         105         76         76         32 bit mode           FXRSTOR         m4096         121         77         77         64 bit mode           XSAVE         247         107         107         32 bit mode           XSAVE         304         107         107         64 bit mode           XRSTOR         257         122         122         32 bit mode           XRSTOR         257         122         122         64 bit mode	VZEROALL		25	25	p0 p1 p5 p6		12	32 bit
VZEROALL         m32         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           FXRSTOR         m4096         105         76         76         32 bit mode           FXRSTOR         m4096         121         77         77         64 bit mode           XSAVE         247         107         107         32 bit mode           XSAVE         304         107         107         64 bit mode           XRSTOR         257         122         122         32 bit mode           XRSTOR         257         122         122         64 bit mode								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           FXRSTOR         m4096         105         76         76         32 bit mode           FXRSTOR         m4096         121         77         77         64 bit mode           XSAVE         247         107         107         32 bit mode           XSAVE         304         107         107         64 bit mode           XRSTOR         257         122         122         32 bit mode           XRSTOR         257         122         122         64 bit mode	VZEROALL		34	34	p0 p1 p5 p6		12	
STMXCSR         m32         3         4         p0 p4 p6 p237         5         2           FXSAVE         m4096         136         78         78         32 bit mode           FXRSTOR         m4096         105         76         76         32 bit mode           FXRSTOR         m4096         121         77         77         64 bit mode           XSAVE         247         107         107         32 bit mode           XSAVE         304         107         107         64 bit mode           XRSTOR         257         122         122         32 bit mode           XRSTOR         257         122         122         64 bit mode		m32				5		
FXSAVE       m4096       106       78       78       32 bit mode         FXSAVE       m4096       136       64       64       64 bit mode         FXRSTOR       m4096       105       76       76       32 bit mode         FXRSTOR       m4096       121       77       77       64 bit mode         XSAVE       247       107       107       32 bit mode         XSAVE       304       107       107       64 bit mode         XRSTOR       257       122       122       32 bit mode         XRSTOR       257       122       122       64 bit mode								
FXSAVE         m4096         136         64         64         64 bit mode           FXRSTOR         m4096         105         76         76         32 bit mode           FXRSTOR         m4096         121         77         77         64 bit mode           XSAVE         247         107         107         32 bit mode           XSAVE         304         107         107         64 bit mode           XRSTOR         257         122         122         32 bit mode           XRSTOR         257         122         122         64 bit mode					P. P. P. P. P. P. P. P. P. P. P. P. P.			32 hit mode
FXRSTOR         m4096         105         76         76         32 bit mode           FXRSTOR         m4096         121         77         77         64 bit mode           XSAVE         247         107         107         32 bit mode           XSAVE         304         107         107         64 bit mode           XRSTOR         257         122         122         32 bit mode           XRSTOR         257         122         122         64 bit mode								
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								
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								
XSAVE       304       107       107       64 bit mode         XRSTOR       257       122       122       32 bit mode         XRSTOR       257       122       122       64 bit mode		m4096						
XRSTOR         257         122         122         32 bit mode           XRSTOR         257         122         122         64 bit mode								
XRSTOR 257 122 64 bit mode								
XSAVEOPT m   168   74   74								64 bit mode
	XSAVEOPT	m	168			74	74	

## Intel Pentium 4

# List of instruction timings and uop breakdown

This list is measured for a Pentium 4, model 2. Timings for model 3 may be more like the values for P4E, listed on the next sheet

Explanation of column headings:

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

JNE. etc.

**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.

Number of µops issued from instruction decoder and stored in trace cache. µops:

Number of additional µops issued from microcode ROM. Microcode:

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

next dependent instruction starts in the same execution unit. The numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's, infinity and exceptions increase the delays. The latency of moves to and from memory cannot be measured accurately because of the problem with memory intermediates explained

above under "How the values were measured".

This number is added to the latency if the next dependent instruction is in a Additional latency:

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

ALU1.

This is also called issue latency. This value indicates the number of clock cy-Reciprocal throughput:

cles from the execution of an instruction begins to a subsequent independent

instruction can begin to execute in the same execution subunit. A value of

0.25 indicates 4 instructions per clock cycle in one thread.

The port through which each µop goes to an execution unit. Two independent Port:

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.

Indicates the compatibility of an instruction with other 80x86 family micropro-Instruction set

cessors. The instruction can execute on microprocessors that support the in-

struction set indicated.

### Integer instructions

Instruction	Operands		<b> </b>		<b>&gt;</b>	ਰ ਸ਼	T	т	S	=	z
instruction	Operanus	sdon	Microcode	Latency	Additional latency	Reciprocal through- put	Port	Execution unit	Subunit	Instruction set	Notes
			000	ncy	tior	pro		üti	) E	uct	S
			de		<u>a</u>	cal		on -		ion	
					late	₹		in in		se	
					nc	Juo					
						ļ ģ					
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	1	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		0/1	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		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	int,alu		386	
LEA	r,[r+r*i+i]	3	0	4	0.5-1		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	I	ı	1 .	ı	ı		İ	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, 1	aldo/ 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	aldo/ 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	≥ 4	0/ 1	aluo/ i		86	
NEG	r	1	0	0.5	0.5-1		0	alu0		86	
NEG	m '	3	0	0.0	0.0 1	≥ 3		aido		86	
AAA, AAS		4	27	90		- 0				86	
DAA, DAS		4	57	100						86	
AAD		4	10	22			1	int	fpmul	86	
AAM		4	22	56			1	int	fpdiv	86	
MUL, IMUL	r8/32	4	6	16	0	8	1	int	fpmul	86	
MUL, IMUL	r16	4	7	17	0	8	1	int	fpmul	86	
MUL, IMUL	m8/32	4	7-8	16	0	8	1	int	fpmul	86	
MUL, IMUL	m16	4	10	16	0	8	1	int	fpmul	86	
IMUL	r32,r	4	0	14	0	4.5	1	int	fpmul	386	
IMUL	r32,(r),i	4	0	14	0	4.5	1	int	fpmul	386	
IMUL	r16,r	4	5	16	0	9	1	int	fpmul	386	
IMUL	r16,r,i	4	5	15	0	8	1	int	fpmul	186	
IMUL	r16,m16	4	7	15	0	10	1	int	fpmul	386	
IMUL	r32,m32	4	0	14	0	8	1	int	fpmul	386	
IMUL	r,m,i	4	7	14	0	10	1	int	fpmul	186	
DIV	r8/m8	4	20	61	0	24	1	int	fpdiv	86	а
DIV	r16/m16	4	18	53	0	23	1	int	fpdiv	86	a
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	a
IDIV	r32/m32	4	20	50	0	23	1	int	fpdiv	386	a
CBW	.02/02	2	0	1	0.5-1		0	alu0	.pa.r	86	
CWD, CDQ		2	0	1	0.5-1		0/1	alu0/1		86	
CWDE		1	0	0.5	0.5-1		0	alu0		386	
Landa tana											
Logic instructions		4		0.5	0.5.4	0.5	_	٥٠٠١٠		00	
AND, OR, XOR	r,r	1	0	0.5			0	alu0		86	С
AND, OR, XOR	r,m	2	0	≥1	0.5-1					86	С
AND, OR, XOR	m,r	3	0	≥8	0.5.4	≥ 4	_	- I O		86	С
TEST	r,r	1	0	0.5	0.5-1		0	alu0		86	С
TEST	r,m	2	0	≥ 1	0.5-1			-1.0		86	С
NOT	r	1	0	0.5	0.5-1	0.5	0	alu0		86	

NOT m 4 0   ≥4	86	
SHL, SHR, SAR r,i 1 0 4 1 1 1 int mmxsh	186	
SHL, SHR, SAR r,CL 2 0 6 0 1 1 int mmxsh	86	d
ROL, ROR r,i 1 0 4 1 1 1 int mmxsh	186	d
ROL, ROR r,CL 2 0 6 0 1 1 int mmxsh	86	d
RCL, RCR r,1 1 0 4 1 1 1 int mmxsh	86	d
RCL, RCR r,i 4 15 16 0 15 1 int mmxsh	186	d
RCL, RCR r,CL 4 15 16 0 14 1 int mmxsh	86	d
SHL,SHR,SAR,ROL,		
ROR   m,i/CL   4   7-8   10   0   10   1   int   mmxsh	86	d
RCL, RCR m,1 4 7 10 0 10 1 int mmxsh	86	d
RCL, RCR m,i/CL 4 18 18-28 14 1 int mmxsh	86	d
SHLD, SHRD r,r,i/CL 4 14 14 0 14 1 int mmxsh	386	
SHLD, SHRD m,r,i/CL 4 18 14 0 14 1 int mmxsh	386	
BT r,i 3 0 4 0 2 1 int mmxsh	386	d
BT r,r 2 0 4 0 1 1 int mmxsh	386	d
BT m,i 4 0 4 0 2 1 int mmxsh	386	d
BT 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	386	
BSF, BSR	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	u
CLD 4 7 52 0 52	86	
STD 4 5 48 0 48	86	
CLI 4 5 35 35 35	86	
STI 4 12 43 43 43	86	
	00	
Control transfer instructions		
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           CALL         far         4         34         0         0         0         0	86 86	
	86	
	86	
CALL m(far) 4 38 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	86	
RETN 4 0 2 0 alu0 branch	86	
RETN	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	1110		287	
FSTSW	AX	6	0	11	0	3	1			287	
		_	_	' '	U	_	-				
FNSTSW	m16	4	4			6	0			87	
FNSTCW	m16	4	4			6	0			87	
FLDCW	m16	4	7	(3)		(8)	0,2			87	f
Arithmetic instructions											
FADD(P),FSUB(R)(P)	r	1	0	5	1	1	1	fp	add	87	
FADD,FSUB(R)		2	0	5	1		1	-	add	87	
` ` '	m 10		4		•	1 .	-	fp			
FIADD, FISUB(R)	m16	3		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	J,
FCHS		1	0	2	1	1	1	fp	misc	87	
FCOM(P), FUCOM(P)	r	1	0	2	0	1	1	fp	misc	87	
FCOM(P)	m	2	0	2	0	1	1	fp	misc	87	
FCOMPP, FUCOMPP	""	2	0	2	0	'	1	-	misc	87	
· ·	_		_				-	fp			
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	nops	Microcode	Latency	Additional latency	Reciprocal through-	Port	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	1	0	6	0	1	0, 1	mov			
MOVQ	mm,mm	-	0	2	1	2			ob:ft	mmx	
	xmm,xmm	1	-		'		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/	,		_								
QDQ	xmm,r/m	1	0	4	1	2	1	mmx	shift	sse2	а
PUNPCKLBW/WD/DQ/Q			0						- 1- 164		_
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	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	а
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	I .	ı	I .	

PMIN/MAXUB	r,r/m	1	0	2	1	1,2	1	mmx	alu	sse	a,j	ĺ
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	a	
Other												
EMMS		4	11	12		12	0			mmx		ĺ

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

In a transition			7		<b>&gt;</b>	70 T	77	т	m	=	7
Instruction	Operands	sdori	Microcode	Latency	Additional latency	Reciprocal through- put	Port	Execution unit	Subunit	Instruction set	Notes
Move instructions											
MOVAPS/D	r,r	1	0	6	0	1	0	mov		sse	
MOVAPS/D	r,m	1	0	≈ 7	0	1	2			sse	
MOVAPS/D	m,r	2	0	≈ 7		2	0			sse	
MOVUPS/D	r,r	1	0	6	0	1	0	mov		sse	
MOVUPS/D	r,m	4	0			2	2			sse	k
MOVUPS/D	m,r	4	6			8	0			sse	k
MOVSS	r,r	1	0	2	0	2	1	mmx	shift	sse	
MOVSD	r,r	1	0	2	1	2	1	mmx	shift	sse	
MOVSS, MOVSD	r,m	1	0	≈ 7	0	1	2			sse	
MOVSS, MOVSD	m,r	2	0			2	0			sse	
MOVHLPS	r,r	1	0	4	0	2	1	mmx	shift	sse	
MOVLHPS	r,r	1	0	2	0	2	1	mmx	shift	sse	
MOVHPS/D, MOVLPS/D											
	r,m	3	0			4	2			sse	
MOVHPS/D, MOVLPS/D							_				
MOVATEDO/E	m,r	2	0			2	0			sse	
MOVNTPS/D	m,r	2	0			4	0			sse/2	
MOVMSKPS/D	r32,r	2	0	6	1	3	1	fp	1 :6	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	

