Juego de Instrucciones SSE

(Conversión)

De Integers a Floating Point

CVTPI2PS Op1, Op2	Convert Parallel Ints to Parallel Scalar

Op1 contiene 4 **single precision 32-bit** floating point values : [XMM]
Op2 contiene 2 **32-bit integer** values : [MM | M64]

Op1 [0] = (float) Op2 [0]
Op1 [1] = (float) Op2 [1]
Op1 [2] = Op1 [2]
Op1 [3] = Op1 [3]

CVTPI2PD Op1, Op2	Convert Parallel Ints to Parallel Scalar

Op1 contiene 2 **double precision 64-bit** floating point values : [XMM]
Op2 contiene 2 **32-bit integer** values : [MM | M64]

Op1 [0] = (float 64-b) Op2 [0]
Op1 [1] = (float 64-b) Op2 [1]

CVTDQ2PS Op1, Op2	Convert Parallel Ints

Op1 contiene 4 **single precision 32-bit** floating point values : [XMM]
Op2 contiene 4 **32-bit integer** values : [XMM | M128]

Op1 [0] = (float) Op2 [0]
Op1 [1] = (float) Op2 [1]
Op1 [2] = (float) Op2 [2]
Op1 [3] = (float) Op2 [3]

CVTDQ2PD Op1, Op2	Convert Parallel Scalars to Parallel Ints

Op1 contiene 2 **single precision 64-bit** floating point values : [XMM]
Op2 contiene 2 **32-bit integer** values : [XMM | M64]

Op1 [0] = (float) Op2 [0]
Op1 [1] = (float) Op2 [1]

CVTSI2SS Op1, Op2	Convert Single Int to Single Scalars

Op1 contiene 4 **single precision 32-bit** floating point values : [XMM]
Op2 contiene 1 **32-bit integer** values : [REGx86 | M32]

Op1 [0] = (float) Op2 [0]
Op1 [1] = Op1 [1]
Op1 [2] = Op1 [2]
Op1 [3] = Op1 [3]

CVTSI2SD Op1, Op2	Convert Single Int to Single Scalar

Op1 contiene 2 **64-bit integer** values : [XMM]
Op2 contiene 1 **single precision 32-bit** floating point values : [REGx86 | M32]

Op1 [0] = (float 64-b) Op2 [0]
Op1 [1] = Op1 [1]

De Floating Point a Integers sin Truncamiento

CVTPS2PI Op1, Op2	Convert Parallel Scalars to Parallel Ints

Op1 contiene 2 **32-bit integer** values : [MM]
Op2 contiene 2 **single precision 32-bit** floating point values : [XMM | M64]

Op1 [0] = (long) Op2 [0]
Op1 [1] = (long) Op2 [1]

CVTPS2DQ Op1, Op2	Convert Parallel Scalars to Parallel Ints

Op1 contiene 4 **32-bit integer** values : [XMM]
Op2 contiene 4 **single precision 32-bit** floating point values : [XMM | M128]

Op1 [0] = (long) Op2 [0]
Op1 [1] = (long) Op2 [1]
Op1 [2] = (long) Op2 [2]
Op1 [3] = (long) Op2 [3]

CVTPD2DQ Op1, Op2	Convert Parallel Scalars to Parallel Ints

Op1 contiene 4 **32-bit integer** values : [XMM]
Op2 contiene 2 **single precision 64-bit** floating point values : [XMM | M128]

Op1 [0] = (long) Op2 [0]
Op1 [1] = (long) Op2 [1]
Op1 [2] = 0
Op1 [3] = 0

CVTPD2PI Op1, Op2	Convert Parallel Scalars to Parallel Ints

Op1 contiene 2 **32-bit integer** values : [MM]
Op2 contiene 2 **single precision 64-bit** floating point values : [XMM | M128]

Op1 [0] = (long) Op2 [0]
Op1 [1] = (long) Op2 [1]

CVTSS2SI Op1, Op2	Convert Single Scalar to Single Int

Op1 contiene 1 **32-bit integer** values : [REGx86]
Op2 contiene 1 **single precision 32-bit** floating point values : [XMM | M32]

