Arm MVE Intrinsics

Reference for ACLE Q2 2020

Non-Confidential

Issue Q220-00
101809

Arm MVE Intrinsics

Reference

Release information

Document history

Issue	Date	Confidentiality	Change
Q219-00	30 June 2019	Non-Confidential	Version ACLE Q2 2019.
Q319-00	30 September 2019	Non-Confidential	Version ACLE Q3 2019
Q419-00	31 December 2019	Non-Confidential	Version ACLE Q4 2019
Q220-00	30 May 2020	Non-Confidential	Version ACLE Q2 2020

Non-Confidential Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Product Status

The information in this document is final, that is for a developed product.

Web Address

http://www.arm.com

About this document

This document is complementary to the main Arm C Language Extensions (ACLE) specification, which can be found on developer.arm.com.

List of Intrinsics

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vcreateq_f16(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
float32x4_t [__arm_]vcreateq_f32(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
int8x16_t [__arm_]vcreateq_s8(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
int16x8_t [__arm_]vcreateq_s16(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
int32x4_t [__arm_]vcreateq_s32(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
int64x2_t [__arm_]vcreateq_s64(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
uint8x16_t [__arm_]vcreateq_u8(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
uint16x8_t [__arm_]vcreateq_u16(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
uint32x4_t [__arm_]vcreateq_u32(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
uint64x2_t [__arm_]vcreateq_u64(uint64_t a, uint64_t b)	a -> [Rt, Rt2] b -> [Rt3, Rt4]	VMOV Qd[2],Qd[0],Rt3,Rt VMOV Qd[3],Qd[1],Rt4,Rt2	Qd -> result	MVE
uint8x16_t [__arm_]vddupq[_n]_u8(uint32_t a, const int imm)	a -> Rn imm in [1,2,4,8]	VDDUP.U8 Qd,Rn,imm	Qd -> result	MVE
uint16x8_t [__arm_]vddupq[_n]_u16(uint32_t a, const int imm)	a -> Rn imm in [1,2,4,8]	VDDUP.U16 Qd,Rn,imm	Qd -> result	MVE
uint32x4_t [__arm_]vddupq[_n]_u32(uint32_t a, const int imm)	a -> Rn imm in [1,2,4,8]	VDDUP.U32 Qd,Rn,imm	Qd -> result	MVE
uint8x16_t [__arm_]vddupq[_wb]_u8(uint32_t * a, const int imm)	*a -> Rn imm in [1,2,4,8]	VDDUP.U8 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vddupq[_wb]_u16(uint32_t * a, const int imm)	*a -> Rn imm in [1,2,4,8]	VDDUP.U16 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]vddupq[_wb]_u32(uint32_t * a, const int imm)	*a -> Rn imm in [1,2,4,8]	VDDUP.U32 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]vddupq_m[_n_u8](uint8x16_t inactive, uint32_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U8 Qd,Rn,imm	Qd -> result	MVE
uint16x8_t [__arm_]vddupq_m[_n_u16](uint16x8_t inactive, uint32_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U16 Qd,Rn,imm	Qd -> result	MVE
uint32x4_t [__arm_]vddupq_m[_n_u32](uint32x4_t inactive, uint32_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U32 Qd,Rn,imm	Qd -> result	MVE
uint8x16_t [__arm_]vddupq_m[_wb_u8](uint8x16_t inactive, uint32_t * a, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U8 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vddupq_m[_wb_u16](uint16x8_t inactive, uint32_t * a, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U16 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]vddupq_m[_wb_u32](uint32x4_t inactive, uint32_t * a, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U32 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]vddupq_x[_n]_u8(uint32_t a, const int imm, mve_pred16_t p)	a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U8 Qd,Rn,imm	Qd -> result	MVE
uint16x8_t [__arm_]vddupq_x[_n]_u16(uint32_t a, const int imm, mve_pred16_t p)	a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U16 Qd,Rn,imm	Qd -> result	MVE
uint32x4_t [__arm_]vddupq_x[_n]_u32(uint32_t a, const int imm, mve_pred16_t p)	a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U32 Qd,Rn,imm	Qd -> result	MVE
uint8x16_t [__arm_]vddupq_x[_wb]_u8(uint32_t * a, const int imm, mve_pred16_t p)	*a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U8 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vddupq_x[_wb]_u16(uint32_t * a, const int imm, mve_pred16_t p)	*a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U16 Qd,Rn,imm	Qd -> result Rn -> *a	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vddupq_x[_wb]_u32(uint32_t * a, const int imm, mve_pred16_t p)	*a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDDUPT.U32 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]vdwdupq[_n]_u8(uint32_t a, uint32_t b, const int imm)	a -> Rn b -> Rm imm in [1,2,4,8]	VDWDUP.U8 Qd,Rn,Rm,imm	Qd -> result	MVE
uint16x8_t [__arm_]vdwdupq[_n]_u16(uint32_t a, uint32_t b, const int imm)	a -> Rn b -> Rm imm in [1,2,4,8]	VDWDUP.U16 Qd,Rn,Rm,imm	Qd -> result	MVE
uint32x4_t [__arm_]vdwdupq[_n]_u32(uint32_t a, uint32_t b, const int imm)	a -> Rn b -> Rm imm in [1,2,4,8]	VDWDUP.U32 Qd,Rn,Rm,imm	Qd -> result	MVE
uint8x16_t [__arm_]vdwdupq[_wb]_u8(uint32_t * a, uint32_t b, const int imm)	*a -> Rn b -> Rm imm in [1,2,4,8]	VDWDUP.U8 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vdwdupq[_wb]_u16(uint32_t * a, uint32_t b, const int imm)	*a -> Rn b -> Rm imm in [1,2,4,8]	VDWDUP.U16 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]vdwdupq[_wb]_u32(uint32_t * a, uint32_t b, const int imm)	*a -> Rn b -> Rm imm in [1,2,4,8]	VDWDUP.U32 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]vdwdupq_m[_n_u8](uint8x16_t inactive, uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U8 Qd,Rn,Rm,imm	Qd -> result	MVE
uint16x8_t [__arm_]vdwdupq_m[_n_u16](uint16x8_t inactive, uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U16 Qd,Rn,Rm,imm	Qd -> result	MVE
uint32x4_t [__arm_]vdwdupq_m[_n_u32](uint32x4_t inactive, uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U32 Qd,Rn,Rm,imm	Qd -> result	MVE
uint8x16_t [__arm_]vdwdupq_m[_wb_u8](uint8x16_t inactive, uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U8 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vdwdupq_m[_wb_u16](uint16x8_t inactive, uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U16 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]vdwdupq_m[_wb_u32](uint32x4_t inactive, uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U32 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]vdwdupq_x[_n]_u8(uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U8 Qd,Rn,Rm,imm	Qd -> result	MVE
uint16x8_t [__arm_]vdwdupq_x[_n]_u16(uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U16 Qd,Rn,Rm,imm	Qd -> result	MVE
uint32x4_t [__arm_]vdwdupq_x[_n]_u32(uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U32 Qd,Rn,Rm,imm	Qd -> result	MVE
uint8x16_t [__arm_]vdwdupq_x[_wb]_u8(uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	*a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U8 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vdwdupq_x[_wb]_u16(uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	*a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U16 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]vdwdupq_x[_wb]_u32(uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	*a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VDWDUPT.U32 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]vidupq[_n]_u8(uint32_t a, const int imm)	a -> Rn imm in [1,2,4,8]	VIDUP.U8 Qd,Rn,imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vidupq[_n]_u16(uint32_t a, const int imm)	a -> Rn imm in [1,2,4,8]	VIDUP.U16 Qd,Rn,imm	Qd -> result	MVE
uint32x4_t [__arm_]vidupq[_n]_u32(uint32_t a, const int imm)	a -> Rn imm in [1,2,4,8]	VIDUP.U32 Qd,Rn,imm	Qd -> result	MVE
uint8x16_t [__arm_]vidupq[_wb]_u8(uint32_t * a, const int imm)	*a -> Rn imm in [1,2,4,8]	VIDUP.U8 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vidupq[_wb]_u16(uint32_t * a, const int imm)	*a -> Rn imm in [1,2,4,8]	VIDUP.U16 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]vidupq[_wb]_u32(uint32_t * a, const int imm)	*a -> Rn imm in [1,2,4,8]	VIDUP.U32 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]vidupq_m[_n_u8](uint8x16_t inactive, uint32_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U8 Qd,Rn,imm	Qd -> result	MVE
uint16x8_t [__arm_]vidupq_m[_n_u16](uint16x8_t inactive, uint32_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U16 Qd,Rn,imm	Qd -> result	MVE
uint32x4_t [__arm_]vidupq_m[_n_u32](uint32x4_t inactive, uint32_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U32 Qd,Rn,imm	Qd -> result	MVE
uint8x16_t [__arm_]vidupq_m[_wb_u8](uint8x16_t inactive, uint32_t * a, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U8 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vidupq_m[_wb_u16](uint16x8_t inactive, uint32_t * a, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U16 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]vidupq_m[_wb_u32](uint32x4_t inactive, uint32_t * a, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U32 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]vidupq_x[_n]_u8(uint32_t a, const int imm, mve_pred16_t p)	a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U8 Qd,Rn,imm	Qd -> result	MVE
uint16x8_t [__arm_]vidupq_x[_n]_u16(uint32_t a, const int imm, mve_pred16_t p)	a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U16 Qd,Rn,imm	Qd -> result	MVE
uint32x4_t [__arm_]vidupq_x[_n]_u32(uint32_t a, const int imm, mve_pred16_t p)	a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U32 Qd,Rn,imm	Qd -> result	MVE
uint8x16_t [__arm_]vidupq_x[_wb]_u8(uint32_t * a, const int imm, mve_pred16_t p)	*a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U8 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]vidupq_x[_wb]_u16(uint32_t * a, const int imm, mve_pred16_t p)	*a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U16 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]vidupq_x[_wb]_u32(uint32_t * a, const int imm, mve_pred16_t p)	*a -> Rn imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIDUPT.U32 Qd,Rn,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]viwdupq[_n]_u8(uint32_t a, uint32_t b, const int imm)	a -> Rn b -> Rm imm in [1,2,4,8]	VIWDUP.U8 Qd,Rn,Rm,imm	Qd -> result	MVE
uint16x8_t [__arm_]viwdupq[_n]_u16(uint32_t a, uint32_t b, const int imm)	a -> Rn b -> Rm imm in [1,2,4,8]	VIWDUP.U16 Qd,Rn,Rm,imm	Qd -> result	MVE
uint32x4_t [__arm_]viwdupq[_n]_u32(uint32_t a, uint32_t b, const int imm)	a -> Rn b -> Rm imm in [1,2,4,8]	VIWDUP.U32 Qd,Rn,Rm,imm	Qd -> result	MVE
uint8x16_t [__arm_]viwdupq[_wb]_u8(uint32_t * a, uint32_t b, const int imm)	*a -> Rn b -> Rm imm in [1,2,4,8]	VIWDUP.U8 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]viwdupq[_wb]_u16(uint32_t * a, uint32_t b, const int imm)	*a -> Rn b -> Rm imm in [1,2,4,8]	VIWDUP.U16 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]viwdupq[_wb]_u32(uint32_t * a, uint32_t b, const int imm)	*a -> Rn b -> Rm imm in [1,2,4,8]	VIWDUP.U32 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]viwdupq_m[_n_u8](uint8x16_t inactive, uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U8 Qd,Rn,Rm,imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]viwdupq_m[_n_u16](uint16x8_t inactive, uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U16 Qd,Rn,Rm,imm	Qd -> result	MVE
uint32x4_t [__arm_]viwdupq_m[_n_u32](uint32x4_t inactive, uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U32 Qd,Rn,Rm,imm	Qd -> result	MVE
uint8x16_t [__arm_]viwdupq_m[_wb_u8](uint8x16_t inactive, uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U8 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]viwdupq_m[_wb_u16](uint16x8_t inactive, uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U16 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]viwdupq_m[_wb_u32](uint32x4_t inactive, uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	inactive -> Qd *a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U32 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint8x16_t [__arm_]viwdupq_x[_n]_u8(uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U8 Qd,Rn,Rm,imm	Qd -> result	MVE
uint16x8_t [__arm_]viwdupq_x[_n]_u16(uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U16 Qd,Rn,Rm,imm	Qd -> result	MVE
uint32x4_t [__arm_]viwdupq_x[_n]_u32(uint32_t a, uint32_t b, const int imm, mve_pred16_t p)	a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U32 Qd,Rn,Rm,imm	Qd -> result	MVE
uint8x16_t [__arm_]viwdupq_x[_wb]_u8(uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	*a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U8 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint16x8_t [__arm_]viwdupq_x[_wb]_u16(uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	*a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U16 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
uint32x4_t [__arm_]viwdupq_x[_wb]_u32(uint32_t * a, uint32_t b, const int imm, mve_pred16_t p)	*a -> Rn b -> Rm imm in [1,2,4,8] p -> Rp	VMSR P0,Rp VPST VIWDUPT.U32 Qd,Rn,Rm,imm	Qd -> result Rn -> *a	MVE
int8x16_t [__arm_]vdupq_n_s8(int8_t a)	a -> Rt	VDUP.8 Qd,Rt	Qd -> result	MVE/NEON
int16x8_t [__arm_]vdupq_n_s16(int16_t a)	a -> Rt	VDUP.16 Qd,Rt	Qd -> result	MVE/NEON
int32x4_t [__arm_]vdupq_n_s32(int32_t a)	a -> Rt	VDUP.32 Qd,Rt	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vdupq_n_u8(uint8_t a)	a -> Rt	VDUP.8 Qd,Rt	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vdupq_n_u16(uint16_t a)	a -> Rt	VDUP.16 Qd,Rt	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vdupq_n_u32(uint32_t a)	a -> Rt	VDUP.32 Qd,Rt	Qd -> result	MVE/NEON
float16x8_t [__arm_]vdupq_n_f16(float16_t a)	a -> Rt	VDUP.16 Qd,Rt	Qd -> result	MVE/NEON
float32x4_t [__arm_]vdupq_n_f32(float32_t a)	a -> Rt	VDUP.32 Qd,Rt	Qd -> result	MVE/NEON
int8x16_t [__arm_]vdupq_m[_n_s8](int8x16_t inactive, int8_t a, mve_pred16_t p)	inactive -> Qd a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.8 Qd,Rt	Qd -> result	MVE
int16x8_t [__arm_]vdupq_m[_n_s16](int16x8_t inactive, int16_t a, mve_pred16_t p)	inactive -> Qd a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.16 Qd,Rt	Qd -> result	MVE
int32x4_t [__arm_]vdupq_m[_n_s32](int32x4_t inactive, int32_t a, mve_pred16_t p)	inactive -> Qd a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.32 Qd,Rt	Qd -> result	MVE