COVTPSZPD												
CVTPS2PD	Conversion											
CVTPD2PS         f,r/m         2         0         10         1         2         1         fp-mmx         sse2         a           CVTSD2SS         f,r/m         4         0         10         1         2         1         fp-mmx         sse2         a           CVTDQ2PS         f,r/m         1         0         4         1         2         1         fp         sse2         a           CVTQ2PD         f,r/m         1         0         4         1         2         1         fp         sse2         a           CVT(T)PS2DQ         f,r/m         2         0         9         1         2         1         fp-mmx         sse2         a           CVT(T)PD2DQ         f,r/m         2         0         9         1         2         1         fp-mmx         sse2         a           CVT(T)PS2PI         xmm,mm         4         0         11         1         5         1         fp-mmx         sse2         a           CVT(T)PS2PI         mm,xmm         3         0         1         3         1         fp-mmx         sse2         a           CVT(T)PS2PI         mm,xmm         3 <td></td> <td>r.r/m</td> <td>4</td> <td>0</td> <td>7</td> <td>1</td> <td>4</td> <td>1</td> <td>mmx</td> <td>shift</td> <td>sse2</td> <td>а</td>		r.r/m	4	0	7	1	4	1	mmx	shift	sse2	а
CVTSD2SS			2	0	10	1	2	1				
CVTSS2SD				_			_	-	'		_	а
CVTDQ2PS         r,r/m         1         0         4         1         2         1         fp         sse2         ase2         a           CVTDQ2PD         r,r/m         1         0         4         1         2         1         fp         sse2         a           CVT(T)PS2DQ         r,r/m         1         0         4         1         2         1         fp-mmx         sse2         a           CVT(T)PD2DQ         r,r/m         2         0         9         1         2         1         fp-mmx         sse2         a           CVT(T)PD2P1         xmm,,mm         4         0         11         1         5         1         fp-mmx         sse2         a           CVT(T)PD2P1         mm,xmm         3         0         11         1         3         0,1         fp-mmx         sse2         a           CVT(T)PD2P1         mm,xmm         3         0         11         1         3         0,1         fp-mmx         sse2         a           CVT(T)SD2SI         rasper mm,r32         3         0         10         1         3         1         fp-mmx         sse2         a							-					
CVTTQ2PD				_	_					Simil		
CVT(T)PS2DQ										sse2		u
CVT(T)PD2DQ         r,r/m         2         0         9         1         2         1         fp-mmx         sse2         a           CVTPI2PS         xmm,mm         4         0         10         1         4         1         mmx         sse2         a           CVT(T)PS2PI         xmm,xmm         3         0         7         0         2         0,1         fp-mmx         sse2         a           CVT(T)PD2PI         mm,xmm         3         0         11         1         3         0,1         fp-mmx         sse2         a           CVT(SLSSS         xmm,r32         3         0         10         1         3         1         fp-mmx         sse2         a           CVT(T)SD2SI         r32,xmm         2         0         8         1         2.5         1         fp         sse2         a           CVT(T)SD2SI         r32,xmm         2         0         8         1         2.5         1         fp         add         sse2         a           CVT(T)SD2SI         r,r/m         1         0         4         1         2         1         fp         add         sse2         a				_	_	-				3302		2
CVTPI2PS	1 1		1		· ·			-		ssa2		u
CVTPI2PD	` '			_	_	· ·		-		3302		2
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         sse2         a           CVT(T)SD2SI         r32,xmm         2         0         8         1         2.5         1         fp         sse2         a           CVT(T)SD2SI         r32,xmm         2         0         8         1         2.5         1         fp         add         sse2         a           Arithmetic         Arithmetic           ADDPS/D ADDSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           AULLPS/D MULSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           DIVSD         r,r/m         1         0         33         0				_			1			ssa2		u
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           CVT(T)SD2SI         r32,xmm         2         0         8         1         2.5         1         fp-mmx         sse2         a           CVT(T)SD2SI         r32,xmm         2         0         8         1         2.5         1         fp         sse2         a           Arithmetic         A         1         0         4         1         2         1         fp         add         sse         a           SUBPS/D SUBSS/D SUBSS/D SUBSS/D FL,r/m         1         0         4         1         2         1         fp         add         sse         a           BUIVSD SUBSS/D FL,r/m         1         0         4         1         2         1         fp         ddv         sse         a           DIVPS FLYM         1         0         39         0         39         1         fp         div         sse2         a,h <t< td=""><td></td><td>-</td><td> </td><td>_</td><td></td><td>-</td><td></td><td></td><td></td><td></td><td></td><td></td></t<>		-		_		-						
CVTSI2SS         xmm,r32         3         0         10         1         3         1         fp-mmx         sse         a           CVTSI2SD         xmm,r32         4         0         15         1         6         1         fp-mmx         sse2         a           CVT(T)SD2SI         r32,xmm         2         0         8         1         2.5         1         fp         sse2         a           Arithmetic           ADDPS/D ADDSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           AULPS/D MULSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           MULPS/D MULSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           DIVPS         r,r/m         1         0         6         1         2         1         fp         div         sse         a,h           DIVPD         r,r/m         1         0         38         0         38	1		_			-		-				
CVTSI2SD         xmm,r32         4         0         15         1         6         1         fp-mmx         sse2         a           CVT(T)SD2SI         r32,xmm         2         0         8         1         2.5         1         fp         sse2         a           Arithmetic         ADDPS/D ADDSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           SUBPS/D SUBSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           MULPS/D MULSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           DIVSS         r,r/m         1         0         6         6         1         2         1         fp         mul         sse         a           DIVSD         r,r/m         1         0         39         0         39         1         fp         div         sse2         a,h           RCPPS RCPSS         r,r/m         1         0         4         1         2	` '	·		_		· ·		-				
CVT(T)SD2SI		· ·		_		-						
CVT(T)SS2SI         r32,xmm         2         0         8         1         2.5         1         fp         sse         a           Arithmetic         ADDPS/D ADDSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           SUBPS/D SUBSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           MULPS/D MULSS/D         r,r/m         1         0         6         1         2         1         fp         add         sse         a           DIVPS         r,r/m         1         0         23         0         23         1         fp         div         sse         a,h           DIVPD         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         1         0         4         1         2		,		_		· ·	-			ssez	_	
Arithmetic	, ,					· ·		-				
ADDPS/D ADDSS/D SUBPS/D SUBSS/D SUBPS/D SUBSS/D N/r,r/m 1 0 4 1 2 1 fp add sse a MULPS/D MULSS/D N/r,r/m 1 0 4 1 2 1 fp add sse a MULPS/D MULSS/D N/r,r/m 1 0 6 1 2 1 fp mul sse a DIVSS N,r,r/m 1 0 33 0 23 1 fp div sse a,h DIVPS N,r,r/m 1 0 38 0 38 1 fp div sse2 a,h DIVPD N,r,r/m 1 0 69 0 69 1 fp div sse2 a,h DIVPD NAXSS/DMINPS/D MAXSS/DMINPS/D MAXSS/DMINPS/D MINSS/D CMPCCPS/D CMPCCSS/D CMPCCSS/D CMPCCSS/D CMPCSS/D T,r,r/m 1 0 4 1 2 1 fp add sse a  CMPCCPS/D CMPCCSS/D CMPC	CV1(1)55251	r32,xmm	2	U	8	1	2.5	1	тр		sse	а
SUBPS/D SUBSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           MULPS/D MULSS/D         r,r/m         1         0         6         1         2         1         fp         mul         sse         a           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           DIVPD         r,r/m         1         0         69         0         69         1         fp         div         sse2         a,h           DIVPD         r,r/m         1         0         69         0         69         1         fp         div         sse2         a,h           DIVPD         r,r/m         1         0         4         1         2         1         fp         div         sse2         a,h           MAXSS/D         mAXSS/DMINPS/D         r,r/m         1         0         4         1         2	Arithmetic											
SUBPS/D SUBSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           MULPS/D MULSS/D         r,r/m         1         0         6         1         2         1         fp         mul         sse         a           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           DIVPD         r,r/m         1         0         69         0         69         1         fp         div         sse2         a,h           DIVPD         r,r/m         1         0         69         0         69         1         fp         div         sse2         a,h           DIVPD         r,r/m         1         0         4         1         2         1         fp         div         sse2         a,h           MAXSS/D         mAXSS/DMINPS/D         r,r/m         1         0         4         1         2	ADDPS/D ADDSS/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         a           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           DIVPD         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           DIVPD         r,r/m         1         0         4         1         2         1         fp         div         sse2         a,h           DIVPD         r,r/m         1         0         4         1         2         1         fp         div         sse2         a,h           MXXSS/DMINPS/D         r,r/m         1         0         4         1         2         1			1	0	4	1		1		add	sse	а
DIVSS			1	0	6	1		1				а
DIVPS			1	0	23	0						
DIVSD				_		_		-				
DIVPD				_		_						
RCPPS RCPSS				-		_						
MAXPS/D MAXSS/DMINPS/D MINSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           CMPccPS/D CMPccSS/D CMPccSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           COMISS/D UCOMISS/D CMPS/D UCOMISS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           Logic ANDPS/D ANDNPS/D ORPS/D XORPS/D         r,r/m         1         0         2         1         2         1         mmmx         alu         sse         a           SQRTSD SQRTSS         r,r/m         1         0         23         0         23         1         fp         div         sse         a,h           SQRTSD         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           SQRTSD         r,r/m         1         0				0		-				<b></b>		
MAXSS/DMINPS/D MINSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           CMPccPS/D CMPccSS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           COMISS/D UCOMISS/D         r,r/m         1         0         4         1         2         1         fp         add         sse         a           Logic ANDPS/D ANDNPS/D ORPS/D XORPS/D         r,r/m         1         0         2         1         2         1         mm         alu         sse         a           Math SQRTSS         r,r/m         1         0         23         0         23         1         fp         div         sse         a,h           SQRTSS         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           SQRTSD         r,r/m         1         0         69         0		.,	_		-	•						
CMPccPS/D CMPccSS/D COMISS/D UCOMISS/D         r,r/m         1         0         4         1         2         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           Math SQRTSS         r,r/m         1         0         23         0         23         1         fp         div         sse         a,h           SQRTPS         r,r/m         1         0         23         0         23         1         fp         div         sse         a,h           SQRTSS         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           RSQRTPS         r,r/m         2         4         1												
CMPccSS/D         r,r/m         1         0         4         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         r,r/m         1         0         2         1         2         1         mmm         alu         sse         a           Math         SQRTSS         r,r/m         1         0         23         0         23         1         fp         div         sse         a,h           SQRTSS         r,r/m         1         0         23         0         23         1         fp         div         sse         a,h           SQRTSS         r,r/m         1         0         39         0         39         1         fp         div         sse2         a,h           SQRTSD         r,r/m         1         0         38         0         38         1         fp         div         sse2         a,h           SQRTSS         r,r/m         1         0         69	MINSS/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         a           Logic         ANDPS/D ANDNPS/D ORPS/D ORPS/D XORPS/D         r,r/m         1         0         2         1         2         1         mmm         a           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         mm         4         8         98         100         1         mmx         sse         a           Other         LDMXCSR<	CMPccPS/D											
Logic         ANDPS/D ANDNPS/D ORPS/D XORPS/D         r,r/m         1         0         2         1         2         1         mmx         alu         sse         a           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         a           DMXCSR         m         4         8         98         100         1	CMPccSS/D	r,r/m	1	0	4	1	2	1	fp	add	sse	а
ANDPS/D ANDNPS/D ORPS/D XORPS/D         r,r/m         1         0         2         1         2         1         mmx         alu         sse         a           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         a           DMXCSR         m         4         8         98         100         1         sse	COMISS/D UCOMISS/D	r,r/m	2	0	6	1	3	1	fp	add	sse	а
ANDPS/D ANDNPS/D ORPS/D XORPS/D         r,r/m         1         0         2         1         2         1         mmx         alu         sse         a           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         a           DMXCSR         m         4         8         98         100         1         sse												
Math         r,r/m         1         0         2         1         2         1         mmx         alu         sse         a           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         a           DMXCSR         m         4         8         98         100         1         sse         sse           STMXCSR												
Math         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         a           Other         DMXCSR         m         4         8         98         100         1         sse         sse           STMXCSR         m         4         4         6         1         sse         sse		_										
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         a           Other         LDMXCSR         m         4         8         98         100         1         sse         sse           STMXCSR         m         4         4         6         1         sse         sse	ORPS/D XORPS/D	r,r/m	1	0	2	1	2	1	mmx	alu	sse	а
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         a           Other         LDMXCSR         m         4         8         98         100         1         sse         sse           STMXCSR         m         4         4         6         1         sse         sse	Math											
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         a           Other         LDMXCSR         m         4         8         98         100         1         sse         sse           STMXCSR         m         4         4         6         1         sse         sse		r.r/m	1	0	23	0	23	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         a           Other           LDMXCSR         m         4         8         98         100         1         sse         sse           STMXCSR         m         4         4         6         1         sse         sse				_		_						
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         a           Other           LDMXCSR         m         4         8         98         100         1         sse         sse           STMXCSR         m         4         4         6         1         sse         sse				_		_		-				
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         a           Other         DMXCSR         m         4         8         98         100         1         sse           STMXCSR         m         4         4         6         1         sse				Ĭ.		-						
RSQRTPS         r,r/m         2         0         4         1         4         1         mmx         sse         a           Other         LDMXCSR         m         4         8         98         100         1         sse           STMXCSR         m         4         4         6         1         sse				_		-		-				
Other         m         4         8         98         100         1         sse           STMXCSR         m         4         4         6         1         sse						-						
LDMXCSR         m         4         8         98         100         1         sse           STMXCSR         m         4         4         6         1         sse												
STMXCSR         m         4         4         6         1         sse												
					98						sse	
		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.

