Packed Arithmetic Intrinsics

Intrinsic	Operation	Corresponding instruction
_mm_add_ss	Add	ADDSS
_mm_add_ps	Add	ADDPS
_mm_sub_ss	Subtract	SUBSS
_mm_sub_ps	Subtract	SUBPS
_mm_mul_ss	Multiply	MULSS
_mm_mul_ps	Multiply	MULPS
_mm_div_ss	Divide	DIVSS
_mm_div_ps	Divide	DIVPS
_mm_sqrt_ss	Square root	SQRTSS
_mm_sqrt_ps	Square root	SQRTPS
_mm_rcp_ss	Reciprocal	RCPSS
_mm_rcp_ps	Reciprocal	RCPPS
_mm_rsqrt_ss	Reciprocal square root	RSQRTSS
_mm_rsqrt_ps	Reciprocal squared root	RSQRTPS
_mm_min_ss	Minimum	MINSS
_mm_min_ps	Minimum	MINPS
_mm_max_ss	Maximum	MAXSS
_mm_max_ps	Maximum	MAXPS

Logical Intrinsics

Intrinsic	Operation	Corresponding instruction
_mm_and_ps	Bitwise AND	ANDPS
_mm_andnot_ps	Logical AND NOT	ANDNPS
_mm_or_ps	Bitwise OR	ORPS
_mm_xor_ps	Bitwise Exclusive OR	XORPS

Compare Intrinsics

Intrinsic	Comparison	Corresponding instruction
_mm_cmpeq_ss	Equal	CMPEQSS
_mm_cmpeq_ps	Equal	CMPEQPS
_mm_cmplt_ss	Less than	CMPLTSS
_mm_cmplt_ps	Less than	CMPLTPS
_mm_cmple_ss	Less than or equal	CMPLESS
_mm_cmple_ps	Less than or equal	CMPLEPS
_mm_cmpgt_ss	Greater than	CMPLTSS
_mm_cmpgt_ps	Greater than	CMPLTPS
_mm_cmpge_ss	Greater than or equal	CMPLESS
_mm_cmpge_ps	Greater than or equal	CMPLEPS
_mm_cmpneq_ss	Not equal	CMPNEQSS

_mm_cmpneq_ps	Not equal	CMPNEQPS
_mm_cmpnlt_ss	Not less than	CMPNLTSS
_mm_cmpnlt_ps	Not less than	CMPNLTPS
_mm_cmpnle_ss	Not less than or equal	CMPNLESS
_mm_cmple_ps	Not less than or equal	CMPNLEPS
_mm_cmpngt_ss	Not greater than	CMPNLTSS
_mm_cmpngt_ps	Not greater than	CMPNLTPS
_mm_cmpnge_ss	Not greater than or equal	CMPNLESS
_mm_cmpnge_ps	Not greater than or equal	CMPNLEPS
_mm_cmpord_ss	Ordered	CMPORDSS
_mm_cmpord_ps	Ordered	CMPORDPS
_mm_cmpunord_ss	Unordered	CMPUNORDSS
_mm_cmpunord_ps	Unordered	CMPUNORDPS
_mm_comieq_ss	Equal	COMISS
_mm_comilt_ss	Less than	COMISS
_mm_comile_ss	Less than or equal	COMISS
_mm_comigt_ss	Greater than	COMISS
_mm_comige_ss	Greater than or equal	COMISS
_mm_comineq_ss	Not equal	COMISS
_mm_ucomieq_ss	Equal	UCOMISS
_mm_ucomilt_ss	Less than	UCOMISS
_mm_ucomile_ss	Less than or equal	UCOMISS
_mm_ucomigt_ss	Greater than	UCOMISS
_mm_ucomige_ss	Greater than or equal	UCOMISS
_mm_ucomineq_ss	Not equal	UCOMISS

Conversion Operations

Intrinsic name	Corresponding instruction
_mm_cvtss_si32	CVTSS2SI
_mm_cvtps_pi32	CVTPS2PI
_mm_cvttss_si32	CVTTSS2SI
_mm_cvttps_pi32	CVTTPS2PI
_mm_cvtsi32_ss	CVTSI2SS
_mm_cvtpi32_ps	CVTTPS2PI
_mm_cvtpi16_ps	Composite
_mm_cvtpu16_ps	Composite
_mm_cvtpi8_ps	Composite
_mm_cvtpu8_ps	Composite
_mm_cvtpi32x2_ps	Composite
_mm_cvtps_pi16	Composite
_mm_cvtps_pi8	Composite