uint8x16_t [__arm_]vdupq_m[_n_u8](uint8x16_t inactive, uint8_t a, mve_pred16_t p)	inactive -> Qd a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.8 Qd,Rt	Qd -> result	MVE
uint16x8_t [__arm_]vdupq_m[_n_u16](uint16x8_t inactive, uint16_t a, mve_pred16_t p)	inactive -> Qd a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.16 Qd,Rt	Qd -> result	MVE
uint32x4_t [__arm_]vdupq_m[_n_u32](uint32x4_t inactive, uint32_t a, mve_pred16_t p)	inactive -> Qd a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.32 Qd,Rt	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vdupq_m[_n_f16](float16x8_t inactive, float16_t a, mve_pred16_t p)	inactive -> Qd a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.16 Qd,Rt	Qd -> result	MVE
float32x4_t [__arm_]vdupq_m[_n_f32](float32x4_t inactive, float32_t a, mve_pred16_t p)	inactive -> Qd a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.32 Qd,Rt	Qd -> result	MVE
int8x16_t [__arm_]vdupq_x_n_s8(int8_t a, mve_pred16_t p)	a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.8 Qd,Rt	Qd -> result	MVE
int16x8_t [__arm_]vdupq_x_n_s16(int16_t a, mve_pred16_t p)	a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.16 Qd,Rt	Qd -> result	MVE
int32x4_t [__arm_]vdupq_x_n_s32(int32_t a, mve_pred16_t p)	a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.32 Qd,Rt	Qd -> result	MVE
uint8x16_t [__arm_]vdupq_x_n_u8(uint8_t a, mve_pred16_t p)	a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.8 Qd,Rt	Qd -> result	MVE
uint16x8_t [__arm_]vdupq_x_n_u16(uint16_t a, mve_pred16_t p)	a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.16 Qd,Rt	Qd -> result	MVE
uint32x4_t [__arm_]vdupq_x_n_u32(uint32_t a, mve_pred16_t p)	a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.32 Qd,Rt	Qd -> result	MVE
float16x8_t [__arm_]vdupq_x_n_f16(float16_t a, mve_pred16_t p)	a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.16 Qd,Rt	Qd -> result	MVE
float32x4_t [__arm_]vdupq_x_n_f32(float32_t a, mve_pred16_t p)	a -> Rt p -> Rp	VMSR P0,Rp VPST VDUPT.32 Qd,Rt	Qd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMP.F16 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMP.F32 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VCMP.I8 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VCMP.I16 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VCMP.I32 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VCMP.I8 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VCMP.I16 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VCMP.I32 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VCMP.F16 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VCMP.F32 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VCMP.I8 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VCMP.I16 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VCMP.I32 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VCMP.I8 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VCMP.I16 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VCMP.I32 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I8 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
mve_pred16_t [__arm_]vcmpeqq_m[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I16 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I32 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I8 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I16 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I32 eq,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I8 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I16 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I32 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I8 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I16 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpeqq_m[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I32 eq,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMP.F16 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMP.F32 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VCMP.I8 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VCMP.I16 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VCMP.I32 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VCMP.I8 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VCMP.I16 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VCMP.I32 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
mve_pred16_t [__arm_]vcmpneq_m[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I8 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I16 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I32 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I8 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I16 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.I32 ne,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VCMP.F16 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VCMP.F32 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VCMP.I8 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VCMP.I16 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VCMP.I32 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VCMP.I8 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VCMP.I16 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VCMP.I32 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I8 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I16 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I32 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I8 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I16 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpneq_m[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.I32 ne,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMP.F16 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMP.F32 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
mve_pred16_t [__arm_]vcmpgeq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VCMP.S8 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VCMP.S16 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VCMP.S32 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S8 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S16 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S32 ge,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VCMP.F16 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VCMP.F32 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VCMP.S8 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VCMP.S16 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VCMP.S32 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S8 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S16 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgeq_m[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S32 ge,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMP.F16 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMP.F32 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VCMP.S8 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VCMP.S16 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VCMP.S32 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
mve_pred16_t [__arm_]vcmpgtq_m[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S8 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S16 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S32 gt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VCMP.F16 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VCMP.F32 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VCMP.S8 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VCMP.S16 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VCMP.S32 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S8 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S16 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpgtq_m[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S32 gt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMP.F16 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMP.F32 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VCMP.S8 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VCMP.S16 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VCMP.S32 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S8 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S16 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S32 le,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VCMP.F16 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VCMP.F32 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
mve_pred16_t [__arm_]vcmpleq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VCMP.S8 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VCMP.S16 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VCMP.S32 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S8 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S16 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpleq_m[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S32 le,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMP.F16 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMP.F32 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VCMP.S8 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VCMP.S16 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VCMP.S32 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S8 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S16 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.S32 lt,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VCMP.F16 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VCMP.F32 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VCMP.S8 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VCMP.S16 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VCMP.S32 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F16 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.F32 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
mve_pred16_t [__arm_]vcmpltq_m[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S8 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S16 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpltq_m[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.S32 lt,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VCMP.U8 cs,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VCMP.U16 cs,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VCMP.U32 cs,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq_m[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.U8 cs,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq_m[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.U16 cs,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq_m[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.U32 cs,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VCMP.U8 cs,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VCMP.U16 cs,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VCMP.U32 cs,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq_m[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.U8 cs,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq_m[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.U16 cs,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmpcsq_m[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.U32 cs,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VCMP.U8 hi,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VCMP.U16 hi,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VCMP.U32 hi,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq_m[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.U8 hi,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq_m[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.U16 hi,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq_m[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMPT.U32 hi,Qn,Qm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VCMP.U8 hi,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VCMP.U16 hi,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VCMP.U32 hi,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq_m[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.U8 hi,Qn,Rm VMRS Rd,P0	Rd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
mve_pred16_t [__arm_]vcmphiq_m[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.U16 hi,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vcmphiq_m[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VCMPT.U32 hi,Qn,Rm VMRS Rd,P0	Rd -> result	MVE
int8x16_t [__arm_]vminq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VMIN.S8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vminq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMIN.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vminq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMIN.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vminq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VMIN.U8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vminq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VMIN.U16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vminq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VMIN.U32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vminq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vminq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vminq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vminq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vminq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vminq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vminq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vminq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vminq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vminq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vminq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vminq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINT.U32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vminaq[_s8](uint8x16_t a, int8x16_t b)	a -> Qda b -> Qm	VMINA.S8 Qda,Qm	Qda -> result	MVE
uint16x8_t [__arm_]vminaq[_s16](uint16x8_t a, int16x8_t b)	a -> Qda b -> Qm	VMINA.S16 Qda,Qm	Qda -> result	MVE