**Reciprocal**This 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	
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			~.3⊓ ≈.5-1.1r	1	4n			86			
ILLE INICAGA		1	۱۱ .۱۱ -ن.ح	U- I	.411	1 1		00			

								i			1	
REP MOVSD		1	≈1.1n	≈ 1.4	n			86				
REP MOVSQ		1	≈1.1n	≈ 1.4				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,aid		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	С	
INC, DEC	r	2	0	1	0	0.5	0/1	alu0/1		86		
INC, DEC		4	0	5	U	3	0/1	aluo/ i		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		
AAA, AAS											m	
DAA, DAS		1	16	29			_	:4		86	m	
AAD		2	5	13			1	int	mul	86	m	
AAM		2	17	71			1	int	fpdiv	86	m	
MUL, IMUL	r8	1	0	10	0		1	int	mul	86		
MUL, IMUL	r16	4	0	11	0		1	int	mul	86		
MUL, IMUL	r32	3	0	11	0		1	int	mul	86		
MUL, IMUL	r64	1	5	11	0		1	int	mul	x64		
MUL, IMUL	m8	2	0	10	0		1	int	mul	86		
MUL, IMUL	m16	2	5	11	0		1	int	mul	86		
MUL, IMUL	m32	3	0	11	0		1	int	mul	86		
MUL, IMUL	m64	2	6	11	0		1	int	mul	x64		
IMUL	r16,r16	1	0	10	0	2.5	1	int	mul	386		
IMUL	r16,r16,i	2	0	11	0	2.5	1	int	mul	186		
IMUL	r32,r32	1	0	10	0	2.5	1	int	mul	386		
IMUL	r32,(r32),i	1	0	10	0	2.5	1	int	mul	386		
IMUL	r64,r64	1	0	10	0	2.5	1	int	mul	x64		
IMUL	r64,(r64),i	1	0	10	0	2.5	1	int	mul	x64		
IMUL	r16,m16	2	0	10	0	2.5	1	int	mul	386		
IMUL	r32,m32	2	0	10	0	2.5	1	int	mul	386		
IMUL	r64,m64	2	0	10	0	2.5	1	int	mul	x64		
IMUL	r,m,i	3	0	10	0	1-2.5	1	int	mul	186		
DIV	r8/m8	1	20	74	0	34	1	int	fpdiv	86	а	
DIV	r16/m16	1	19	73	0	34	1	int	fpdiv	86	а	
DIV	r32/m32	1	21	76	0	34	1	int	fpdiv	386	а	
DIV	r64/m64	1	31	63	0	52	1	int	fpdiv	x64	а	

IDIV 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 2 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 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 86 386 x64 86 386 x64 386 x64 86	a a a
Logic AND, OR, XOR	rr	1	0	1	0	0.5	0	alu0		86	
1	r,r				_		0	aiuU			С
AND, OR, XOR	r,m	2	0	1	0	1				86	С
AND, OR, XOR	m,r	3	0	5	_	2		-10		86	С
TEST	r,r	1	0	1	0	0.5	0	alu0		86	С
TEST	r,m	2	0	1	0	1				86	С
NOT	r	1	0	1	0	0.5	0	alu0		86	
NOT	m	3	0	5		2				86	
SHL	r,i	1	0	1	0	0.5	1	alu1		186	
SHR, SAR	r8/16/32,i	1	0	1	0	0.5	1	alu1		186	
SHR, SAR	r64,i	1	0	7	0	2	1	alu1		x64	
SHL	r,CL	2	0	2	0	2	1	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, BTS, BTC BTR, BTS, BTC BTR, BTS, BTC BTR, BTS, BTC BTR, BTS, BTC CBT	r64,r64,cl m,r,i m,r,CL r,i 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 2 2 3 2 3 3 3 3 3 3 3 3 3	8 8 8 0 0 7 0 0 6 10 0 0	12 20 20 8 9 8 10 8 9 28 14 16 9 9		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 d
CLD, STD		1	8	. •	0	53				86	
Control transfer instruction	short/near	1	0	0	0	1	0	alu0	branch	86	
JMP	far	2	25	U	U	154	0	aluu	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	aido	Diamon.	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	
BOUND	m	2	11			12				186	m
INT	i	2	67			470				86	
INTO		1	4			26				86	m
Other											
NOP (90)		1	0	0		0.25	0/1	alu0/1		86	
Long NOP (0F 1F)		1	0	0		0.25		alu0/1		ppro	
PAUSE		1	2			50	0. 1	G.GO/ 1		sse2	
LEAVE		4	0	5		5				186	
CLI		1	5			52				86	
STI		1	11			64				86	
CPUID		1	49-90	Ī	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				(sse3)	
MWAIT				(sse3)	

Notes:

p)

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.

LEA with a direct memory operand has 1  $\mu$ op 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  $\mu$ 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	sdon	Microcode	Latency	Additional latency	Reciproca put	Port	Execution unit	Subunit	Instruction	Notes
					latency	Reciprocal through- put		unit		n set	
Move instructions											
FLD	r	1	0	7	0	1	0	mov		87	
FLD	m32/64	1	0		0	1	2	load		87	
FLD	m80	3	3			8	2	load		87	
FBLD	m80	3	74			90	2	load		87	
FST(P)	r	1	0	7	0	1	0	mov		87	
FST(P)	m32/64	2	0	7		2	0	store		87	
FSTP	m80	3	6			10	0	store		87	
FBSTP	m80	3	311			400	0	store		87	
FXCH	r	1	0	0	0	1	0	mov		87	
FILD	m16	3	2			8	2	load		87	
FILD	m32/64	2	0			2	2	load		87	
FIST(P)	m	3	0			2.5	0	store		87	
FISTTP	m	3	0			2.5	0	store		sse3	
FLDZ		1	0			2	0	mov		87	

FLD1		2	0			2	0	mov		87		
FCMOVcc	st0,r	4	0	5	1	4	1	fp		PPro	е	
FFREE	r	3	0			3	0	mov		87		
FINCSTP, FDECSTP		1	0	0	0	1	0	mov		87		
FNSTSW	AX	4	0		0	3	1			287		
FSTSW	AX	6	0		0	3	1			287		
FNSTSW	m16	2	3			8	0			87		
FNSTCW	m16	4	0			3	0			87		
FLDCW	m16	3	6			10	0,2			87	f	
Arithmetic instructions												
FADD(P),FSUB(R)(P)	r	1	0	6	1	1	1	fp	add	87		
FADD,FSUB(R)	m	2	0	6	1	1	1	fp	add	87		
FIADD,FISUB(R)	m16	3	3	7	1	6	1	fp	add	87		
FIADD,FISUB(R)	m32	3	0	6	1	2	1	fp	add	87		
FMUL(P)	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	J ,	
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		Ū	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	02	1	0			1	1	fp	misc	87		
FXAM		1	0			1	1	fp	misc	87		
FRNDINT		3	14	28	1	16	0,1	۱۲	111100	87		
FPREM		8	86	220	1	.	1	fp		87		
FPREM1		9	92	220	1		1	fp		387		
			02	220	•		•	ıp.		007		
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		≈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

The latency for FLDCW is 3 when the new value loaded is the same as the value of the control word before the preceding FLDCW, i.e. when alternating

between the same two values. In all other cases, the latency and reciprocal

throughput is > 100.

g) Latency and reciprocal throughput depend on the precision setting in the F.P.

control word. Single precision: 32, double precision: 40, long double precision

(default): 45.

h) Throughput of FP-MUL unit is reduced during the use of the FP-DIV unit.

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

throughput is the same.

# **Integer MMX and XMM instructions**

Instruction	Operands	sdor	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		

MOVQ2DQ	xmm,mm	2	0	10	1	2	0,1	mov-mmx	sse2		
MOVNTQ	m,mm	3	0	10		4	0, 1	mov	3362	sse	
MOVNTDQ	m,xmm	2	0			4	0	mov		sse2	
MOVDDUP		1	0	2	1	2	1	mmx	shift	sse3	
MOVSHDUP	xmm,xmm	'	U		1			IIIIIX	SHIIL	5563	
MOVSLDUP	xmm,xmm	1	0	4	1	2	1	mmx	shift	sse3	
PACKSSWB/DW	AIIIII, AIIIIII		U	_		_	'	1111111	Sillit	3363	
PACKUSWB	mm,r/m	1	0	2	1	2	1	mmx	shift	mmx	a
PACKSSWB/DW	,	•	Ū	_	•	_					_
PACKUSWB	xmm,r/m	1	0	4	1	4	1	mmx	shift	mmx	a
PUNPCKH/LBW/WD/	,										
DQ	mm,r/m	1	0	2	1	2	1	mmx	shift	mmx	a
PUNPCKHBW/WD/DQ/											
QDQ	xmm,r/m	1	0	4	1	4	1	mmx	shift	sse2	а
PUNPCKLBW/WD/DQ/Q											
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
PSUB(U)SB/W PADDQ, PSUBQ	mm,r/m	1	0	2	1	1	1	mmx	alu	sse2	a
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ		· 1	-		-						· ·
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D	mm,r/m xmm,r/m	1	0	2 5	1	1 2	1	mmx fp	alu add	sse2 sse2	a
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D	mm,r/m xmm,r/m r,r/m	1 1 1	0 0	2 5 2	1 1 1	1,2	1 1 1	mmx fp mmx	alu add alu	sse2 sse2 mmx	a a a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW	mm,r/m xmm,r/m r,r/m r,r/m	1 1 1 1	0 0 0	2 5 2 7	1 1 1	1 2 1,2 1,2	1 1 1 1	mmx fp mmx fp	alu add alu mul	sse2 sse2 mmx mmx	a a a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW	mm,r/m xmm,r/m r,r/m r,r/m r,r/m	1 1 1 1 1	0 0 0 0 0	2 5 2 7 7	1 1 1 1	1 2 1,2 1,2 1,2	1 1 1 1 1	mmx fp mmx fp fp	alu add alu mul mul	sse2 sse2 mmx mmx sse	a a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1	0 0 0 0 0	2 5 2 7 7	1 1 1 1 1 1	1 2 1,2 1,2 1,2 1,2	1 1 1 1 1 1	mmx fp mmx fp fp fp	alu add alu mul mul mul	sse2 sse2 mmx mmx sse mmx	a a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	2 5 2 7 7 7	1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1	mmx fp mmx fp fp fp fp	alu add alu mul mul mul mul	sse2 sse2 mmx mmx sse mmx sse2	a a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1	0 0 0 0 0 0 0	2 5 7 7 7 7 2	1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp fp fp mmx	alu add alu mul mul mul mul alu	sse2 sse2 mmx mmx sse mmx sse2 sse	a a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1	0 0 0 0 0 0 0	2 5 2 7 7 7 7 2 2	1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx	alu add alu mul mul mul alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse	a a,j a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB PMIN/MAXSW	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	2 5 2 7 7 7 7 2 2	1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx mmx	alu add alu mul mul mul alu alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse sse	a a,j a,j a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1	0 0 0 0 0 0 0	2 5 2 7 7 7 7 2 2	1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx	alu add alu mul mul mul alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse	a a,j a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB PMIN/MAXSW PSADBW	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	2 5 2 7 7 7 7 2 2	1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx mmx	alu add alu mul mul mul alu alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse sse	a a,j a,j a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB PMIN/MAXSW PSADBW  Logic	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0 0 0 0 0 0	2 5 7 7 7 7 2 2 4	1 1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx mmx mmx	alu add alu mul mul mul alu alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse sse sse	a a a, j a, j a, j a, j a, j a, j a, j a
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB PMIN/MAXSW PSADBW  Logic PAND, PANDN	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0 0	2 5 7 7 7 7 2 2 4	1 1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx mmx mmx mmx	alu add alu mul mul mul alu alu alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse sse sse	a a a,j a,j a,j a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB PMIN/MAXSW PSADBW  Logic PAND, PANDN POR, PXOR	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0 0 0 0 0 0	2 5 7 7 7 7 2 2 4	1 1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx mmx mmx	alu add alu mul mul mul alu alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse sse sse	a a a, j a, j a, j a, j a, j a, j a, j a
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB PMIN/MAXSW PSADBW  Logic PAND, PANDN POR, PXOR PSLL/RLW/D/Q,	mm,r/m xmm,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0 0	2 5 2 7 7 7 7 2 2 4	1 1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx mmx mmx mmx	alu add alu mul mul alu alu alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse sse sse sse	a a a,j a,j a,j a,j a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB PMIN/MAXSW PSADBW  Logic PAND, PANDN POR, PXOR PSLL/RLW/D/Q, PSRAW/D	mm,r/m xmm,r/m  r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1 1 1 1		2 5 2 7 7 7 7 2 2 4	1 1 1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx mmx mmx mmx mmx	alu add alu mul mul mul alu alu alu alu alu shift	sse2 sse2 mmx mmx sse mmx sse2 sse sse sse sse sse mmx mmx mmx	a a a,j a,j a,j a,j a,j a,j a,j a,j
PSUB(U)SB/W PADDQ, PSUBQ PADDQ, PSUBQ PCMPEQB/W/D PCMPGTB/W/D PMULLW PMULHW PMULHUW PMADDWD PMULUDQ PAVGB/W PMIN/MAXUB PMIN/MAXSW PSADBW  Logic PAND, PANDN POR, PXOR PSLL/RLW/D/Q,	mm,r/m xmm,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m r,r/m	1 1 1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0 0	2 5 2 7 7 7 7 2 2 4	1 1 1 1 1 1 1 1 1 1	1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2 1,2	1 1 1 1 1 1 1 1 1 1	mmx fp mmx fp fp fp fp mmx mmx mmx mmx mmx	alu add alu mul mul alu alu alu alu	sse2 sse2 mmx mmx sse mmx sse2 sse sse sse sse sse	a a a,j a,j a,j a,j a,j a,j a,j a,j a,j