Op1 [0] = (long) Op2 [0]

CVTSD2SI Op1, Op2	Convert Single Scalar to Single Int

Op1 contiene 1 **32-bit integer** values : [REGx86]
Op2 contiene 1 **double precision 64-bit** floating point values : [XMM | M64]

Op1 [0] = (long) Op2 [0]

De Floating Point a Integers con Truncamiento

CVTTPS2PI Op1, Op2	Convert Parallel Scalars to Parallel Ints

Op1 contiene 2 **32-bit integer** values : [MM]
Op2 contiene 2 **single precision 32-bit** floating point values : [XMM | M64]

Op1 [0] = (long) Op2 [0]
Op1 [1] = (long) Op2 [1]

CVTTPD2PI Op1, Op2 Convert Parallel Scalars to Parallel Ints

Op1 contiene 2 **32-bit integer** values : [MM]
Op2 contiene 2 **double precision 64-bit** floating point values : [XMM | M128]

Op1 [0] = (long) Op2 [0]
Op1 [1] = (long) Op2 [1]

CVTTPD2DQ Op1, Op2 Convert Parallel Scalars to Parallel Ints

Op1 contiene 4 **32-bit integer** values : [XMM]
Op2 contiene 2 **double precision 64-bit** floating point values : [XMM | M128]

Op1 [0] = (long) Op2 [0]
Op1 [1] = (long) Op2 [1]
Op1 [2] = 0
Op1 [3] = 0

CVTTPS2DQ Op1, Op2 Convert Parallel Scalars to Parallel Ints

Op1 contiene 4 **32-bit integer** values : [XMM]
Op2 contiene 4 **single precision 32-bit** floating point values : [XMM | M128]

Op1 [0] = (long) Op2 [0]
Op1 [1] = (long) Op2 [1]
Op1 [2] = (long) Op2 [2]
Op1 [3] = (long) Op2 [3]

CVTTSS2SI Op1, Op2 Convert Single Scalar to Single Int

Op1 contiene 1 **32-bit integer** values : [REGx86]
Op2 contiene 4 **single precision 32-bit** floating point values : [XMM | M128]

Op1 [0] = (long) Op2 [0]

CVTTSD2SI Op1, Op2 Convert Single Scalar to Single Int

Op1 contiene 1 **32-bit integer** values : [REGx86]
Op2 contiene 1 **double precision 64-bit** floating point values : [XMM | M64]

Op1 [0] = (long) Op2 [0]

De Floating Point a Floating Point

CVTPD2PS Op1, Op2	Convert 64 bits to 32 bits F.P.

Op1 contiene 4 **single precision 32-bit** floating point values : [XMM]
Op2 contiene 2 **single precision 64-bit** floating point values : [XMM | M128]

Op1 [0] = (float) Op2 [0]
Op1 [1] = (float) Op2 [1]
Op1 [2] = 0
Op1 [3] = 0

CVTPS2PD Op1, Op2	Convert 32 bits to 64 bits F.P.

Op1 contiene 2 **single precision 64-bit** floating point values : [XMM]
Op2 contiene 2 **single precision 32-bit** floating point values : [XMM | M64]

Op1 [0] = (float 64-b) Op2 [0]
Op1 [1] = (float 64-b) Op2 [1]

CVTSD2SS Op1, Op2	Convert 64 bits to 32 bits F.P.

Op1 contiene 4 **single precision 32-bit** floating point values : [XMM]
Op2 contiene 2 **single precision 64-bit** floating point values : [XMM | M64]

Op1 [0] = (float) Op2 [0]
Op1 [1] = Op1 [1]
Op1 [2] = Op1 [2]
Op1 [3] = Op1 [3]

CVTSS2SD Op1, Op2	Convert 32 bits to 64 bits F.P.

Op1 contiene 2 **single precision 64-bit** floating point values : [XMM]
Op2 contiene 1 **single precision 32-bit** floating point values : [XMM | M32]

Op1 [0] = (float 64-b) Op2 [0]
Op1 [1] = Op1 [1]