uint32x4_t [__arm_]vminaq[_s32](uint32x4_t a, int32x4_t b)	a -> Qda b -> Qm	VMINA.S32 Qda,Qm	Qda -> result	MVE
uint8x16_t [__arm_]vminaq_m[_s8](uint8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINAT.S8 Qda,Qm	Qda -> result	MVE
uint16x8_t [__arm_]vminaq_m[_s16](uint16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINAT.S16 Qda,Qm	Qda -> result	MVE
uint32x4_t [__arm_]vminaq_m[_s32](uint32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINAT.S32 Qda,Qm	Qda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8_t [__arm_]vminvq[_s8](int8_t a, int8x16_t b)	a -> Rda b -> Qm	VMINV.S8 Rda,Qm	Rda -> result	MVE
int16_t [__arm_]vminvq[_s16](int16_t a, int16x8_t b)	a -> Rda b -> Qm	VMINV.S16 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vminvq[_s32](int32_t a, int32x4_t b)	a -> Rda b -> Qm	VMINV.S32 Rda,Qm	Rda -> result	MVE
uint8_t [__arm_]vminvq[_u8](uint8_t a, uint8x16_t b)	a -> Rda b -> Qm	VMINV.U8 Rda,Qm	Rda -> result	MVE
uint16_t [__arm_]vminvq[_u16](uint16_t a, uint16x8_t b)	a -> Rda b -> Qm	VMINV.U16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vminvq[_u32](uint32_t a, uint32x4_t b)	a -> Rda b -> Qm	VMINV.U32 Rda,Qm	Rda -> result	MVE
int8_t [__arm_]vminvq_p[_s8](int8_t a, int8x16_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINVT.S8 Rda,Qm	Rda -> result	MVE
int16_t [__arm_]vminvq_p[_s16](int16_t a, int16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINVT.S16 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vminvq_p[_s32](int32_t a, int32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINVT.S32 Rda,Qm	Rda -> result	MVE
uint8_t [__arm_]vminvq_p[_u8](uint8_t a, uint8x16_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINVT.U8 Rda,Qm	Rda -> result	MVE
uint16_t [__arm_]vminvq_p[_u16](uint16_t a, uint16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINVT.U16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vminvq_p[_u32](uint32_t a, uint32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINVT.U32 Rda,Qm	Rda -> result	MVE
uint8_t [__arm_]vminavq[_s8](uint8_t a, int8x16_t b)	a -> Rda b -> Qm	VMINAV.S8 Rda,Qm	Rda -> result	MVE
uint16_t [__arm_]vminavq[_s16](uint16_t a, int16x8_t b)	a -> Rda b -> Qm	VMINAV.S16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vminavq[_s32](uint32_t a, int32x4_t b)	a -> Rda b -> Qm	VMINAV.S32 Rda,Qm	Rda -> result	MVE
uint8_t [__arm_]vminavq_p[_s8](uint8_t a, int8x16_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINAVT.S8 Rda,Qm	Rda -> result	MVE
uint16_t [__arm_]vminavq_p[_s16](uint16_t a, int16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINAVT.S16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vminavq_p[_s32](uint32_t a, int32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINAVT.S32 Rda,Qm	Rda -> result	MVE
float16x8_t [__arm_]vminnmq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VMINNM.F16 Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vminnmq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VMINNM.F32 Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vminnmq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vminnmq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vminnmq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vminnmq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vminnmaq[_f16](float16x8_t a, float16x8_t b)	a -> Qda b -> Qm	VMINNMA.F16 Qda,Qm	Qda -> result	MVE
float32x4_t [__arm_]vminnmaq[_f32](float32x4_t a, float32x4_t b)	a -> Qda b -> Qm	VMINNMA.F32 Qda,Qm	Qda -> result	MVE
float16x8_t [__arm_]vminnmaq_m[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMAT.F16 Qda,Qm	Qda -> result	MVE
float32x4_t [__arm_]vminnmaq_m[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMAT.F32 Qda,Qm	Qda -> result	MVE
float16_t [__arm_]vminnmvq[_f16](float16_t a, float16x8_t b)	a -> Rda b -> Qm	VMINNMV.F16 Rda,Qm	Rda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float32_t [__arm_]vminnmvq[_f32](float32_t a, float32x4_t b)	a -> Rda b -> Qm	VMINNMV.F32 Rda,Qm	Rda -> result	MVE
float16_t [__arm_]vminnmvq_p[_f16](float16_t a, float16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMVT.F16 Rda,Qm	Rda -> result	MVE
float32_t [__arm_]vminnmvq_p[_f32](float32_t a, float32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMVT.F32 Rda,Qm	Rda -> result	MVE
float16_t [__arm_]vminnmavq[_f16](float16_t a, float16x8_t b)	a -> Rda b -> Qm	VMINNMAV.F16 Rda,Qm	Rda -> result	MVE
float32_t [__arm_]vminnmavq[_f32](float32_t a, float32x4_t b)	a -> Rda b -> Qm	VMINNMAV.F32 Rda,Qm	Rda -> result	MVE
float16_t [__arm_]vminnmavq_p[_f16](float16_t a, float16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMAVT.F16 Rda,Qm	Rda -> result	MVE
float32_t [__arm_]vminnmavq_p[_f32](float32_t a, float32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMINNMAVT.F32 Rda,Qm	Rda -> result	MVE
int8x16_t [__arm_]vmaxq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VMAX.S8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vmaxq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMAX.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vmaxq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMAX.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vmaxq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VMAX.U8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vmaxq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VMAX.U16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vmaxq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VMAX.U32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vmaxq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmaxq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmaxq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmaxq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmaxq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmaxq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmaxq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmaxq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmaxq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmaxq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmaxq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmaxq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXT.U32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmaxaq[_s8](uint8x16_t a, int8x16_t b)	a -> Qda b -> Qm	VMAXA.S8 Qda,Qm	Qda -> result	MVE
uint16x8_t [__arm_]vmaxaq[_s16](uint16x8_t a, int16x8_t b)	a -> Qda b -> Qm	VMAXA.S16 Qda,Qm	Qda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vmaxaq[_s32](uint32x4_t a, int32x4_t b)	a -> Qda b -> Qm	VMAXA.S32 Qda,Qm	Qda -> result	MVE
uint8x16_t [__arm_]vmaxaq_m[_s8](uint8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXAT.S8 Qda,Qm	Qda -> result	MVE
uint16x8_t [__arm_]vmaxaq_m[_s16](uint16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXAT.S16 Qda,Qm	Qda -> result	MVE
uint32x4_t [__arm_]vmaxaq_m[_s32](uint32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXAT.S32 Qda,Qm	Qda -> result	MVE
int8_t [__arm_]vmaxvq[_s8](int8_t a, int8x16_t b)	a -> Rda b -> Qm	VMAXV.S8 Rda,Qm	Rda -> result	MVE
int16_t [__arm_]vmaxvq[_s16](int16_t a, int16x8_t b)	a -> Rda b -> Qm	VMAXV.S16 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vmaxvq[_s32](int32_t a, int32x4_t b)	a -> Rda b -> Qm	VMAXV.S32 Rda,Qm	Rda -> result	MVE
uint8_t [__arm_]vmaxvq[_u8](uint8_t a, uint8x16_t b)	a -> Rda b -> Qm	VMAXV.U8 Rda,Qm	Rda -> result	MVE
uint16_t [__arm_]vmaxvq[_u16](uint16_t a, uint16x8_t b)	a -> Rda b -> Qm	VMAXV.U16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vmaxvq[_u32](uint32_t a, uint32x4_t b)	a -> Rda b -> Qm	VMAXV.U32 Rda,Qm	Rda -> result	MVE
int8_t [__arm_]vmaxvq_p[_s8](int8_t a, int8x16_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXVT.S8 Rda,Qm	Rda -> result	MVE
int16_t [__arm_]vmaxvq_p[_s16](int16_t a, int16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXVT.S16 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vmaxvq_p[_s32](int32_t a, int32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXVT.S32 Rda,Qm	Rda -> result	MVE
uint8_t [__arm_]vmaxvq_p[_u8](uint8_t a, uint8x16_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXVT.U8 Rda,Qm	Rda -> result	MVE
uint16_t [__arm_]vmaxvq_p[_u16](uint16_t a, uint16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXVT.U16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vmaxvq_p[_u32](uint32_t a, uint32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXVT.U32 Rda,Qm	Rda -> result	MVE
uint8_t [__arm_]vmaxavq[_s8](uint8_t a, int8x16_t b)	a -> Rda b -> Qm	VMAXAV.S8 Rda,Qm	Rda -> result	MVE
uint16_t [__arm_]vmaxavq[_s16](uint16_t a, int16x8_t b)	a -> Rda b -> Qm	VMAXAV.S16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vmaxavq[_s32](uint32_t a, int32x4_t b)	a -> Rda b -> Qm	VMAXAV.S32 Rda,Qm	Rda -> result	MVE
uint8_t [__arm_]vmaxavq_p[_s8](uint8_t a, int8x16_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXAVT.S8 Rda,Qm	Rda -> result	MVE
uint16_t [__arm_]vmaxavq_p[_s16](uint16_t a, int16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXAVT.S16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vmaxavq_p[_s32](uint32_t a, int32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXAVT.S32 Rda,Qm	Rda -> result	MVE
float16x8_t [__arm_]vmaxnmq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VMAXNM.F16 Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vmaxnmq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VMAXNM.F32 Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vmaxnmq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vmaxnmq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vmaxnmq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vmaxnmq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vmaxnmaq[_f16](float16x8_t a, float16x8_t b)	a -> Qda b -> Qm	VMAXNMA.F16 Qda,Qm	Qda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float32x4_t [__arm_]vmaxnmaq[_f32](float32x4_t a, float32x4_t b)	a -> Qda b -> Qm	VMAXNMA.F32 Qda,Qm	Qda -> result	MVE
float16x8_t [__arm_]vmaxnmaq_m[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMAT.F16 Qda,Qm	Qda -> result	MVE
float32x4_t [__arm_]vmaxnmaq_m[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMAT.F32 Qda,Qm	Qda -> result	MVE
float16_t [__arm_]vmaxnmvq[_f16](float16_t a, float16x8_t b)	a -> Rda b -> Qm	VMAXNMV.F16 Rda,Qm	Rda -> result	MVE
float32_t [__arm_]vmaxnmvq[_f32](float32_t a, float32x4_t b)	a -> Rda b -> Qm	VMAXNMV.F32 Rda,Qm	Rda -> result	MVE
float16_t [__arm_]vmaxnmvq_p[_f16](float16_t a, float16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMVT.F16 Rda,Qm	Rda -> result	MVE
float32_t [__arm_]vmaxnmvq_p[_f32](float32_t a, float32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMVT.F32 Rda,Qm	Rda -> result	MVE
float16_t [__arm_]vmaxnmavq[_f16](float16_t a, float16x8_t b)	a -> Rda b -> Qm	VMAXNMAV.F16 Rda,Qm	Rda -> result	MVE
float32_t [__arm_]vmaxnmavq[_f32](float32_t a, float32x4_t b)	a -> Rda b -> Qm	VMAXNMAV.F32 Rda,Qm	Rda -> result	MVE
float16_t [__arm_]vmaxnmavq_p[_f16](float16_t a, float16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMAVT.F16 Rda,Qm	Rda -> result	MVE
float32_t [__arm_]vmaxnmavq_p[_f32](float32_t a, float32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VMAXNMAVT.F32 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq[_s8](uint32_t a, int8x16_t b, int8x16_t c)	a -> Rda b -> Qn c -> Qm	VABAV.S8 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq[_s16](uint32_t a, int16x8_t b, int16x8_t c)	a -> Rda b -> Qn c -> Qm	VABAV.S16 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq[_s32](uint32_t a, int32x4_t b, int32x4_t c)	a -> Rda b -> Qn c -> Qm	VABAV.S32 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq[_u8](uint32_t a, uint8x16_t b, uint8x16_t c)	a -> Rda b -> Qn c -> Qm	VABAV.U8 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq[_u16](uint32_t a, uint16x8_t b, uint16x8_t c)	a -> Rda b -> Qn c -> Qm	VABAV.U16 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq[_u32](uint32_t a, uint32x4_t b, uint32x4_t c)	a -> Rda b -> Qn c -> Qm	VABAV.U32 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq_p[_s8](uint32_t a, int8x16_t b, int8x16_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VABAVT.S8 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq_p[_s16](uint32_t a, int16x8_t b, int16x8_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VABAVT.S16 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq_p[_s32](uint32_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VABAVT.S32 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq_p[_u8](uint32_t a, uint8x16_t b, uint8x16_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VABAVT.U8 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq_p[_u16](uint32_t a, uint16x8_t b, uint16x8_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VABAVT.U16 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vabavq_p[_u32](uint32_t a, uint32x4_t b, uint32x4_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VABAVT.U32 Rda,Qn,Qm	Rda -> result	MVE