#### 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	on	<u> </u>	La	Ad	Rec	Port	Ex	NS.	Ing	Z
	·	hobs	Microcode	Latency	Additional latency	Reciprocal through- put	)TI	Execution unit	Subunit	Instruction set	Notes
					ncy	ough-					
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	a	
CVT(T)PD2PI	mm,xmm	4	0	12	1	3	0,1	fp-mmx	sse2	a	
CVTSI2SS	xmm,r32	3	0	20	1	4	1	fp-mmx	sse	a	
CVTSI2SD	xmm,r32	4	0	20	1	5	1	fp-mmx	sse2	a	
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
(. / 5 5 2 5 .	,	-			-			.,			
Arithmetic											
ADDPS/D ADDSS/D	r,r/m	1	0	5	1	2	1	fp	add	sse	a
SUBPS/D SUBSS/D	r,r/m	1	0	5	1	2	1	fp	add	sse	a
ADDSUBPS/D	r,r/m	1	0	5	1	2	1	fp	add	sse3	a
HADDPS/D HSUBPS/D	r,r/m	3	0	13	1	5-6	1	fp	add	sse3	a
MULPS/D MULSS/D	r,r/m	1	0	7	1	2	1	fp	mul	sse	a
DIVSS	r,r/m	1	0	32	1	23	1	fp	div	sse	a,h
DIVPS	r,r/m	1	0	41	1	41	1	fp	div	sse	a,h
DIVSD	r,r/m	1	0	40	1	40	1	fp	div	sse2	a,h
DIVPD	r,r/m	1	0	71	1	71	1	fp	div	sse2	a,h
RCPPS RCPSS	r,r/m	2	0	6	1	4	1	mmx		sse	a
MAXPS/D	ŕ										
MAXSS/DMINPS/D											
MINSS/D	r,r/m	1	0	5	1	2	1	fp	add	sse	а
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	а
Logic											
ANDPS/D ANDNPS/D ORPS/D XORPS/D	r,r/m	1	0	2	1	2	1	mmy	alu	000	
ON SID XON SID	1,1/111	'	U		ı		'	mmx	aiu	sse	а
Math											
SQRTSS	r,r/m	1	0	32	1	32	1	fp	div	sse	a,h
SQRTPS	r,r/m	1	0	41	1	41	1	fp	div	sse	a,h
SQRTSD	r,r/m	1	0	40	1	40	1	fp	div	sse2	a,h
SQRTPD	r,r/m	1	0	71	1	71	1	fp	div	sse2	a,h
RSQRTSS	r,r/m	2	0	5	1	3	1	mmx		sse	a
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

XLAT							
PUSH	XLAT		3		6	6	
PUSH		r		np			
PUSH							
PUSH				119			
PUSHA(D)   PUSHA(D)   PUSHA(D)   PUSHA(D)   PUSHA(D)   PUSHA(D)   PUSHA(D)   POP   r							
PUSHA(D)		SI					
POP	1						N
POP					_		Not in x64 mode
POP							
POP		(E/R)SP		np	1		
POPF(D/Q)		m					
POPA(D)	POP	sr	7			31	
LAHF	POPF(D/Q)		19			28	
LAHF   SAHF	POPA(D)		16			12	Not in x64 mode
SAHF   SALC	` '		1	ALU0+1	2	2	
SALC							
LEA				ALOUT			Not in v64 mode
LEA	SALC				'	3	
BSWAP		r m	4	A C L 14	1 1	4	
LDS LES LFS LGS LSS							on input register
PREFETCHNTA         m         1         Mem         1         I         Mem         1         I		r		ALU0			
PREFETCHTO/1/2         m         1         Mem         1         1/2           MFENCE         1         1         1/2         1           SFENCE         1         1         1/2         1           ADD SUB         r, r/i         1         ALU0/1         1         1/2           ADD SUB         r, rm         1         ALU0/1, Mem         1         1           ADD SUB         m, r/i         1         2         1         1           ADD SUB         m, r/i         1         2         1         4         1         1         1         1         1         1         2		m	10		30	30	
LFENCE	PREFETCHNTA	m	1	Mem		1	
MFENCE         1 <td>PREFETCHT0/1/2</td> <td>m</td> <td>1</td> <td>Mem</td> <td></td> <td>1</td> <td></td>	PREFETCHT0/1/2	m	1	Mem		1	
Arithmetic instructions	LFENCE		1			1/2	
Arithmetic instructions	MFENCE		1			1	
Arithmetic instructions							
ADD SUB         r,r/i         1         ALU0/1         1         1/2           ADD SUB         r,m         1         ALU0/1, Mem         1           ADD SUB         m,r/i         1         2         1           ADC SBB         r,r/i         1         2         2           ADC SBB         r,m         1         2         2           ADC SBB         m,r/i         1         2         2           CMP         r,r/i         1         ALU0/1         1         1/2           INC DEC NEG NOT         r         1         ALU0/1         1         1/2           INC DEC NEG NOT         m         1         1         1         1/2           INC DEC NEG NOT         m         1         1         1         1/2         Not in x64 mode           DAA         13         16         Not in x64 mode         Not in x64 mode         Not in x64 mode         Not in x64 mode <td>3. 2.132</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>	3. 2.132						
ADD SUB         r,m         1         ALU0/1, Mem         1           ADD SUB         m,r/i         1         2         1           ADC SBB         r,r/i         1         2         2           ADC SBB         r,m         1         2         2           ADC SBB         m,r/i         1         2         2           CMP         r,r/i         1         ALU0/1         1         1/2           CMP         m,r/i         1         ALU0/1         1         1/2           INC DEC NEG NOT         r         1         ALU0/1         1         1/2           INC DEC NEG NOT         m         1         1         A         Not in x64 mode           AAS         20         20 <td< td=""><td>Arithmetic instructions</td><td></td><td></td><td></td><td></td><td></td><td></td></td<>	Arithmetic instructions						
ADD SUB         r,m         1         ALU0/1, Mem         1           ADD SUB         m,r/i         1         2         1           ADC SBB         r,r/i         1         2         2           ADC SBB         r,m         1         2         2           ADC SBB         m,r/i         1         2         2           CMP         r,r/i         1         ALU0/1         1         1/2           CMP         m,r/i         1         ALU0/1         1         1/2           INC DEC NEG NOT         r         1         ALU0/1         1         1/2           INC DEC NEG NOT         m         1         1         A         Not in x64 mode           AAS         20         20 <td< td=""><td>ADD SUB</td><td>r.r/i</td><td>1</td><td>ALU0/1</td><td>1</td><td>1/2</td><td></td></td<>	ADD SUB	r.r/i	1	ALU0/1	1	1/2	
ADD SUB         m,r/i         1         2         1           ADC SBB         r,r/i         1         2         2           ADC SBB         r,m         1         2         2           ADC SBB         m,r/i         1         2         2           CMP         r,r/i         1         ALU0/1         1         1/2           CMP         m,r/i         1         1         1/2           INC DEC NEG NOT         r         1         ALU0/1         1         1/2           INC DEC NEG NOT         m         1         1         1           AAS         13         12         Not in x64 mode         Not in x64 mode           DAA         20         20         Not in x64 mode         Not in x64 mode         Not in x64 mode           AAM<							
ADC SBB ADC SC SC SC SC SC SC SC SC SC SC SC SC SC				7 (2007 1, 1VIOI			
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         1/2           INC DEC NEG NOT         r         1         ALU0/1         1         1/2           INC DEC NEG NOT         m         1         1         1           AAB         13         12         Not in x64 mode         Not in x64 mode           DAA         20         20         Not in x64 mode         Not in x64 mode         Not in x64 mode							
ADC SBB							
CMP         r,r/i         1         ALU0/1         1         1/2           CMP         m,r/i         1         1         1         1/2           INC DEC NEG NOT         r         1         ALU0/1         1         1/2           INC DEC NEG NOT         m         1         1         1/2           INC NOT NOT NOT NOT NOT NOT NOT NOT NOT NOT							
CMP         m,r/i         1         ALU0/1         1         1/2           INC DEC NEG NOT         m         1         1         1/2           INC DEC NEG NOT         m         1         1         1           AAA         13         16         Not in x64 mode           AAS         13         12         Not in x64 mode           DAA         20         20         Not in x64 mode           DAS         21         25         Not in x64 mode           AAD         4         7         Not in x64 mode           AAM         10         24         Not in x64 mode           MUL IMUL         r8         3         ALU0, Mul         7         7           MUL IMUL         r16         4         ALU0, Mul         6         6           MUL IMUL         r32         3         ALU0, Mul         6         6           MUL IMUL         r64         8         ALU0, Mul         14         14           IMUL         r16,r16         2         ALU0, Mul         5         2           IMUL         r64,r64         6         ALU0, Mul         13         11							
INC DEC NEG NOT				ALU0/1	1		
INC DEC NEG NOT	CMP	m,r/i	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         MUL IMUL       r8       3       ALU0, Mul       7       7         MUL IMUL       r16       4       ALU0, Mul       6       6         MUL IMUL       r32       3       ALU0, Mul       6       6         MUL IMUL       r64       8       ALU0, Mul       14       14         IMUL       r16,r16       2       ALU0, Mul       6       5         IMUL       r32,r32       1       ALU0, Mul       5       2         IMUL       r64,r64       6       ALU0, Mul       13       11	INC DEC NEG NOT	r	1	ALU0/1	1	1/2	
AAS       13       12       Not in x64 mode         DAA       20       20       Not in x64 mode         DAS       21       25       Not in x64 mode         AAD       4       7       Not in x64 mode         AAM       10       24       Not in x64 mode         MUL IMUL       r8       3       ALU0, Mul       7       7         MUL IMUL       r16       4       ALU0, Mul       6       6         MUL IMUL       r32       3       ALU0, Mul       6       6         MUL IMUL       r64       8       ALU0, Mul       14       14         IMUL       r16,r16       2       ALU0, Mul       6       5         IMUL       r32,r32       1       ALU0, Mul       5       2         IMUL       r64,r64       6       ALU0, Mul       13       11	INC DEC NEG NOT	m	1		1		
DAA       20       20       Not in x64 mode         DAS       21       25       Not in x64 mode         AAD       4       7       Not in x64 mode         NOT in x64 mode       Not in x64 mode       Not in x64 mode         NUL IMUL       r8       3       ALU0, Mul       7       7         MUL IMUL       r16       4       ALU0, Mul       6       6         MUL IMUL       r32       3       ALU0, Mul       6       6         MUL IMUL       r64       8       ALU0, Mul       14       14         IMUL       r16,r16       2       ALU0, Mul       6       5         IMUL       r32,r32       1       ALU0, Mul       5       2         IMUL       r64,r64       6       ALU0, Mul       13       11	AAA		13		16		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         NOT       Not in x64 mode       Not in x64 mode         NOT       Not in x64 mode         NOT       Not in x64 mode         NOT       NOT	AAS		13		12		Not in x64 mode
DAS       21       25       Not in x64 mode         AAD       4       7       Not in x64 mode         AAM       10       24       Not in x64 mode         MUL IMUL       r8       3       ALU0, Mul       7       7         MUL IMUL       r16       4       ALU0, Mul       6       6         MUL IMUL       r32       3       ALU0, Mul       6       6         MUL IMUL       r64       8       ALU0, Mul       14       14         IMUL       r16,r16       2       ALU0, Mul       6       5         IMUL       r32,r32       1       ALU0, Mul       5       2         IMUL       r64,r64       6       ALU0, Mul       13       11							
AAD       4       7       Not in x64 mode         AAM       10       24       Not in x64 mode         MUL IMUL       r8       3       ALU0, Mul       7       7         MUL IMUL       r16       4       ALU0, Mul       6       6         MUL IMUL       r32       3       ALU0, Mul       6       6         MUL IMUL       r64       8       ALU0, Mul       14       14         IMUL       r16,r16       2       ALU0, Mul       6       5         IMUL       r32,r32       1       ALU0, Mul       5       2         IMUL       r64,r64       6       ALU0, Mul       13       11							
AAM       10       24       Not in x64 mode         MUL IMUL       r8       3       ALU0, Mul       7       7         MUL IMUL       r16       4       ALU0, Mul       6       6         MUL IMUL       r32       3       ALU0, Mul       6       6         MUL IMUL       r64       8       ALU0, Mul       14       14         IMUL       r16,r16       2       ALU0, Mul       6       5         IMUL       r32,r32       1       ALU0, Mul       5       2         IMUL       r64,r64       6       ALU0, Mul       13       11							
MUL IMUL       r8       3       ALU0, Mul       7       7         MUL IMUL       r16       4       ALU0, Mul       6       6         MUL IMUL       r32       3       ALU0, Mul       6       6         MUL IMUL       r64       8       ALU0, Mul       14       14         IMUL       r16,r16       2       ALU0, Mul       6       5         IMUL       r32,r32       1       ALU0, Mul       5       2         IMUL       r64,r64       6       ALU0, Mul       13       11							
MUL IMUL       r16       4       ALU0, Mul       6       6         MUL IMUL       r32       3       ALU0, Mul       6       6         MUL IMUL       r64       8       ALU0, Mul       14       14         IMUL       r16,r16       2       ALU0, Mul       6       5         IMUL       r32,r32       1       ALU0, Mul       5       2         IMUL       r64,r64       6       ALU0, Mul       13       11		_				_	Not in x64 mode
MUL IMUL     r32     3     ALU0, Mul     6     6       MUL IMUL     r64     8     ALU0, Mul     14     14       IMUL     r16,r16     2     ALU0, Mul     6     5       IMUL     r32,r32     1     ALU0, Mul     5     2       IMUL     r64,r64     6     ALU0, Mul     13     11							
MUL IMUL     r64     8     ALU0, Mul     14     14       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     2     ALU0, Mul     6     5       IMUL     r32,r32     1     ALU0, Mul     5     2       IMUL     r64,r64     6     ALU0, Mul     13     11							
IMUL     r32,r32     1     ALU0, Mul     5     2       IMUL     r64,r64     6     ALU0, Mul     13     11		r64			14	14	
IMUL r64,r64 6 ALU0, Mul 13 11	IMUL	r16,r16	2	ALU0, Mul	6		
IMUL r64,r64 6 ALU0, Mul 13 11	IMUL	r32,r32	1	ALU0, Mul	5	2	
	IMUL		6		13	11	