Miscellaneous Intrinsics

Intrinsic	Operation	Corresponding instruction
_mm_shuffle_ps	Shuffles	SHUFPS
_mm_shuffle_pi16	Shuffles	PSHUFW
_mm_unpackhi_ps	Unpacks high	UNPCKHPS
_mm_unpacklo_ps	Unpacks low	UNPCKLPS
_mm_loadh_pi	Loads high	MOVHPS reg, mem
_mm_storeh_pi	Stores high	MOVHPS mem, reg
_mm_movehl_ps	Moves high to low	MOVHLPS
_mm_movelh_ps	Moves low to high	MOVLHPS
_mm_loadl_pi	Loads low	MOVLPS reg, mem
_mm_storel_pi	Stores low	MOVLPS mem, reg
_mm_movemask_ps	Creates four-bit mask	MOVMSKPS
_mm_getcsr	Returns MXCSR register	STMXCSR
_mm_setcsr	Sets MXCSR register	LDMXCSR

Memory and Initialization Load Operations

Intrinsic	Operation	Corresponding instruction
_mm_load_ss	Loads the low value and clears the three high values	MOVSS
_mm_load1_ps	Loads one value into all four words	MOVSS + Shuffling
_mm_load_ps	Loads four values, address aligned	MOVAPS
_mm_loadu_ps	Loads four values, address unaligned	MOVUPS
_mm_loadr_ps	Loads four values, in reverse order	MOVAPS + Shuffling

Memory and Initialization Set Operations

Intrinsic	Operation	Corresponding instruction
_mm_set_ss	Sets the low value and clears the three high values	Composite
_mm_set1_ps	Sets all four words with the same value	Composite
_mm_set_ps	Sets four values, address aligned	Composite
_mm_setr_ps	Sets four values, in reverse order	Composite
_mm_setzero_ps	Clears all four values	Composite

Memory and Initialization Store Operations

Intrinsic	Operation	Corresponding instruction
_mm_store_ss	Stores the low value	MOVSS

_mm_store1_ps	Stores the low value across all four words	MOVSS + Shuffling
_mm_store_ps	Stores four values, address aligned	MOVAPS
_mm_storeu_ps	Stores four values, address unaligned	MOVUPS
_mm_storer_ps	Stores four values, in reverse order	MOVAPS + Shuffling
_mm_move_ss	Sets the low word, and passes in three high values	MOVSS

Integer Intrinsics

Intrinsic	Operation	Corresponding instruction
_mm_extract_pi16	Extracts one of four words	PEXTRW
_mm_insert_pi16	Inserts a word	PINSRW
_mm_max_pi16	Computes the maximum	PMAXSW
_mm_max_pu8	Computes the maximum, unsigned	PMAXUB
_mm_min_pi16	Computes the minimum	PMINSW
_mm_min_pu8	Computes the minimum, unsigned	PMINUB
_mm_movemask_pi8	Creates an 8-bit mask	PMOVMSKB
_mm_mulhi_pu16	Multiplies, returning high bits	PMULHUW
_mm_shuffle_pi16	Returns a combination of four words	PSHUFW
_mm_maskmove_si64	Conditionally stores data	MASKMOVQ
_mm_avg_pu8	Computes rounded average	PAVGB
_mm_avg_pu16	Computes rounded average	PAVGW
_mm_sad_pu8	Computes sum of absolute differences	PSADBW

Cache support

void _mm_prefetch(char * p , int i); PREFETCH

Loads one cache line of data from address p to a location closer to the processor. The value i specifies the type of prefetch operation: the constants _MM_HINT_T0, _MM_HINT_T1, _MM_HINT_T2, and _MM_HINT_NTA, corresponding to the type of prefetch instruction, should be used.

void _mm_stream_pi(__m64 * p , __m64 a); MOVNTQ

Stores the data in a to the address p without polluting the caches. This intrinsic requires you to empty the multimedia state for the MMX register. See Understanding the EMMS Instruction section.

void _mm_stream_ps(float * p , __m128 a); MOVNTPS

Stores the data in a to the address p without polluting the caches. The address must be 16-byte

aligned.

void _mm_sfence(void); SFENCE

Guarantees that every preceding store is globally visible before any subsequent store.