int8x16_t [__arm_]vabdq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VABD.S8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vabdq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VABD.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vabdq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VABD.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint8x16_t [__arm_]vabdq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VABD.U8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vabdq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VABD.U16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vabdq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VABD.U32 Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vabdq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VABD.F16 Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vabdq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VABD.F32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vabdq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vabdq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vabdq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vabdq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vabdq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vabdq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.U32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vabdq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vabdq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.F32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vabdq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vabdq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vabdq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vabdq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vabdq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vabdq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.U32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vabdq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vabdq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VABDT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vabsq[_f16](float16x8_t a)	a -> Qm	VABS.F16 Qd,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vabsq[_f32](float32x4_t a)	a -> Qm	VABS.F32 Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vabsq[_s8](int8x16_t a)	a -> Qm	VABS.S8 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vabsq[_s16](int16x8_t a)	a -> Qm	VABS.S16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vabsq[_s32](int32x4_t a)	a -> Qm	VABS.S32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vabsq_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vabsq_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.F32 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vabsq_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.S8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vabsq_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.S16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vabsq_m[_s32](int32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.S32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vabsq_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vabsq_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.F32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vabsq_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.S8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vabsq_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.S16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vabsq_x[_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VABST.S32 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vadciq[_s32](int32x4_t a, int32x4_t b, unsigned * carry_out)	a -> Qn b -> Qm	VADCI.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry_out	MVE
uint32x4_t [__arm_]vadciq[_u32](uint32x4_t a, uint32x4_t b, unsigned * carry_out)	a -> Qn b -> Qm	VADCI.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry_out	MVE
int32x4_t [__arm_]vadciq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, unsigned * carry_out, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADCIT.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry_out	MVE
uint32x4_t [__arm_]vadciq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, unsigned * carry_out, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADCIT.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry_out	MVE
int32x4_t [__arm_]vadcq[_s32](int32x4_t a, int32x4_t b, unsigned * carry)	a -> Qn b -> Qm *carry -> Rt	VMRS Rs,FPSCR_nzcvqc BFI Rs,Rt,#29,#1 VMSR FPSCR_nzcvqc,Rs VADC.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry	MVE
uint32x4_t [__arm_]vadcq[_u32](uint32x4_t a, uint32x4_t b, unsigned * carry)	a -> Qn b -> Qm *carry -> Rt	VMRS Rs,FPSCR_nzcvqc BFI Rs,Rt,#29,#1 VMSR FPSCR_nzcvqc,Rs VADC.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry	MVE
int32x4_t [__arm_]vadcq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, unsigned * carry, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm *carry -> Rt p -> Rp	VMRS Rs,FPSCR_nzcvqc BFI Rs,Rt,#29,#1 VMSR FPSCR_nzcvqc,Rs VMSR P0,Rp VPST VADCT.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry	MVE
uint32x4_t [__arm_]vadcq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, unsigned * carry, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm *carry -> Rt p -> Rp	VMRS Rs,FPSCR_nzcvqc BFI Rs,Rt,#29,#1 VMSR FPSCR_nzcvqc,Rs VMSR P0,Rp VPST VADCT.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vaddq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VADD.F16 Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vaddq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VADD.F32 Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vaddq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VADD.F16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vaddq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VADD.F32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vaddq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VADD.I8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vaddq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VADD.I16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vaddq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VADD.I32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vaddq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VADD.I8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vaddq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VADD.I16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vaddq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VADD.I32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vaddq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VADD.I8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vaddq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VADD.I16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vaddq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VADD.I32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vaddq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VADD.I8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vaddq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VADD.I16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vaddq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VADD.I32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vaddq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vaddq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vaddq_m[_n_f16](float16x8_t inactive, float16x8_t a, float16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.F16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vaddq_m[_n_f32](float32x4_t inactive, float32x4_t a, float32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.F32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vaddq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vaddq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vaddq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vaddq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vaddq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vaddq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vaddq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I8 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vaddq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vaddq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vaddq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, uint8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vaddq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, uint16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vaddq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, uint32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vaddq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vaddq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vaddq_x[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.F16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vaddq_x[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.F32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vaddq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vaddq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vaddq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vaddq_x[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vaddq_x[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vaddq_x[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vaddq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vaddq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vaddq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VADDT.I32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vaddq_x[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vaddq_x[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vaddq_x[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VADDT.I32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vclsq[_s8](int8x16_t a)	a -> Qm	VCLS.S8 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vclsq[_s16](int16x8_t a)	a -> Qm	VCLS.S16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vclsq[_s32](int32x4_t a)	a -> Qm	VCLS.S32 Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vclsq_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLST.S8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vclsq_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLST.S16 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vclsq_m[_s32](int32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLST.S32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vclsq_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLST.S8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vclsq_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLST.S16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vclsq_x[_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLST.S32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vclzq[_s8](int8x16_t a)	a -> Qm	VCLZ.I8 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vclzq[_s16](int16x8_t a)	a -> Qm	VCLZ.I16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vclzq[_s32](int32x4_t a)	a -> Qm	VCLZ.I32 Qd,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vclzq[_u8](uint8x16_t a)	a -> Qm	VCLZ.I8 Qd,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vclzq[_u16](uint16x8_t a)	a -> Qm	VCLZ.I16 Qd,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vclzq[_u32](uint32x4_t a)	a -> Qm	VCLZ.I32 Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vclzq_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vclzq_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vclzq_m[_s32](int32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vclzq_m[_u8](uint8x16_t inactive, uint8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I8 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vclzq_m[_u16](uint16x8_t inactive, uint16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vclzq_m[_u32](uint32x4_t inactive, uint32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vclzq_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vclzq_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vclzq_x[_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vclzq_x[_u8](uint8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I8 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vclzq_x[_u16](uint16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vclzq_x[_u32](uint32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCLZT.I32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vnegq[_f16](float16x8_t a)	a -> Qm	VNEG.F16 Qd,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vnegq[_f32](float32x4_t a)	a -> Qm	VNEG.F32 Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vnegq[_s8](int8x16_t a)	a -> Qm	VNEG.S8 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vnegq[_s16](int16x8_t a)	a -> Qm	VNEG.S16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vnegq[_s32](int32x4_t a)	a -> Qm	VNEG.S32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vnegq_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vnegq_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.F32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vnegq_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.S8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vnegq_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.S16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vnegq_m[_s32](int32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.S32 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vnegq_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vnegq_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.F32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vnegq_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.S8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vnegq_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.S16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vnegq_x[_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VNEGT.S32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmulhq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VMULH.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmulhq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMULH.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmulhq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMULH.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmulhq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VMULH.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulhq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VMULH.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulhq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VMULH.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmulhq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmulhq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmulhq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmulhq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulhq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulhq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmulhq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmulhq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmulhq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmulhq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulhq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulhq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULHT.U32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmullbq_poly[_p8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VMULLB.P8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmullbq_poly[_p16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VMULLB.P16 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmullbq_int[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VMULLB.S8 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmullbq_int[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMULLB.S16 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int64x2_t [__arm_]vmullbq_int[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMULLB.S32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmullbq_int[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VMULLB.U8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmullbq_int[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VMULLB.U16 Qd,Qn,Qm	Qd -> result	MVE