IMUL	r32,r32,i	1	ALU0, Mul	5	2	
IMUL	r64,r64,i	1 7	ALU0, Mul	14	14	
MUL IMUL	m8	3	ALU0, Mul	6	14	
MUL IMUL	m16	5	ALU0, Mul	7		
MUL IMUL	m32	4		7		
			ALUO, Mul			
MUL IMUL	m64	8	ALU0, Mul	14	00	
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		2	ALU0	5		
CWDE		1	ALU0	1		
CDQE		1	ALU0	1		
CWD		2	ALU0	5		
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	n	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	n	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	10		1-2 more if mem
SHLD	r32,r32,i	2	ALU0	5		1-2 more if mem
SHLD	r64,r64,i	10	ALU0	11		1-2 more if mem
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
						1-2 more if mem
SHRD SHRD	r32,r32,i r64,r64,i	2	ALU0 ALU0	5 9		1-2 more if mem
		10				
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
BT	r,r/i	1	ALU1	1	1	
BT	m,r	9		10		
BT	m,i	2		5		

BTR BTS BTC	r,r/i	1	ALU1	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	1,111	4			6	Not in x64 mode
		7			O	Not in XO+ mode
String instructions						
LODS		3		6		
REP LODS		5n+11		3n+50		
STOS		2		5		
REP STOS		3n+10		2n+4		
MOVS		4		6		
REP MOVS		4n+11		2n - 4n		fastest for high n
SCAS		3		6		
REP SCAS		5n+16		3n+60		
CMPS		5		7		
REP CMPS		6n+16		4n+40		
Other						
NOP (90)		1	ALU0/1		1/2	
Long NOP (0F 1F)		1	ALU0/1		1/2	
PAUSE		5	7.12071	24	., _	
ENTER	a,0	14		23		
	a,0	14		23		1

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 instructions										
	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 FPREM1		1 26 37		1 ~110 ~130	1	
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	
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 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	(*)!!!!!,(*)!!!!!	'	11.0/1	'	1/2	
I SIGNET SIGNW I SIGNE	(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	
I SLL/RLDQ	AHIIII,I	I	FFU	1	I	
Other						
EMMS		9			9	

Floating point XMM instructions

Operands	μops	Unit	Latency	Reciprocal	Remarks
				throughput	

laa			I		I	I
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	
UNPCKH/LPS		1	FP0	1	1	
UNPCKH/LPD	xmm,xmm	1	FP0	1	'	
ONFON I/LFD	xmm,xmm	ı	rru	'		
Conversion						
CVTPD2PS	ymm ymm	4		11	11	
CVTSD2SS	xmm,xmm	3		10	10	
CVTSD233 CVTPS2PD	xmm,xmm			7		
	xmm,xmm	4			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	
A middle man and i						
Arithmetic	-	_	FD.4	_	_	
ADDSS SUBSS	xmm,xmm	1	FP1	5	1	
ADDSD SUBSD	xmm,xmm	1	FP1	5	1	
ADDPS SUBPS	xmm,xmm	1	FP1	5	1	
ADDPD SUBPD	xmm,xmm	3	FP1	6	6	
ADDSUBPS	xmm,xmm	1	FP1	5	1	
ADDSUBPD	xmm,xmm	3	FP1	6	6	
HADDPS HSUBPS	xmm,xmm	5	FP0+1	8	7	
HADDPD HSUBPD	xmm,xmm	5	FP0+1	8	7	
MULSS	xmm,xmm	1	FP0, Mul	4	1	I

MULSD	vmm vmm	1	FP0, Mul	5	2	
MULPS	xmm,xmm	1	FP0, Mul	5	2	
MULPD	xmm,xmm		FP0, Mul	9	9	
	xmm,xmm	6	· '		9 31	
DIVSS	xmm,xmm	3	FP0, Div	31 60	60	
DIVSD	xmm,xmm	3	FP0, Div			
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:** 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, x = 128 bit xmm

register, (x)mm = mmx or xmm register, m = memory, m32 = 32-bit memory

operand, etc.

**μορs:** The number of μops from the decoder or ROM. A μop that goes to multiple

units is counted as one.

**Unit:** Tells which execution unit is used. Instructions that use the same unit cannot

execute simultaneously.

IP0 and IP1 means integer port 0 or 1 and their associated pipelines

IP0/1 means that either integer unit can be used. IP0+1 means that both units are used at the same time.

Mem means memory execution cluster

FP0 means floating point port 0 (includes multiply, divide, convert and shuf-

fle).

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	1/2	
MOV	r,i	1	IP0/1	1	1/2	
MOV	r,m	1	Mem	3	1	All addr. modes
MOV	m,r	1	Mem	4	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	5	5	
XCHG	r,m	3		10	10	Implicit lock
XLAT		4		5	5	-
PUSH	r	1	IP0+1		1	

PUSH	i	1	IP0+1		1	
PUSH	m	3	IP0+1		5	
PUSHF(D/Q)		18	IP0+1		29	
PUSHA(D)		10			10	Not in x64 mode
POP	r	2			1	
POP	(E/R)SP	2			3	
POP		6			6	
	m					
POPF(D/Q)		21			47	
POPA(D)		17			14	Not in x64 mode
LAHF		1		1	1	
SAHF		1	IP0	2	1	
SALC		2		6	4	Not in x64 mode
LEA	r,[r+d]	1	IP0/1	1	1	
LEA	r,[r+r*s]	1	IP1	1	1	
LEA	r,[r+r*s+d]	1	IP0+1	2	2	
LEA	r,[rip+d]	1	IP0/1	_	0.5	
LEA		2	11 0/1	4	4	
BSWAP	r16,[m]		IP0			
	r	1	IPU	1	1	
MOVBE	r16,m16	1			2	
MOVBE	r32/64,m32/64	1			1	
MOVBE	m,r	1			1	
PREFETCHNTA	m	1			1	
PREFETCHT0/1/2	m	1			1	
PREFETCHNTW	m	1			1	
LFENCE		2			8	
MFENCE		2			14	
SFENCE		1			7	
SPENCE		ı			/	
A vidla va adi a imadov adi a va						
Arithmetic instructions			l locate		4.0	
ADD SUB	r,r/i	1	IP0/1	1	1/2	
ADD SUB	r,m	1	IP0/1, Mem		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			2	
ADC SBB	m,r/i	1		6	2	
CMP	r,r/i	1	IP0/1	1	1/2	
CMP	m,r/i	1			1	
INC DEC	r	1	IP0/1	1	1/2	latency to flag=2
NEG NOT	r	1	IP0/1	1	1/2	lateries to mag =
INC DEC NEG NOT	m	1	11 0/ 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	
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	
I				5 5	2	
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		
DIV				24	19	
	r/m8	9	IP0, FP0			
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		2	IP0	4		
CWDE		1	IP0	1		
CDQE		1	IP0	1		
CWD		2	IP0	4		
CDQ		1	IP0	1		
CQO		1	IP0	1		
	r16 r16		11.0		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	1/2	
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	1/2	
TEST	m,r/i	1	IP0/1, Mem		1	
SHR SHL SAR	r,i/cl		IP0	1		
SHR SHL SAR		1		1	1	
	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
SHRD	r32,r32,i	2	IP0	4		4 more if mem
SHRD	r64,r64,i	8-10	IP0	10		3 more if mem
אוועט	104,104,1	0-10	IFU	10		

SHRD SHRD SHRD BT BT BT BT BT BTR BTS BTC BTR BTS BTC BTR BTS BTC BTR BTS BTC CCC CCC STC CMC CLD STD	r16,r16,cl r32,r32,cl r64,r64,cl r,r/i m,r m,i r,r/i m,r m,i r,r/m	7 2 2 1 7 1 1 8 1 10 1 1 1 4 5	IP0 IP0 IP0 IP0+1 IP0+1 IP0+1 IP0+1 IP0/1 IP0+1 IP0+1	10 4 4 1 9 1 1 10 2	1 10 1 10 1 1 1 7 35	2 more if mem 2 more if mem 2 more if mem
Control transfer instruction	ns					
JMP	short/near	1 1	IP1		2	
JMP	r	1			2	
JMP	m(near)	1			2	
Conditional jump	short/near	1	IP1		1-2	
J(E/R)CXZ	short	2			2-15	
LOOP	short	7			10-20	
LOOP(N)E	short	8				
CALL	near	1			2	
CALL	r	1			9	
CALL	m	3			14	
RET	i	1 1			3 3	
RET BOUND	r,m	10			3 10	Not in x64 mode
INTO	1,111	4			7	Not in x64 mode
String instructions						
LODS		3		5		
REP LODS STOS		~4n 2		~2n 4		
3103		4		4		nor hyto hoot
REP STOS		~0.12B		~0.1B		per byte, best case
MOVS		5		6		0000
						per byte, best
REP MOVS		~ 0.2B		~0.15B		case
SCAS		3		5		
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		1/2	
Long NOP (0F 1F)		1	IP0/1		1/2	
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		3	1	
FLD	m80	5		9	8	
FBLD	m80	59		68	68	
FST(P)	r	1		1	0.5	
FST(P)	m32/m64	1		4	2	
FSTP	m80	8		9	9	
FBSTP	m80	204		239	239	
FXCH	r	2	FP0+1	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	m32/64,(x)mm	1	Mem	4	1	
MOVD MOVQ	(x)mm,r32/64	1		3	1	
MOVD MOVQ	(x)mm,m32/64	1	Mem	3	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	3	1	
MOVDQA MOVDQU	m128, x	1	Mem	4	1	
LDDQU	x, m128	1	Mem	3	1	
MOVDQ2Q	mm, x	1		1	1 1	
MOVQ2DQ	x,mm	1		1	1 1	
MOVNTQ	m64,mm	1	Mem	~370	1 1	
MOVNTDQ	m128,x	1	Mem	~370	1 1	
MOVNTDQA	x, m128	1		3	1	

PACKSSWB/DW						
PACKUSWB	(x)mm, (x)mm	1	FP0	1	1	
PACKUSDW	X,X	1	FP0	1	1	
PUNPCKH/LBW/WD/DQ	(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		1	1	
PMOVSX/ZX BW BD BQ DW						
DQ	x,m	1		1	1	
PSHUFB	mm,mm	1	FP0	1	1	
PSHUFB	X,X	4	FP0	5	5	
PSHUFW	mm,mm,i	1	FP0	1	1	
PSHUFL/HW	x,x,i	1	FP0	1	1	
PSHUFD	x,x,i	1	FP0	1	1	
PALIGNR	x,x,i	1	FP0	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	5	
PMOVMSKB	1	3 1	IVICIII		1	
	r32,(x)mm	•		4	1	
PINSRW	(x)mm,r32,i	1		3	l 4	
PINSRB/D/Q	x,r32,i	1		3	1	
PINSRB/D/Q	x,m8,i	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	
PEXTRB/W	m8/16,x,i	5			6	
PEXTRD	m32,x,i	4			5	
PEXTRQ	m64,x,i	4			8	
Auddhus adda da adamadda a						
Arithmetic instructions		4				
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	
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	
PMULL/HW PMULHUW	x, x	1	FP0	5	2	
PMULHRSW	mm,mm	1	FP0	4	1	
PMULHRSW	X, X	1	FP0	5	2	
PMULLD	X, X	7	FP0	11	11	+1 if mem
PMULDQ	x, x	1	FP0	5	2	
PMULUDQ	mm,mm	1	FP0	4	1	
PMULUDQ	x, x	1	FP0	5	2	
PMADDWD	mm,mm	1	FP0	4	1	
PMADDWD	x, x	1	FP0	5	2	
PMADDUBSW	mm,mm	1	FP0	4	_ 1	
PMADDUBSW	x, x	1	FP0	5	2	
1	1 -7	-		-	<del>-</del>	ı

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						
PCLMULQDQ	x,x,i	8	FP0	10	10	+1 if mem
I OLINIOL QUQ	^,^,1	U	110	10	10	' I II IIIGIII
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	3	1	
MOVAPS/D	m128,x	1	Mem	4	1	
MOVUPS/D	x,m128	1	Mem	3	1	
MOVUPS/D	m128,x	1	Mem	4	1	
MOVSS/D	x, x	1	FP0/1	1	0.5	
MOVSS/D	x,m32/64	1	Mem	3	1	
MOVSS/D	m32/64,x	1	Mem	4	1	
MOVHPS/D MOVLPS/D	x,m64	1	Mem	3	1	
MOVHPS/D MOVLPS/D	m64,x	1	Mem	4	1	
MOVLHPS MOVHLPS	X,X	1	FP0	1	1	
BLENDPS/PD	x,x/m,i	1		1	1	
BLENDVPS/PD	x,x,xmm0	2	FP0+1	4	4	
BLENDVPS/PD	x,m,xmm0	3	FP0+1	5	5	

INSERTPS			Oiiv	Cillioni			
INSERTPS	INSERTPS	x.x.i	1		1	1	
EXTRACTPS							
EXTRACTPS					-		
MOWNSKPS/ID		1 ' '			5		
MOVNTPS/D				FP0			
SHUFPS SHUFPD SHUFPD X,X,i 1 FP0 1 1 1 MOVDDUP X, x 1 FP0 MOVBH/LDUP X, x 1 FP0 X, X 1 FP1 X, X 1 F			-				
SHUPPD							
MOVDDUP         X, X         1         FP0         1         1           MOVSHILDUP         X, X         1         FP0         1         1           UNPCKHILPS         X, X         1         FP0         1         1           UNPCKHILPD         X, X         1         FP0         1         1           COTHOLOGIS         X, X         1         FP0         1         1           COTROLOGIS         X, X         1         FP0         4         2           CVTD2SS         X, X         1         FP0         4         2           CVTS2SDD         X, X         1         FP0         4         2           CVTTQ2PS         X, X         1         FP0         5         2           CVTTQ1PS2DQ         X, X         1         FP0         5         2           CVTTQ1PDDQ         X, X         1         FP0         5         2           CVTTP12PS         X,mm         1         FP0         5         2           CVTT(1)P52P1         mm,x         1         FP0         5         2           CVT(1)P52P3         mm,x         1         FP0         5         2						· -	
MOVSH/LDUP							
UNPCKH/LPS UNPCKH/LPD							
UNPCKH/LPD							
Conversion   Con							
CVTPD2PS         x, x         1         FP0         5         2           CVTSD2SS         x, x         1         FP0         4         2           CVTS2PD         x, x         1         FP0         5         2           CVTD2PS         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)PS2PI         mm, x         1         FP0         4         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)S2SI         r32, x         1         FP0         5         2           CVT(T)SD2SI         r32, x         1         FP0         5         2           CVT(T)SD2SI         r32, x         1         FP0         5         <	UNPCKH/LPD	x, x	1	FP0	1	1	
CVTPD2PS         x, x         1         FP0         5         2           CVTSD2SS         x, x         1         FP0         4         2           CVTS2PD         x, x         1         FP0         5         2           CVTD2PS         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)PS2PI         mm, x         1         FP0         4         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)S2SI         r32, x         1         FP0         5         2           CVT(T)SD2SI         r32, x         1         FP0         5         2           CVT(T)SD2SI         r32, x         1         FP0         5         <	Conversion						
CVTSD2SS		x. x	1	FP0	5	2	
CVTPS2PD         x, x         1         FP0         5         2           CVTSS2SD         x, x         1         FP0         4         2           CVTD2PS         x, x         1         FP0         5         2           CVT(T) PS2DQ         x, x         1         FP0         5         2           CVT(T)PD2PD         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)PS2PI         mm, x         1         FP0         5         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)S2SI         r32,x         3         FP0         5         1           APDSS SUBSS         x, x         1         FP1         3         1           ADDSD SUBSD         x, x         1         FP1         3							
CVTSS2SD         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)PS2PI         x,mm         1         FP0         4         2           CVT(T)PS2PI         x,mm         1         FP0         4         2           CVT(T)PS2PI         x,mm         1         FP0         5         2           CVT(T)PD2PI         mm,x         1         FP0         5         2           CVT(T)PD2PI         mm,x         1         FP0         5         2           CVT(T)S2SS         x,32,x         1         FP0         5         2           CVT(T)SD2SI         r32,x         3         FP0         5         1           Arithmetic           ADDS SUBSD         x, x         1         FP1         3         1           ADDS SUBSD							
CVTDQ2PS         x, x         1         FP0         5         2           CVT(T) PS2DQ         x, x         1         FP0         5         2           CVT(DQ2PD         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         5         2           CVT(T)PD2PI         mm,x         1         FP0         5         2           CVT(T)S2SS         x,732         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         1           APDSDS SUBSS         x, x         1         FP1         3         1           ADDSD SUBSD         x, x         1         FP1         3         1           ADDSD SUBPD         x, x         1         FP1         3							
CVT(T) PS2DQ							
CVTDQ2PD							
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           CVT(T)PD2PI         x,mm         1         FP0         5         2           CVT(T)PD2PI         mm,x         1         FP0         5         2           CVT(T)S2SI         x,r32         1         FP0         5         2           CVT(T)S2SI         r32,x         1         FP0         5         2           CVT(T)S2SI         x,r32         1         FP0         5         1           Arithmetic           Arithmetic           ADDSUBSS         x,x         1         FP1         3         1 <td>, ,</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>	, ,						
CVTPI2PS         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           CVTS12SS         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           A, x         1         FP1         3         1           ADDPS SUBPS         x, x         1         FP1         3         1           ADDPS SUBPS         x, x         1         FP1         3         1           ADDPS SUBPS         x, x         1         FP1         3         1           ADDSUBPS         x, x         1         FP1         3         1           ADDSUBPS         x, x         1         FP1         4         2           HADDSUBPS							
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           CVTS12SS         x,r32         1         FP0         5         2           CVT(T)S2SI         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           ADDPS SUBPS         x, x         1         FP1         3         1           ADDSD SUBPD         x, x         1         FP1         3         1           ADDSUBPD SUBPD         x, x         1         FP1         4         2           ADDSUBPD SUBPS         x, x         1         FP1         4         2           HADDPS HSUBPS         x, x         4         6         6         +1 if mem           MULSS         x, x </td <td>, ,</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>	, ,						
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         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           ADDSD SUBPD         x, x         1         FP1         3         1           ADDSUBPD         x, x         1         FP1         3         1           ADDSUBPD         x, x         1         FP1         4         2           ADDSUBPD         x, x         4         6         6         +1 if mem           HADDPS HSUBPS         x, x							
CVT(T) PD2PI         mm,x         1         FP0         5         2           CVTSI2SS         x,r32         1         FP0         5         2           CVTSI2SD         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           HADDPS HSUBPS         x, x         1<	1 ' '					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           ADDSD SUBPD         x, x         1         FP1         3         1           ADDSUBPD         x, x         1         FP1         4         2           HADDPS HSUBPS         x, x         1         FP1         4         2           HADDPS HSUBPD         x, x <t< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td></t<>							
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           ADDSS 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           ADDSUBPS         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	, ,	· ·					
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 SUBPD         X, X         1         FP1         3         1           ADDSUBPD         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         1         FP0         4         1           MULSS         X, X         1 </td <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>							
Arithmetic         x, x         1         FP0         5         1           ADDSS SUBSS         x, x         1         FP1         3         1           ADDSD SUBSD         x, x         1         FP1         3         1           ADDSD SUBPD         x, x         1         FP1         3         1           ADDSUBPD         x, x         1         FP1         4         2           ADDSUBPS         x, x         1         FP1         3         1           ADDSUBPD         x, x         4         6         6         +1 if mem           HADDPS HSUBPS         x, x         4         6         5         +1 if mem           HADDPS HSUBPD         x, x         4         6         5         +1 if mem           MULSS         x, x         1         FP0         4         1           MULPS         x, x         1         FP0         5 <t< td=""><td>1 ' '</td><td></td><td></td><td></td><td></td><td></td><td></td></t<>	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         X, X         1         FP1         4         2           ADDSUBPS         X, X         1         FP1         3         1           ADDSUBPD         X, X         1         FP1         4         2           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         34         32							
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       3       1         ADDSUBPD       x, x       1       FP1       4       2         HADDPS HSUBPS       x, x       4       6       6       +1 if mem         HADDPS HSUBPD       x, x       4       6       5       +1 if mem         MULSS       x, x       1       FP0       4       1         MULSS       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       39       39         DIVPS       x, x       6       FP0       39       39         DIVPD       x, x       6       FP0       69       69         RCPS	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         ADDSUBPD       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         HADDPS HSUBPD       x, x       4       6       5       +1 if mem         MULSS       x, x       1       FP0       4       1         MULSS       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       39       39         DIVPS       x, x       6       FP0       39       39         DIVPD       x, x       6       FP0       69       69         RCPS	Arithmetic						
ADDSD SUBSD		Y Y	1	FP1	3	1	
ADDPS SUBPS ADDPD SUBPD ADDSUBPD ADDSUBPS ADDSUBPS ADDSUBPS ADDSUBPS ADDSUBPS ADDSUBPS ADDSUBPS ADDSUBPS ADDSUBPS ADDSUBPS ADDSUBPD ADDSUBPD ADDSUBPS ADDSUB							
ADDPD SUBPD ADDSUBPS ADDSUBPS ADDSUBPS AN, X ADDSUBPD ADD							
ADDSUBPS       x, x       1       FP1       3       1         ADDSUBPD       x, x       1       FP1       4       2         HADDPS HSUBPS       x, x       4       6       6       6       +1 if mem         HADDPD HSUBPD       x, x       4       6       5       +1 if mem         MULSS       x, x       1       FP0       4       1         MULSD       x, x       1       FP0       5       2         MULPS       x, x       1       FP0       5       2         MULPD       x, x       1       FP0       7       4         DIVSS       x, x       1       FP0       19       17         DIVSD       x, x       1       FP0       34       32         DIVPS       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       1       1 <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>							
ADDSUBPD       x, x       1       FP1       4       2         HADDPS HSUBPS       x, x       4       6       6       6         HADDPD HSUBPD       x, x       4       6       5       +1 if mem         MULSS       x, x       1       FP0       4       1       +1 if mem         MULSD       x, x       1       FP0       5       2       4       1       4							
HADDPS HSUBPS       x, x       4       6       6       +1 if mem         HADDPD HSUBPD       x, x       4       6       5       +1 if mem         MULSS       x, x       1       FP0       4       1         MULSD       x, x       1       FP0       5       2         MULPS       x, x       1       FP0       5       2         MULPD       x, x       1       FP0       7       4         DIVSS       x, x       1       FP0       19       17         DIVSD       x, x       1       FP0       34       32         DIVPS       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							
HADDPD HSUBPD       x, x       4       6       5       +1 if mem         MULSS       x, x       1       FP0       4       1         MULSD       x, x       1       FP0       5       2         MULPS       x, x       1       FP0       5       2         MULPD       x, x       1       FP0       7       4         DIVSS       x, x       1       FP0       19       17         DIVSD       x, x       1       FP0       34       32         DIVPS       x, x       6       FP0       39       39         DIVPD       x, x       6       FP0       69       69         RCPSS       x, x       1       FP0       4       1         RCPPS       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				FPI			1.4 if mann
MULSS       x, x       1       FP0       4       1         MULSD       x, x       1       FP0       5       2         MULPS       x, x       1       FP0       5       2         MULPD       x, x       1       FP0       7       4         DIVSS       x, x       1       FP0       19       17         DIVSD       x, x       1       FP0       34       32         DIVPS       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							
MULSD       x, x       1       FP0       5       2         MULPS       x, x       1       FP0       5       2         MULPD       x, x       1       FP0       7       4         DIVSS       x, x       1       FP0       19       17         DIVSD       x, x       1       FP0       34       32         DIVPS       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							+1 if mem
MULPS       x, x       1       FP0       5       2         MULPD       x, x       1       FP0       7       4         DIVSS       x, x       1       FP0       19       17         DIVSD       x, x       1       FP0       34       32         DIVPS       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				· ·			
MULPD       x, x       1       FP0       7       4         DIVSS       x, x       1       FP0       19       17         DIVSD       x, x       1       FP0       34       32         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							
DIVSS         x, x         1         FP0         19         17           DIVSD         x, x         1         FP0         34         32           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							
DIVSD         x, x         1         FP0         34         32           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							
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		X, X					
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		X, X					
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		x, x	6		39		
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	DIVPD	x, x	6	FP0	69	69	
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	RCPSS	x, x	1	FP0	4	1	
CMPccSS/D PS/D         x, x         1         FP1         3         1           COMISS/D UCOMISS/D         x, x         1         FP1         1	RCPPS		5	FP0	9	8	
COMISS/D UCOMISS/D x, x 1 FP1 1	CMPccSS/D PS/D				3	1	
			1			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

### 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  $\mu$ ops. Therefore the number of  $\mu$ ops cannot be determined except in simple cases.