uint64x2_t [__arm_]vmullbq_int[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VMULLB.U32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmullbq_poly_m[_p8](uint16x8_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.P8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmullbq_poly_m[_p16](uint32x4_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.P16 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmullbq_int_m[_s8](int16x8_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.S8 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmullbq_int_m[_s16](int32x4_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vmullbq_int_m[_s32](int64x2_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmullbq_int_m[_u8](uint16x8_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmullbq_int_m[_u16](uint32x4_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint64x2_t [__arm_]vmullbq_int_m[_u32](uint64x2_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.U32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmullbq_poly_x[_p8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.P8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmullbq_poly_x[_p16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.P16 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmullbq_int_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.S8 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmullbq_int_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vmullbq_int_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmullbq_int_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmullbq_int_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint64x2_t [__arm_]vmullbq_int_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLBT.U32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulltq_poly[_p8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VMULLT.P8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulltq_poly[_p16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VMULLT.P16 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmulltq_int[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VMULLT.S8 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmulltq_int[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMULLT.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vmulltq_int[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMULLT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulltq_int[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VMULLT.U8 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vmulltq_int[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VMULLT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint64x2_t [__arm_]vmulltq_int[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VMULLT.U32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulltq_poly_m[_p8](uint16x8_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.P8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulltq_poly_m[_p16](uint32x4_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.P16 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmulltq_int_m[_s8](int16x8_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.S8 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmulltq_int_m[_s16](int32x4_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vmulltq_int_m[_s32](int64x2_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulltq_int_m[_u8](uint16x8_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulltq_int_m[_u16](uint32x4_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint64x2_t [__arm_]vmulltq_int_m[_u32](uint64x2_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.U32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulltq_poly_x[_p8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.P8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulltq_poly_x[_p16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.P16 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmulltq_int_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.S8 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmulltq_int_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vmulltq_int_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulltq_int_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulltq_int_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint64x2_t [__arm_]vmulltq_int_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULLTT.U32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vmulq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VMUL.F16 Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vmulq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VMUL.F32 Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vmulq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VMUL.F16 Qd,Qn,Rm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vmulq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VMUL.F32 Qd,Qn,Rm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vmulq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VMUL.I8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vmulq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMUL.I16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vmulq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMUL.I32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vmulq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VMUL.I8 Qd,Qn,Rm	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int16x8_t [__arm_]vmulq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VMUL.I16 Qd,Qn,Rm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vmulq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VMUL.I32 Qd,Qn,Rm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vmulq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VMUL.I8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vmulq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VMUL.I16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vmulq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VMUL.I32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vmulq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VMUL.I8 Qd,Qn,Rm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vmulq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VMUL.I16 Qd,Qn,Rm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vmulq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VMUL.I32 Qd,Qn,Rm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vmulq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vmulq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vmulq_m[_n_f16](float16x8_t inactive, float16x8_t a, float16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.F16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vmulq_m[_n_f32](float32x4_t inactive, float32x4_t a, float32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.F32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vmulq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmulq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmulq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmulq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vmulq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vmulq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vmulq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmulq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, uint8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vmulq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, uint16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I16 Qd,Qn,Rm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vmulq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, uint32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vmulq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vmulq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vmulq_x[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.F16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vmulq_x[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.F32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vmulq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmulq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmulq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmulq_x[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vmulq_x[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vmulq_x[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vmulq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmulq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmulq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMULT.I32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmulq_x[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vmulq_x[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vmulq_x[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VMULT.I32 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vsbciq[_s32](int32x4_t a, int32x4_t b, unsigned * carry_out)	a -> Qn b -> Qm	VSBCI.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry_out	MVE
uint32x4_t [__arm_]vsbciq[_u32](uint32x4_t a, uint32x4_t b, unsigned * carry_out)	a -> Qn b -> Qm	VSBCI.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry_out	MVE
int32x4_t [__arm_]vsbciq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, unsigned * carry_out, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSBCIT.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry_out	MVE
uint32x4_t [__arm_]vsbciq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, unsigned * carry_out, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSBCIT.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry_out	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vsbcq[_s32](int32x4_t a, int32x4_t b, unsigned * carry)	a -> Qn b -> Qm *carry -> Rt	VMRS Rs,FPSCR_nzcvqc BFI Rs,Rt,#29,#1 VMSR FPSCR_nzcvqc,Rs VSBC.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry	MVE
uint32x4_t [__arm_]vsbcq[_u32](uint32x4_t a, uint32x4_t b, unsigned * carry)	a -> Qn b -> Qm *carry -> Rt	VMRS Rs,FPSCR_nzcvqc BFI Rs,Rt,#29,#1 VMSR FPSCR_nzcvqc,Rs VSBC.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry	MVE
int32x4_t [__arm_]vsbcq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, unsigned * carry, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm *carry -> Rt p -> Rp	VMRS Rs,FPSCR_nzcvqc BFI Rs,Rt,#29,#1 VMSR FPSCR_nzcvqc,Rs VMSR P0,Rp VPST VSBCT.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry	MVE
uint32x4_t [__arm_]vsbcq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, unsigned * carry, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm *carry -> Rt p -> Rp	VMRS Rs,FPSCR_nzcvqc BFI Rs,Rt,#29,#1 VMSR FPSCR_nzcvqc,Rs VMSR P0,Rp VPST VSBCT.I32 Qd,Qn,Qm VMRS Rt,FPSCR_nzcvqc LSR Rt,#29 AND Rt,#1	Qd -> result Rt -> *carry	MVE
int8x16_t [__arm_]vsubq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VSUB.I8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vsubq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VSUB.I16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vsubq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VSUB.I32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vsubq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VSUB.I8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vsubq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VSUB.I16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vsubq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VSUB.I32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vsubq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VSUB.I8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vsubq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VSUB.I16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vsubq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VSUB.I32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vsubq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VSUB.I8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vsubq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VSUB.I16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vsubq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VSUB.I32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vsubq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VSUB.F16 Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vsubq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VSUB.F32 Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vsubq[_n_f16](float16x8_t a, float16_t b)	a -> Qn b -> Rm	VSUB.F16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vsubq[_n_f32](float32x4_t a, float32_t b)	a -> Qn b -> Rm	VSUB.F32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vsubq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vsubq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vsubq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I32 