**Port:** Tells which execution port or unit is used. Instructions that use the same port

cannot execute simultaneously.

Integer add, Boolean, shift, etc.Integer add, Boolean, move, jump.

I12: Can use either I1 or I2, whichever is vacant first.MA: Multiply, divide and square root on all operand types.MB: Various Integer and floating point SIMD operations.

**MBfadd:** Floating point addition subunit under MB.

**SA:** Memory store address.

ST: Memory store. LD: Memory load.

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

numbers are minimum values. Cache misses, misalignment, and exceptions may increase the clock counts considerably. Floating point operands are presumed to be normal numbers. Denormal numbers, NAN's and infinity increase the delays very much, except in XMM move, shuffle and Boolean instructions. Floating point overflow, underflow, denormal or NAN results give a similar de-

lay.

Note: There is an additional latency for moving data from one unit or subunit to another. A table of these latencies is given in manual 3: "The microarchitecture of Intel, AMD and VIA CPUs". These additional latencies are not included in the listings below where the source and destination operands are of the same type.

**Reciprocal throughput:** The average number of clock cycles per instruction for a series of independent instructions of the same kind in the same thread.

#### Integer instructions

	Operands	µops	Port	Latency	Reciprocal thruoghput	Remarks
Move instructions						
MOV	r,r	1	12	1	1	
MOV	r,i	1	12	1	1	
						Latency 4 on
MOV	r,m	1	LD	2	1	pointer register
MOV	m,r	1	SA, ST	2	1.5	
MOV	m,i	1	SA, ST		1.5	
MOV	r,sr				1	
MOV	m,sr				2	
MOV	sr,r			20	20	
MOV	sr,m			20	20	

MOVNTI	m,r		SA, ST	2	1.5	
MOVSX MOVSXD MOVZX		1	12	1	_	
MOVSX MOVSXD	r,r	1 2		1	1 1	
MOVZX MOVSAD	r,m		LD, I2	3		
	r,m	1	LD	2	1	
CMOVec	r,r	2	I1, I2	2	1	
CMOVcc	r,m		LD, I1	5 3	2 3	
XCHG	r,r	3	12			Imamiliait la alc
XCHG	r,m			20	20	Implicit lock
XLAT	m		CA CT	6	1.0	
PUSH	r :		SA, ST		1-2	
PUSH	i		SA, ST		1-2	
PUSH	m		Ld, SA, ST		2	
PUSH	sr			•	17	
PUSHF(D/Q)				8	8	
PUSHA(D)					15	Not in x64 mode
POP	r		LD		1.25	
POP	(E/R)SP				4	
POP	m				5	
POP	sr				20	
POPF(D/Q)				9	9	
POPA(D)					12	Not in x64 mode
LAHF		1	I1	1	1	
SAHF		1	I1	1	1	
SALC				9	6	Not in x64 mode
LEA	r,m	1	SA	1	1	3 clock latency on 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	I1	1	1	
ADC SBB	r,m	2	LD I1		1	
ADC SBB	m,r/i	3	LD I1 SA ST	5	2	
CMP	r,r/i	1	l12	1	1/2	
CMP	m,r/i	2	LD I12		1	
INC DEC NEG NOT	r	1	l12	1	1/2	
INC DEC NEG NOT	m	3	LD I12 SA ST	5		
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	I12	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	l1	1	1	
ROR ROL	r,i/cl	1	I1	1	1	
RCR RCL	r,1	1	I1	1	1	
RCR RCL	r,i/cl		l1	28+3n	28+3n	
SHLD SHRD	r16,r16,i		l1	11	11	
SHLD SHRD	r32,r32,i		I1	7	7	
SHLD	r64,r64,i		I1	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		I1	43	43	
BT	r,r/i	1	I1	1	1	
ВТ	m,r		I1		8	
BT	m,i	2	l1		1	
BTR BTS BTC	r,r/i	2	I1	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	

1	SETcc	r		I1	2	1	
Control transfer instructions		m					
Control transfer instructions	CLC STC CMC			I1			
Short/near   far	CLD STD				3	3	
JMP         short/near far         1         12         3         3         bytes block Not in x64 mode 8 if ≠2 jumps in 16 bytes block 9 if ≠2 jumps in 16 bytes block 4 mode 8 if ≠2 jumps in 16 bytes block 4 mode 7 mode 8 mode 9 mode 7 mode 9 mode 7 mode 9 mode 7 mode 9 mo	Control transfer instruc	tions					
JMP	JMP	short/near	1	12	3	3	8 if >2 jumps in 16 bytes block
JMP	JMP	far			58		1
JMP JMP JMP (m(rar)) JMP (m(rar)) JMP (m(rar)) JMP (m(rar)) JMP (m(rar)) JMP (m(rar)) Short/near         3 3 3 do.         3 3 do.         3 3 do.         3 if not jumping. 3 if jumping. 4 if jum	IMP	r		12	3	3	
JMP							-
Conditional jump		, ,					uo.
3 if jumping. 8 if >2 jumps in 16 bytes block		' '				1_3_8	1 if not jumping
LOOP LOOP(N)E Short Short LOOP(N)E Short S	Conditional jump	Shortmean			1-0-0	1.0.0	3 if jumping. 8 if >2 jumps in 16
LOOP LOOP(N)E Short Short LOOP(N)E Short S	J(E/R)CXZ	short			1-3-8	1-3-8	do.
LOOP(N)E       short       25       25       8 if >2 jumps in 16 bytes block not in x64 mode         CALL       far       72       72       Not in x64 mode       8 if >2 jumps in 16 bytes block       Not in x64 mode       8 if >2 jumps in 16 bytes block       do.       do.       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       do.       st if >2 jumps in 16 bytes block       do.       st if >2 jumps in 16 bytes block       do.       st if >2 jumps in 16 bytes block       do.       st if >2 jumps in 16 bytes block <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>							
String instructions   CALL							
CALL       near far       3       3       bytes block         CALL       far       72       72       Not in x64 mode       8 if >2 jumps in 16 bytes block         CALL       m(near)       4       3       do.         CALL       m(far)       72       72       8 if >2 jumps in 16 bytes block         RETN       i       3       3       do.         RETF       i       39       39       39         RETF       i       39       39       Not in x64 mode         INTO       13       Not in x64 mode       Not in x64 mode         String instructions       1       3n+22       1-2         SCOSB/W/D/Q       1-2       Small: 2n+2, Big: 6 bytes per clock         MOVSB/W/D/Q       2       Small: 2n+2, Big: 6 bytes per clock         SCASB/W/D/Q       1       1	2001 (11)2	0.1010					8 if >2 iumps in 16
CALL	CALL	near			3	3	
CALL       r       m(near)       3       3       bytes block         CALL       m(far)       72       72       8 if >2 jumps in 16         RETN       3       3       3       3       4       3       3       3       4       3       3       3       4       3       60.       8 if >2 jumps in 16       6       60.	CALL	far			72	72	Not in x64 mode
CALL       m(near) m(far)       4       3 do.         CALL       m(far)       72       72         RETN       3 3 3 do.       8 if >2 jumps in 16 bytes block         RETN       3 3 3 do.       4 do.         RETF       39 39 39 39 39 do.       39 39 do.         BOUND       r,m       13 Not in x64 mode Not in x64 mode         Not in x64 mode       7 Not in x64 mode         String instructions       1 3n+22 Small: 2n+2, Big: 6 bytes per clock         STOSB/W/D/Q       2 Small: 2n+2, Big: 6 bytes per clock         MOVSB/W/D/Q       2 Small: 2n+45, Big: 6 bytes per clock         SCASB/W/D/Q       1							8 if >2 jumps in 16
CALL       m(far)       72       72       8 if >2 jumps in 16 bytes block         RETN       i       3       3       3 do.         RETF       i       39       39       39         BOUND       r,m       13       Not in x64 mode         INTO       7       Not in x64 mode         String instructions         LODSB/W/D/Q       1       3n+22         STOSB/W/D/Q       3n+22       5mall: 2n+2, Big: 6 bytes per clock         MOVSB/W/D/Q       2       5mall: 2n+45, Big: 6 bytes per clock         SCASB/W/D/Q       1       1	CALL	r			3	3	bytes block
RETN	CALL	m(near)			4	3	do.
RETN RETN RETR RETF RETF RETF BOUND IT,m RETIGHT RETF RETF RETF RETF RETF RETF RETF RET	CALL	m(far)			72	72	
RETN       i       3       3       30       39       30       Not in x64 mode       7       Not in x64 mode       7       Not in x64 mode       31       31       Not in x64 mode       31       31       31       Not in x64 mode       31       31       31       31       31       Not in x64 mode       31	DETN				2	2	
RETF RETF BOUND IT,m BOUND INTO  String instructions LODSB/W/D/Q REP LODSB/W/D/Q REP STOSB/W/D/Q REP STOSB/W/D/Q REP MOVSB/W/D/Q							-
RETF		1					do.
BOUND INTO 13 Not in x64 mode 7 Not in x64 mode 7 Not in x64 mode 8 Not in x64 mode 15 Not in x64 mode 16 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 16 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 17 Not in x64 mode 18 Not in x64 mode 17 Not in x64 mode 18 Not in x64 mode 18 Not in x64 mode 18 Not in x64 mode 19 No							
Not in x64 mode		-			39		Notice Of seconds
String instructions  LODSB/W/D/Q REP LODSB/W/D/Q STOSB/W/D/Q REP STOSB/W/D/Q REP STOSB/W/D/Q  MOVSB/W/D/Q REP MOVSB/W/D/Q  STABIL: 2n+45, Big: 6 bytes per clock  SCASB/W/D/Q  1  2  Small: 2n+45, Big: 6 bytes per clock  1		r,m					
1	INTO					/	Not in x64 mode
REP LODSB/W/D/Q STOSB/W/D/Q REP STOSB/W/D/Q REP STOSB/W/D/Q MOVSB/W/D/Q REP MOVSB/W/D/Q Small: 2n+2, Big: 6 bytes per clock  2 Small: 2n+45, Big: 6 bytes per clock  SCASB/W/D/Q 1	String instructions						
STOSB/W/D/Q REP STOSB/W/D/Q  MOVSB/W/D/Q REP MOVSB/W/D/Q  REP MOVSB/W/D/Q  SCASB/W/D/Q  1-2  Small: 2n+2,  Big: 6 bytes  per clock  SCASB/W/D/Q  1	LODSB/W/D/Q					1	
REP STOSB/W/D/Q  MOVSB/W/D/Q  REP MOVSB/W/D/Q  Small: 2n+2, Big: 6 bytes per clock  2  Small: 2n+45, Big: 6 bytes per clock  SCASB/W/D/Q  1	REP LODSB/W/D/Q					3n+22	
MOVSB/W/D/Q REP MOVSB/W/D/Q Small: 2n+45, Big: 6 bytes per clock  SCASB/W/D/Q  1	STOSB/W/D/Q					1-2	
MOVSB/W/D/Q REP MOVSB/W/D/Q Small: 2n+45, Big: 6 bytes per clock  SCASB/W/D/Q  1	REP STOSB/W/D/Q					Small: 2n+2,	
REP MOVSB/W/D/Q  Small: 2n+45, Big: 6 bytes per clock  SCASB/W/D/Q  1						Big: 6 bytes per clock	
REP MOVSB/W/D/Q  Small: 2n+45, Big: 6 bytes per clock  SCASB/W/D/Q  1	MOVSB/W/D/Q					2	
SCASB/W/D/Q Big: 6 bytes per clock						_	
						Big: 6 bytes	
	SCASB/W/D/O					1	
	REP SCASB					2.2n	