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vsubq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vsubq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vsubq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vsubq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vsubq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vsubq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vsubq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, uint8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vsubq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, uint16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vsubq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, uint32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vsubq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vsubq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vsubq_m[_n_f16](float16x8_t inactive, float16x8_t a, float16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.F16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vsubq_m[_n_f32](float32x4_t inactive, float32x4_t a, float32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.F32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vsubq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vsubq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vsubq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vsubq_x[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vsubq_x[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vsubq_x[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vsubq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vsubq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I16 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vsubq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.I32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vsubq_x[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vsubq_x[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vsubq_x[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.I32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vsubq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.F16 Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vsubq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VSUBT.F32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vsubq_x[_n_f16](float16x8_t a, float16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.F16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vsubq_x[_n_f32](float32x4_t a, float32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VSUBT.F32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vcaddq_rot90[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCADD.F16 Qd,Qn,Qm,#90	Qd -> result	MVE/NEON
float32x4_t [__arm_]vcaddq_rot90[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCADD.F32 Qd,Qn,Qm,#90	Qd -> result	MVE/NEON
int8x16_t [__arm_]vcaddq_rot90[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VCADD.I8 Qd,Qn,Qm,#90	Qd -> result	MVE
int16x8_t [__arm_]vcaddq_rot90[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VCADD.I16 Qd,Qn,Qm,#90	Qd -> result	MVE
int32x4_t [__arm_]vcaddq_rot90[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VCADD.I32 Qd,Qn,Qm,#90	Qd -> result	MVE
uint8x16_t [__arm_]vcaddq_rot90[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VCADD.I8 Qd,Qn,Qm,#90	Qd -> result	MVE
uint16x8_t [__arm_]vcaddq_rot90[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VCADD.I16 Qd,Qn,Qm,#90	Qd -> result	MVE
uint32x4_t [__arm_]vcaddq_rot90[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VCADD.I32 Qd,Qn,Qm,#90	Qd -> result	MVE
float16x8_t [__arm_]vcaddq_rot270[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCADD.F16 Qd,Qn,Qm,#270	Qd -> result	MVE/NEON
float32x4_t [__arm_]vcaddq_rot270[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCADD.F32 Qd,Qn,Qm,#270	Qd -> result	MVE/NEON
int8x16_t [__arm_]vcaddq_rot270[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VCADD.I8 Qd,Qn,Qm,#270	Qd -> result	MVE
int16x8_t [__arm_]vcaddq_rot270[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VCADD.I16 Qd,Qn,Qm,#270	Qd -> result	MVE
int32x4_t [__arm_]vcaddq_rot270[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VCADD.I32 Qd,Qn,Qm,#270	Qd -> result	MVE
uint8x16_t [__arm_]vcaddq_rot270[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VCADD.I8 Qd,Qn,Qm,#270	Qd -> result	MVE
uint16x8_t [__arm_]vcaddq_rot270[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VCADD.I16 Qd,Qn,Qm,#270	Qd -> result	MVE
uint32x4_t [__arm_]vcaddq_rot270[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VCADD.I32 Qd,Qn,Qm,#270	Qd -> result	MVE
float16x8_t [__arm_]vcaddq_rot90_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.F16 Qd,Qn,Qm,#90	Qd -> result	MVE
float32x4_t [__arm_]vcaddq_rot90_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.F32 Qd,Qn,Qm,#90	Qd -> result	MVE
int8x16_t [__arm_]vcaddq_rot90_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I8 Qd,Qn,Qm,#90	Qd -> result	MVE
int16x8_t [__arm_]vcaddq_rot90_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I16 Qd,Qn,Qm,#90	Qd -> result	MVE
int32x4_t [__arm_]vcaddq_rot90_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I32 Qd,Qn,Qm,#90	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint8x16_t [__arm_]vcaddq_rot90_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I8 Qd,Qn,Qm,#90	Qd -> result	MVE
uint16x8_t [__arm_]vcaddq_rot90_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I16 Qd,Qn,Qm,#90	Qd -> result	MVE
uint32x4_t [__arm_]vcaddq_rot90_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I32 Qd,Qn,Qm,#90	Qd -> result	MVE
float16x8_t [__arm_]vcaddq_rot270_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.F16 Qd,Qn,Qm,#270	Qd -> result	MVE
float32x4_t [__arm_]vcaddq_rot270_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.F32 Qd,Qn,Qm,#270	Qd -> result	MVE
int8x16_t [__arm_]vcaddq_rot270_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I8 Qd,Qn,Qm,#270	Qd -> result	MVE
int16x8_t [__arm_]vcaddq_rot270_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I16 Qd,Qn,Qm,#270	Qd -> result	MVE
int32x4_t [__arm_]vcaddq_rot270_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I32 Qd,Qn,Qm,#270	Qd -> result	MVE
uint8x16_t [__arm_]vcaddq_rot270_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I8 Qd,Qn,Qm,#270	Qd -> result	MVE
uint16x8_t [__arm_]vcaddq_rot270_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I16 Qd,Qn,Qm,#270	Qd -> result	MVE
uint32x4_t [__arm_]vcaddq_rot270_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I32 Qd,Qn,Qm,#270	Qd -> result	MVE
float16x8_t [__arm_]vcaddq_rot90_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.F16 Qd,Qn,Qm,#90	Qd -> result	MVE
float32x4_t [__arm_]vcaddq_rot90_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.F32 Qd,Qn,Qm,#90	Qd -> result	MVE
int8x16_t [__arm_]vcaddq_rot90_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I8 Qd,Qn,Qm,#90	Qd -> result	MVE
int16x8_t [__arm_]vcaddq_rot90_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I16 Qd,Qn,Qm,#90	Qd -> result	MVE
int32x4_t [__arm_]vcaddq_rot90_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I32 Qd,Qn,Qm,#90	Qd -> result	MVE
uint8x16_t [__arm_]vcaddq_rot90_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I8 Qd,Qn,Qm,#90	Qd -> result	MVE
uint16x8_t [__arm_]vcaddq_rot90_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I16 Qd,Qn,Qm,#90	Qd -> result	MVE
uint32x4_t [__arm_]vcaddq_rot90_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I32 Qd,Qn,Qm,#90	Qd -> result	MVE
float16x8_t [__arm_]vcaddq_rot270_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.F16 Qd,Qn,Qm,#270	Qd -> result	MVE
float32x4_t [__arm_]vcaddq_rot270_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.F32 Qd,Qn,Qm,#270	Qd -> result	MVE
int8x16_t [__arm_]vcaddq_rot270_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I8 Qd,Qn,Qm,#270	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int16x8_t [__arm_]vcaddq_rot270_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I16 Qd,Qn,Qm,#270	Qd -> result	MVE
int32x4_t [__arm_]vcaddq_rot270_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I32 Qd,Qn,Qm,#270	Qd -> result	MVE
uint8x16_t [__arm_]vcaddq_rot270_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I8 Qd,Qn,Qm,#270	Qd -> result	MVE
uint16x8_t [__arm_]vcaddq_rot270_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I16 Qd,Qn,Qm,#270	Qd -> result	MVE
uint32x4_t [__arm_]vcaddq_rot270_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCADDT.I32 Qd,Qn,Qm,#270	Qd -> result	MVE
float16x8_t [__arm_]vcmlaq[_f16](float16x8_t a, float16x8_t b, float16x8_t c)	a -> Qda b -> Qn c -> Qm	VCMLA.F16 Qda,Qn,Qm,#0	Qda -> result	MVE/NEON
float32x4_t [__arm_]vcmlaq[_f32](float32x4_t a, float32x4_t b, float32x4_t c)	a -> Qda b -> Qn c -> Qm	VCMLA.F32 Qda,Qn,Qm,#0	Qda -> result	MVE/NEON
float16x8_t [__arm_]vcmlaq_rot90[_f16](float16x8_t a, float16x8_t b, float16x8_t c)	a -> Qda b -> Qn c -> Qm	VCMLA.F16 Qda,Qn,Qm,#90	Qda -> result	MVE/NEON
float32x4_t [__arm_]vcmlaq_rot90[_f32](float32x4_t a, float32x4_t b, float32x4_t c)	a -> Qda b -> Qn c -> Qm	VCMLA.F32 Qda,Qn,Qm,#90	Qda -> result	MVE/NEON
float16x8_t [__arm_]vcmlaq_rot180[_f16](float16x8_t a, float16x8_t b, float16x8_t c)	a -> Qda b -> Qn c -> Qm	VCMLA.F16 Qda,Qn,Qm,#180	Qda -> result	MVE/NEON
float32x4_t [__arm_]vcmlaq_rot180[_f32](float32x4_t a, float32x4_t b, float32x4_t c)	a -> Qda b -> Qn c -> Qm	VCMLA.F32 Qda,Qn,Qm,#180	Qda -> result	MVE/NEON
float16x8_t [__arm_]vcmlaq_rot270[_f16](float16x8_t a, float16x8_t b, float16x8_t c)	a -> Qda b -> Qn c -> Qm	VCMLA.F16 Qda,Qn,Qm,#270	Qda -> result	MVE/NEON
float32x4_t [__arm_]vcmlaq_rot270[_f32](float32x4_t a, float32x4_t b, float32x4_t c)	a -> Qda b -> Qn c -> Qm	VCMLA.F32 Qda,Qn,Qm,#270	Qda -> result	MVE/NEON
float16x8_t [__arm_]vcmlaq_m[_f16](float16x8_t a, float16x8_t b, float16x8_t c, mve_pred16_t p)	a -> Qda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VCMLAT.F16 Qda,Qn,Qm,#0	Qda -> result	MVE
float32x4_t [__arm_]vcmlaq_m[_f32](float32x4_t a, float32x4_t b, float32x4_t c, mve_pred16_t p)	a -> Qda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VCMLAT.F32 Qda,Qn,Qm,#0	Qda -> result	MVE
float16x8_t [__arm_]vcmlaq_rot90_m[_f16](float16x8_t a, float16x8_t b, float16x8_t c, mve_pred16_t p)	a -> Qda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VCMLAT.F16 Qda,Qn,Qm,#90	Qda -> result	MVE
float32x4_t [__arm_]vcmlaq_rot90_m[_f32](float32x4_t a, float32x4_t b, float32x4_t c, mve_pred16_t p)	a -> Qda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VCMLAT.F32 Qda,Qn,Qm,#90	Qda -> result	MVE
float16x8_t [__arm_]vcmlaq_rot180_m[_f16](float16x8_t a, float16x8_t b, float16x8_t c, mve_pred16_t p)	a -> Qda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VCMLAT.F16 Qda,Qn,Qm,#180	Qda -> result	MVE
float32x4_t [__arm_]vcmlaq_rot180_m[_f32](float32x4_t a, float32x4_t b, float32x4_t c, mve_pred16_t p)	a -> Qda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VCMLAT.F32 Qda,Qn,Qm,#180	Qda -> result	MVE
float16x8_t [__arm_]vcmlaq_rot270_m[_f16](float16x8_t a, float16x8_t b, float16x8_t c, mve_pred16_t p)	a -> Qda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VCMLAT.F16 Qda,Qn,Qm,#270	Qda -> result	MVE
float32x4_t [__arm_]vcmlaq_rot270_m[_f32](float32x4_t a, float32x4_t b, float32x4_t c, mve_pred16_t p)	a -> Qda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VCMLAT.F32 Qda,Qn,Qm,#270	Qda -> result	MVE
float16x8_t [__arm_]vcmulq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMUL.F16 Qd,Qn,Qm,#0	Qd -> result	MVE
float32x4_t [__arm_]vcmulq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMUL.F32 Qd,Qn,Qm,#0	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot90[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMUL.F16 Qd,Qn,Qm,#90	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float32x4_t [__arm_]vcmulq_rot90[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMUL.F32 Qd,Qn,Qm,#90	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot180[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMUL.F16 Qd,Qn,Qm,#180	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_rot180[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMUL.F32 Qd,Qn,Qm,#180	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot270[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VCMUL.F16 Qd,Qn,Qm,#270	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_rot270[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VCMUL.F32 Qd,Qn,Qm,#270	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F16 Qd,Qn,Qm,#0	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F32 Qd,Qn,Qm,#0	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot90_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F16 Qd,Qn,Qm,#90	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_rot90_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F32 Qd,Qn,Qm,#90	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot180_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F16 Qd,Qn,Qm,#180	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_rot180_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F32 Qd,Qn,Qm,#180	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot270_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F16 Qd,Qn,Qm,#270	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_rot270_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F32 Qd,Qn,Qm,#270	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F16 Qd,Qn,Qm,#0	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F32 Qd,Qn,Qm,#0	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot90_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F16 Qd,Qn,Qm,#90	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_rot90_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F32 Qd,Qn,Qm,#90	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot180_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F16 Qd,Qn,Qm,#180	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_rot180_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F32 Qd,Qn,Qm,#180	Qd -> result	MVE
float16x8_t [__arm_]vcmulq_rot270_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F16 Qd,Qn,Qm,#270	Qd -> result	MVE
float32x4_t [__arm_]vcmulq_rot270_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VCMULT.F32 Qd,Qn,Qm,#270	Qd -> result	MVE