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						

						Lower precision:
FADD(P) FSUB(R)(P)	r/m	1	MB	2	1	Lat: 4, Thr: 2
FMUL(P)	r/m	1	MA	4	2	
FDIV(R)(P)	r/m		MA	15-42	15-42	
FABS		1	MB	1	1	
FCHS		1	MB	1	1	
FCOM(P) FUCOM	r/m	1	MB		1	
FCOMPP FUCOMPP		1	MB		1	
FCOMI(P) FUCOMI(P)	r	1	MB		1	
FIADD FISUB(R)	m		MB		2	
FIMUL	m				4	
FIDIV(R)	m				42	
FICOM(P)	m	1			2	
FTST		1	MB		1	
FXAM					41	
FPREM				151-171		
FPREM1				106-155		
FRNDINT				29		
Math				00		
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	112	0	1/2	
FNCLEX			· <del>-</del>		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	

140) (504	100		04.07		1 40
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	,				_
PACKUSWB	V,V	1	MB	1	1
PUNPCKH/LBW/WD/	','		2	•	
DQ	V,V	1	MB	1	1
PUNPCKH/LQDQ	V,V	1	MB	1	1
PSHUFB	· ·	1	MB	1	1
	V,V	-			-
PSHUFW	mm,mm,i	1	MB	1	1
PSHUFL/HW	x,x,i	1	MB	1	1
PSHUFD	x,x,i	1	MB	1	1
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
I II CITTO	(x)////////////////////////////////////				
Arithmetic instructions	I				
PADD/SUB(U)(S)B/W/D					
FADD/30B(0)(3)B/W/D	V,V	1	MB	1	1
PADDQ PSUBQ	V,V V,V	1	MB	1	1
PHADD(S)W	V, V	ı	IVID	'	<b>'</b>
PHSUB(S)W	V,V	3	MB	3	3
PHADDD PHSUBD	V,V V,V	3	MB	3	3
PCMPEQ/GTB/W/D	V,V V,V	1	MB	1	1
PMULL/HW PMULHUW		1	MA	3	1
PMULHRSW	V,V	1	MA	3	1
PMULUDQ	V,V	1	MA	3	1
· ·	V,V	I	IVIA		
PMADDWD	V,V			4	2
PMADDUBSW	V,V		MD	10	8
PSADBW	V,V	4	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			MD	_	_
	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,İ	1	MB	1	1
PSLL/RLDQ	x,i	1	MB	1	1

Other				
EMMS	1	MB	1	

Floating point XMN	<u>l</u> instructio	ns				
	Operands	μops	Port and Unit	Latency	Reciprocal thruoghput	Remarks
Move instructions						
MOVAPS/D	xmm,xmm	1	MB	1	1 1	
MOVAPS/D	xmm,m128	1	LD	2-3	1 1	
MOVAPS/D	m128,xmm	1	SA ST	2-3	1-2	
MOVUPS/D	xmm,m128	1	LD	2-3	1 1	
MOVUPS/D	m128,xmm	1	SA ST	2-3	1-2	
MOVSS/D	xmm,xmm	1	MB	1	1 1	
MOVSS/D	x,m32/64	1	LD	2-3	1 1	
MOVSS/D	m32/64,x	1	SA ST	2-3	1-2	
MOVHPS/D	xmm,m64			6	1 1	
MOVLPS/D	xmm,m64			6	1 1	
MOVHPS/D	m64,xmm			6	1-2	
MOVLPS/D	m64,xmm			2	1-2	
MOVLHPS MOVHLPS	xmm,xmm	1	MB	1	1 1	
MOVMSKPS/D	r32,xmm			3	1 1	
MOVNTPS/D	m128,xmm			~300	2.5	
SHUFPS	xmm,xmm,i	1	MB	1	1 1	
SHUFPD	xmm,xmm,i	1	MB	1	1 1	
MOVDDUP	xmm,xmm	1	MB	1	1 1	
MOVSH/LDUP	xmm,xmm	1	MB	1	1 1	
UNPCKH/LPS	xmm,xmm	1	MB	1	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	
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	· ·	'	MA	3	1	
MULPD	xmm,xmm		MA	4	2	
DIVSS	xmm,xmm			15-22	15-22	
	xmm,xmm		MA			
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	
Lagia						
Logic	\(\max_1\max_2\max	4	MD	4	4	
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**

The Cope of the line		
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	µops	Port	Latency	Reciprocal thruogh- put	Remarks
Move instructions						
MOV	r,r	1	12	1	1	
MOV	r,i	1	l12	1	1/2	
						Latency 4 on pointer
MOV	r,m	1	LD	2	1	register
MOV	m,r	1	SA, ST	2	1.5	
MOV	m,i	1	SA, ST		1.5	
MOV	r,sr		l12		1/2	
MOV	m,sr				1.5	
MOV	sr,r			20	20	

	I	I	I			
MOV	sr,m			20	20	
MOVNTI	m,r		SA, ST	2	1.5	
MOVSX MOVZX	r,r	1	l12	1	1/2	
MOVSXD	r64,r32	1		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	l12	3	1.5	
XCHG	r,m			18	18	Implicit lock
XLAT	m	3	LD, I1	6	2	
PUSH	r	1	SA, ST		1-2	
PUSH	i	1	SA, ST		1-2	
PUSH	m		LD, SA, ST		2	
PUSH	sr		, ,		6	
PUSHF(D/Q)		3		2	2	
PUSHA(D)		9		_	15	Not in x64 mode
POP	r	2	LD		1.25	1100 210 1
POP	(E/R)SP	_			4	
POP	m	3			2	
POP	sr				11	
POPF(D/Q)	31	3			1	
POPA(D)		16			12	Not in x64 mode
LAHF		10	l1	4		Not in x04 mode
SAHF			II	1 1	1	
		1	11	_	1	Not in vC4 mode
SALC		2		10	6	Not in x64 mode
LEA	r m	1	SA	1	1	Extra latency to other ports
BSWAP	r,m	1	12	1		ports
LDS LES LFS LGS LSS	r	'	12		1	
LDS LES LFS LGS LSS	m	12		28	28	
PREFETCHNTA	m m	1	LD	20	1	
PREFETCHT0/1/2	m m	1	LD		1	
LFENCE MFENCE	""	'	LD		ı	
SFENCE					15	
0. 2.102					10	
Arithmetic instructions						
ADD SUB	r,r/i	1	l12	1	1/2	
ADD SUB	r,m	2	LD 112	•	1	
ADD SUB	m,r/i	3	LD I12 SA ST	5	2	
ADC SBB	r,r/i	1	11	1	1	
ADC SBB		2	LD I1	•	1	
ADC SBB	r,m m, r/i	3		5	2	
	m,r/i		LD I1 SA ST	5		
CMP	r,r/i	1	I12	1	1/2	
CMP	m,r/i	2	LD I12		1/0	
INC DEC NEG NOT	r	1	I12	1	1/2	
INC DEC NEG NOT	m	3	LD I12 SA ST	5	0.7	Notice and the
AAA		12			37	Not in x64 mode
AAS		12			22	Not in x64 mode
DAA		14			22	Not in x64 mode

DAS		14			24	Not in x64 mode
AAD		7			24	Not in x64 mode
AAM		13			31	Not in x64 mode
MUL IMUL	r8	1	12	2		
MUL IMUL	r16	3	12	3		
MUL IMUL	r32	3	12	3		
Wide invide	102					Extra latency to other
MUL IMUL	r64	3	MA	8	8	ports
IMUL	r16,r16	1	12	2	1	
IMUL	r32,r32	1	12	2	1	
	102,102			_	•	Extra latency to other
IMUL	r64,r64	1	MA	5	2	ports
IMUL	r16,r16,i	1	12	2	1	
IMUL	r32,r32,i	1	12	2	1	
	102,102,1	•		_		Extra latency to other
IMUL	r64,r64,i	1	MA	5	2	ports
DIV	r8	-	MA	22-24	22-24	
DIV	r16		MA	24-28	24-28	
DIV	r32		MA	22-30	22-30	
DIV	r64		MA	145-162	145-162	
IDIV	r8		MA	21-24	21-24	
IDIV	r16		MA	24-28	24-28	
IDIV	r32		MA	18-26	18-26	
IDIV	r64		MA	182-200	182-200	
CBW CWDE CDQE	104	1	I2	102-200		
		1	12	1 1	1	
CWD CDQ CQO		<b>'</b>	12	'	1	
Logic instructions						
AND OR XOR	r,r/i	1	l12	1	1/2	
AND OR XOR		2	LD I12		1/2	
	r,m			_		
AND OR XOR	m,r/i	3	LD I12 SA ST	5	2	
TEST	r,r/i	1	I12	1	1/2	
TEST	m,r/i	2	LD 112	_	1	
SHR SHL SAR	r,i/cl	1	l12	1 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	l1	42	42	
BT	r,r/i	1	I1	1	1	
BT	m,r	6	I1		8	
ВТ	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	
SETcc	m	2			2	
						·

CLC STC CMC		3	I1	3	3	
CLD STD		3	I1	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	3	50	Not in x64 mode
	10.	• •				8 if >2 jumps in 16
JMP	r	2	12	3	3	bytes block
JMP	m(near)	2		3	3	do.
JMP	m(far)	17			42	1 if not jumping.
						3 if jumping.
0 1111	,	4	10	4.0.0	4.0.0	8 if >2 jumps in 16 bytes block
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 5		1-3-8 24	1-3-8	
LOOP(N)E	short	5		24	24	9 if >2 iumne in 16
CALL	near	2		3	3	8 if >2 jumps in 16 bytes block
CALL	far	17			58	Not in x64 mode
						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	
RETN		3		3	3	8 if >2 jumps in 16 bytes block
RETN	i	4		3	3	do.
RETF	'	20		3	49	do.
RETF	i	20			49	
BOUND	r,m	9			13	Not in x64 mode
INTO	,,,,,	3			7	Not in x64 mode
String instructions						
LODSB/W/D/Q		2			1	
REP LODSB/W/D/Q		3n			3n+27	
STOSB/W/D/Q		1			1-2 Small:	
					n+40, Big:	
					6-7	
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	
INEL GONOD					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 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 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		
Math						
FSCALE		22		37		
FXTRACT		13		57		
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	

l			1	l <u>-</u>	
MOVDQA	x, m128	1	LD	2	1 1
MOVDQA	m128, x	1	SA ST	2	1-2
MOVDQU	m128, x	1	SA ST	2	1-2
MOVDQU	x, m128	1	LD	2	1 1
LDDQU	x, m128	1	LD	2	1 1
MOVDQ2Q	mm, x	1	MB	1	1
MOVQ2DQ	x,mm	1	MB	1	1
MOVNTQ	m64,mm	2		~360	2
MOVNTDQ	m128,x	2		~360	2
MOVNTDQA	x,m128	1		2	1 1
PACKSSWB/DW	,				
PACKUSWB	V,V	1	MB	1	1
PACKUSDW	x,x	1	MB	1	1 1
PUNPCKH/LBW/WD/DQ	, v,v	1	MB	1	1
PUNPCKH/LQDQ	v,v	1	MB	1	1
PSHUFB	v,v	1	MB	1	1
PSHUFW	mm,mm,i	1	MB	1	1
PSHUFL/HW	x,x,i	1	MB	1	1
PSHUFD	x,x,i	1	MB	1	1 1
PBLENDVB	x,x,xmm0	1	MB	2	2
PBLENDW	x,x,i	1	MB	1	1
PALIGNR	x,x,i x,x,i	1	MB	1	1 1
MASKMOVQ	mm,mm	'	IVID	'	1-2
MASKMOVDQU					1-2
PMOVMSKB	X,X			3	1 1
	r32,(x)mm	4	MD		
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		4	MD	1	1
BQ/WD/WQ/DQ	X,X	1	MB	ı	<b>'</b>
Arithmetic instructions					
PADD/SUB(U)(S)B/W/D					
PADD/30B(0)(3)B/W/D	V,V	1	MB	1	1
PADDQ PSUBQ	V,V V,V	1	MB	1	1 1
PHADD(S)W	V, V	'	IVID	'	' '
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
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 1
PAVGB/W	V,V	1	MB	1	1
I -	,- ,-			l <sup>-</sup>	· · ·

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 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 1	
MOVNTPS/D	m128,x	2		~360	1-2	
SHUFPS	x,x,i	1	MB	1	1 1	
SHUFPD	x,x,i	1	MB	1	1 1	
MOVDDUP	x,x	1	MB	1	1 1	
MOVSH/LDUP	x,x	1	MB	1	1 1	
UNPCKH/LPS	x,x	1	MB	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	

la			I	1 -	1 1
CVTSS2SD	X,X	1		2	
CVTDQ2PS	x,x	1	MB	3	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	1	MBfadd	2	1
ADDSUBPD	x,x	1	MBfadd	2	1
HADDPS HSUBPS	x,x	3	MBfadd	5	3
HADDPD HSUBPD	x,x	3	MBfadd	5	3
MULSS	X,X	1	MA	3	1 1
MULSD	x,x	1	MA	4	2
MULPS	x,x	1	MA	3	1 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 1
CMPccPS/D	X,X	1	MBfadd	2	1
COMISS/D UCOMISS/D	,			_	
	x,x	1	MBfadd	3	1 1
MAXSS/D MINSS/D	X,X	1	MBfadd	2	1 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					
Logic ANDPS/D	vv	1	MB	1	1
ANDES/D	x,x	1	IVID	<b>'</b>	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**

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						