int8x16_t [__arm_]vqabsq[_s8](int8x16_t a)	a -> Qm	VQABS.S8 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vqabsq[_s16](int16x8_t a)	a -> Qm	VQABS.S16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqabsq[_s32](int32x4_t a)	a -> Qm	VQABS.S32 Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqabsq_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VQABST.S8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqabsq_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VQABST.S16 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vqabsq_m[_s32](int32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VQABST.S32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqaddq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VQADD.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vqaddq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VQADD.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqaddq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VQADD.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vqaddq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VQADD.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vqaddq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VQADD.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vqaddq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VQADD.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vqaddq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VQADD.S8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vqaddq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VQADD.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqaddq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VQADD.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vqaddq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VQADD.U8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vqaddq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VQADD.U16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vqaddq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VQADD.U32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqaddq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQADDT.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vqaddq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQADDT.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqaddq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQADDT.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vqaddq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, uint8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQADDT.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vqaddq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, uint16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQADDT.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vqaddq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, uint32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQADDT.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vqaddq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQADDT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqaddq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQADDT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqaddq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQADDT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqaddq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQADDT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqaddq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQADDT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vqaddq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQADDT.U32 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vqdmladhq[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLADH.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmladhq[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLADH.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmladhq[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLADH.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqdmladhq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLADHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmladhq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLADHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmladhq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLADHT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqdmladhxq[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLADHX.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmladhxq[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLADHX.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmladhxq[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLADHX.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqdmladhxq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLADHXT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmladhxq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLADHXT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmladhxq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLADHXT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmladhq[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLADH.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmladhq[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLADH.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqrdmladhq[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLADH.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmladhq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLADHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmladhq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLADHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqrdmladhq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLADHT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmladhxq[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLADHX.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmladhxq[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLADHX.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqrdmladhxq[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLADHX.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmladhxq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLADHXT.S8 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int16x8_t [__arm_]vqrdmladhxq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLADHXT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqrdmladhxq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLADHXT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqdmlahq[_n_s8](int8x16_t add, int8x16_t m1, int8_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VQDMLAH.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqdmlahq[_n_s16](int16x8_t add, int16x8_t m1, int16_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VQDMLAH.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqdmlahq[_n_s32](int32x4_t add, int32x4_t m1, int32_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VQDMLAH.S32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqdmlahq_m[_n_s8](int8x16_t add, int8x16_t m1, int8_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VQDMLAHT.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqdmlahq_m[_n_s16](int16x8_t add, int16x8_t m1, int16_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VQDMLAHT.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqdmlahq_m[_n_s32](int32x4_t add, int32x4_t m1, int32_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VQDMLAHT.S32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqrdmlahq[_n_s8](int8x16_t add, int8x16_t m1, int8_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VQRDMLAH.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqrdmlahq[_n_s16](int16x8_t add, int16x8_t m1, int16_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VQRDMLAH.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqrdmlahq[_n_s32](int32x4_t add, int32x4_t m1, int32_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VQRDMLAH.S32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqrdmlahq_m[_n_s8](int8x16_t add, int8x16_t m1, int8_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMLAHT.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqrdmlahq_m[_n_s16](int16x8_t add, int16x8_t m1, int16_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMLAHT.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqrdmlahq_m[_n_s32](int32x4_t add, int32x4_t m1, int32_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMLAHT.S32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqdmlashq[_n_s8](int8x16_t m1, int8x16_t m2, int8_t add)	m1 -> Qda m2 -> Qn add -> Rm	VQDMLASH.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqdmlashq[_n_s16](int16x8_t m1, int16x8_t m2, int16_t add)	m1 -> Qda m2 -> Qn add -> Rm	VQDMLASH.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqdmlashq[_n_s32](int32x4_t m1, int32x4_t m2, int32_t add)	m1 -> Qda m2 -> Qn add -> Rm	VQDMLASH.S32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqdmlashq_m[_n_s8](int8x16_t m1, int8x16_t m2, int8_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VQDMLASHT.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqdmlashq_m[_n_s16](int16x8_t m1, int16x8_t m2, int16_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VQDMLASHT.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqdmlashq_m[_n_s32](int32x4_t m1, int32x4_t m2, int32_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VQDMLASHT.S32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqrdmlashq[_n_s8](int8x16_t m1, int8x16_t m2, int8_t add)	m1 -> Qda m2 -> Qn add -> Rm	VQRDMLASH.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqrdmlashq[_n_s16](int16x8_t m1, int16x8_t m2, int16_t add)	m1 -> Qda m2 -> Qn add -> Rm	VQRDMLASH.S16 Qda,Qn,Rm	Qda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vqrdmlashq[_n_s32](int32x4_t m1, int32x4_t m2, int32_t add)	m1 -> Qda m2 -> Qn add -> Rm	VQRDMLASH.S32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqrdmlashq_m[_n_s8](int8x16_t m1, int8x16_t m2, int8_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMLASHT.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqrdmlashq_m[_n_s16](int16x8_t m1, int16x8_t m2, int16_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMLASHT.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqrdmlashq_m[_n_s32](int32x4_t m1, int32x4_t m2, int32_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMLASHT.S32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqdmlsdhq[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLSDH.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmlsdhq[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLSDH.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmlsdhq[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLSDH.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqdmlsdhq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLSDHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmlsdhq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLSDHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmlsdhq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLSDHT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqdmlsdhxq[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLSDHX.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmlsdhxq[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLSDHX.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmlsdhxq[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b)	inactive -> Qd a -> Qn b -> Qm	VQDMLSDHX.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqdmlsdhxq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLSDHXT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmlsdhxq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLSDHXT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmlsdhxq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMLSDHXT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmlsdhq[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLSDH.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmlsdhq[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLSDH.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqrdmlsdhq[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLSDH.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmlsdhq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLSDHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmlsdhq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLSDHT.S16 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vqrdmlsdhq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLSDHT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmlsdhxq[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLSDHX.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmlsdhxq[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLSDHX.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqrdmlsdhxq[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b)	inactive -> Qd a -> Qn b -> Qm	VQRDMLSDHX.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmlsdhxq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLSDHXT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmlsdhxq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLSDHXT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqrdmlsdhxq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMLSDHXT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqdmulhq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VQDMULH.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vqdmulhq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VQDMULH.S16 Qd,Qn,Rm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqdmulhq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VQDMULH.S32 Qd,Qn,Rm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqdmulhq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQDMULHT.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vqdmulhq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQDMULHT.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqdmulhq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQDMULHT.S32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vqdmulhq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VQDMULH.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmulhq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VQDMULH.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqdmulhq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VQDMULH.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqdmulhq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMULHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqdmulhq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMULHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmulhq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMULHT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmulhq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VQRDMULH.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmulhq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VQRDMULH.S16 Qd,Qn,Rm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqrdmulhq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VQRDMULH.S32 Qd,Qn,Rm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqrdmulhq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMULHT.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmulhq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMULHT.S16 Qd,Qn,Rm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vqrdmulhq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQRDMULHT.S32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vqrdmulhq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VQRDMULH.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmulhq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VQRDMULH.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqrdmulhq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VQRDMULH.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqrdmulhq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMULHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqrdmulhq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMULHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqrdmulhq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQRDMULHT.S32 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmullbq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VQDMULLB.S16 Qd,Qn,Rm	Qd -> result	MVE
int64x2_t [__arm_]vqdmullbq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VQDMULLB.S32 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqdmullbq_m[_n_s16](int32x4_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQDMULLBT.S16 Qd,Qn,Rm	Qd -> result	MVE
int64x2_t [__arm_]vqdmullbq_m[_n_s32](int64x2_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQDMULLBT.S32 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqdmullbq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VQDMULLB.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vqdmullbq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VQDMULLB.S32 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmullbq_m[_s16](int32x4_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMULLBT.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vqdmullbq_m[_s32](int64x2_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMULLBT.S32 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmulltq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VQDMULLT.S16 Qd,Qn,Rm	Qd -> result	MVE
int64x2_t [__arm_]vqdmulltq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VQDMULLT.S32 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqdmulltq_m[_n_s16](int32x4_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQDMULLTT.S16 Qd,Qn,Rm	Qd -> result	MVE
int64x2_t [__arm_]vqdmulltq_m[_n_s32](int64x2_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQDMULLTT.S32 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqdmulltq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VQDMULLT.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vqdmulltq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VQDMULLT.S32 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqdmulltq_m[_s16](int32x4_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMULLTT.S16 Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vqdmulltq_m[_s32](int64x2_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQDMULLTT.S32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqnegq[_s8](int8x16_t a)	a -> Qm	VQNEG.S8 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vqnegq[_s16](int16x8_t a)	a -> Qm	VQNEG.S16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqnegq[_s32](int32x4_t a)	a -> Qm	VQNEG.S32 Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqnegq_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VQNEGT.S8 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int16x8_t [__arm_]vqnegq_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VQNEGT.S16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqnegq_m[_s32](int32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VQNEGT.S32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqsubq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VQSUB.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vqsubq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VQSUB.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqsubq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VQSUB.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vqsubq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VQSUB.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vqsubq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VQSUB.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vqsubq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VQSUB.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vqsubq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQSUBT.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vqsubq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQSUBT.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vqsubq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQSUBT.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vqsubq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, uint8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQSUBT.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vqsubq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, uint16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQSUBT.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vqsubq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, uint32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VQSUBT.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vqsubq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VQSUB.S8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vqsubq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VQSUB.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqsubq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VQSUB.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vqsubq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VQSUB.U8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vqsubq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VQSUB.U16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vqsubq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VQSUB.U32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqsubq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQSUBT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqsubq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQSUBT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vqsubq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQSUBT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqsubq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQSUBT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqsubq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQSUBT.U16 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vqsubq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VQSUBT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16x2_t [__arm_]vld2q[_s8](int8_t const * addr)	addr -> Rn	VLD20.8 {Qd - Qd2},[Rn] VLD21.8 {Qd - Qd2},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1]	MVE
int16x8x2_t [__arm_]vld2q[_s16](int16_t const * addr)	addr -> Rn	VLD20.16 {Qd - Qd2},[Rn] VLD21.16 {Qd - Qd2},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1]	MVE
int32x4x2_t [__arm_]vld2q[_s32](int32_t const * addr)	addr -> Rn	VLD20.32 {Qd - Qd2},[Rn] VLD21.32 {Qd - Qd2},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1]	MVE
uint8x16x2_t [__arm_]vld2q[_u8](uint8_t const * addr)	addr -> Rn	VLD20.8 {Qd - Qd2},[Rn] VLD21.8 {Qd - Qd2},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1]	MVE
uint16x8x2_t [__arm_]vld2q[_u16](uint16_t const * addr)	addr -> Rn	VLD20.16 {Qd - Qd2},[Rn] VLD21.16 {Qd - Qd2},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1]	MVE
uint32x4x2_t [__arm_]vld2q[_u32](uint32_t const * addr)	addr -> Rn	VLD20.32 {Qd - Qd2},[Rn] VLD21.32 {Qd - Qd2},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1]	MVE
float16x8x2_t [__arm_]vld2q[_f16](float16_t const * addr)	addr -> Rn	VLD20.16 {Qd - Qd2},[Rn] VLD21.16 {Qd - Qd2},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1]	MVE
float32x4x2_t [__arm_]vld2q[_f32](float32_t const * addr)	addr -> Rn	VLD20.32 {Qd - Qd2},[Rn] VLD21.32 {Qd - Qd2},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1]	MVE
int8x16x4_t [__arm_]vld4q[_s8](int8_t const * addr)	addr -> Rn	VLD40.8 {Qd - Qd4},[Rn] VLD41.8 {Qd - Qd4},[Rn] VLD42.8 {Qd - Qd4},[Rn] VLD43.8 {Qd - Qd4},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1] Qd3 -> result.val[2] Qd4 -> result.val[3]	MVE
int16x8x4_t [__arm_]vld4q[_s16](int16_t const * addr)	addr -> Rn	VLD40.16 {Qd - Qd4},[Rn] VLD41.16 {Qd - Qd4},[Rn] VLD42.16 {Qd - Qd4},[Rn] VLD43.16 {Qd - Qd4},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1] Qd3 -> result.val[2] Qd4 -> result.val[3]	MVE
int32x4x4_t [__arm_]vld4q[_s32](int32_t const * addr)	addr -> Rn	VLD40.32 {Qd - Qd4},[Rn] VLD41.32 {Qd - Qd4},[Rn] VLD42.32 {Qd - Qd4},[Rn] VLD43.32 {Qd - Qd4},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1] Qd3 -> result.val[2] Qd4 -> result.val[3]	MVE
uint8x16x4_t [__arm_]vld4q[_u8](uint8_t const * addr)	addr -> Rn	VLD40.8 {Qd - Qd4},[Rn] VLD41.8 {Qd - Qd4},[Rn] VLD42.8 {Qd - Qd4},[Rn] VLD43.8 {Qd - Qd4},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1] Qd3 -> result.val[2] Qd4 -> result.val[3]	MVE
uint16x8x4_t [__arm_]vld4q[_u16](uint16_t const * addr)	addr -> Rn	VLD40.16 {Qd - Qd4},[Rn] VLD41.16 {Qd - Qd4},[Rn] VLD42.16 {Qd - Qd4},[Rn] VLD43.16 {Qd - Qd4},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1] Qd3 -> result.val[2] Qd4 -> result.val[3]	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4x4_t [__arm_]vld4q[_u32](uint32_t const * addr)	addr -> Rn	VLD40.32 {Qd - Qd4},[Rn] VLD41.32 {Qd - Qd4},[Rn] VLD42.32 {Qd - Qd4},[Rn] VLD43.32 {Qd - Qd4},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1] Qd3 -> result.val[2] Qd4 -> result.val[3]	MVE
float16x8x4_t [__arm_]vld4q[_f16](float16_t const * addr)	addr -> Rn	VLD40.16 {Qd - Qd4},[Rn] VLD41.16 {Qd - Qd4},[Rn] VLD42.16 {Qd - Qd4},[Rn] VLD43.16 {Qd - Qd4},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1] Qd3 -> result.val[2] Qd4 -> result.val[3]	MVE
float32x4x4_t [__arm_]vld4q[_f32](float32_t const * addr)	addr -> Rn	VLD40.32 {Qd - Qd4},[Rn] VLD41.32 {Qd - Qd4},[Rn] VLD42.32 {Qd - Qd4},[Rn] VLD43.32 {Qd - Qd4},[Rn]	Qd -> result.val[0] Qd2 -> result.val[1] Qd3 -> result.val[2] Qd4 -> result.val[3]	MVE
int8x16_t [__arm_]vldrbq_s8(int8_t const * base)	base -> Rn	VLDRB.8 Qd,[Rn]	Qd -> result	MVE
int16x8_t [__arm_]vldrbq_s16(int8_t const * base)	base -> Rn	VLDRB.S16 Qd,[Rn]	Qd -> result	MVE
int32x4_t [__arm_]vldrbq_s32(int8_t const * base)	base -> Rn	VLDRB.S32 Qd,[Rn]	Qd -> result	MVE
uint8x16_t [__arm_]vldrbq_u8(uint8_t const * base)	base -> Rn	VLDRB.8 Qd,[Rn]	Qd -> result	MVE
uint16x8_t [__arm_]vldrbq_u16(uint8_t const * base)	base -> Rn	VLDRB.U16 Qd,[Rn]	Qd -> result	MVE
uint32x4_t [__arm_]vldrbq_u32(uint8_t const * base)	base -> Rn	VLDRB.U32 Qd,[Rn]	Qd -> result	MVE
int8x16_t [__arm_]vldrbq_z_s8(int8_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRBT.8 Qd,[Rn]	Qd -> result	MVE
int16x8_t [__arm_]vldrbq_z_s16(int8_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRBT.S16 Qd,[Rn]	Qd -> result	MVE
int32x4_t [__arm_]vldrbq_z_s32(int8_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRBT.S32 Qd,[Rn]	Qd -> result	MVE
uint8x16_t [__arm_]vldrbq_z_u8(uint8_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRBT.8 Qd,[Rn]	Qd -> result	MVE
uint16x8_t [__arm_]vldrbq_z_u16(uint8_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRBT.U16 Qd,[Rn]	Qd -> result	MVE
uint32x4_t [__arm_]vldrbq_z_u32(uint8_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRBT.U32 Qd,[Rn]	Qd -> result	MVE
int16x8_t [__arm_]vldrhq_s16(int16_t const * base)	base -> Rn	VLDRH.16 Qd,[Rn]	Qd -> result	MVE
int32x4_t [__arm_]vldrhq_s32(int16_t const * base)	base -> Rn	VLDRH.S32 Qd,[Rn]	Qd -> result	MVE
uint16x8_t [__arm_]vldrhq_u16(uint16_t const * base)	base -> Rn	VLDRH.16 Qd,[Rn]	Qd -> result	MVE
uint32x4_t [__arm_]vldrhq_u32(uint16_t const * base)	base -> Rn	VLDRH.U32 Qd,[Rn]	Qd -> result	MVE
float16x8_t [__arm_]vldrhq_f16(float16_t const * base)	base -> Rn	VLDRH.16 Qd,[Rn]	Qd -> result	MVE
int16x8_t [__arm_]vldrhq_z_s16(int16_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRHT.S16 Qd,[Rn]	Qd -> result	MVE
int32x4_t [__arm_]vldrhq_z_s32(int16_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRHT.S32 Qd,[Rn]	Qd -> result	MVE
uint16x8_t [__arm_]vldrhq_z_u16(uint16_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRHT.U16 Qd,[Rn]	Qd -> result	MVE
uint32x4_t [__arm_]vldrhq_z_u32(uint16_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRHT.U32 Qd,[Rn]	Qd -> result	MVE
float16x8_t [__arm_]vldrhq_z_f16(float16_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRHT.F16 Qd,[Rn]	Qd -> result	MVE
int32x4_t [__arm_]vldrwq_s32(int32_t const * base)	base -> Rn	VLDRW.32 Qd,[Rn]	Qd -> result	MVE
uint32x4_t [__arm_]vldrwq_u32(uint32_t const * base)	base -> Rn	VLDRW.32 Qd,[Rn]	Qd -> result	MVE
float32x4_t [__arm_]vldrwq_f32(float32_t const * base)	base -> Rn	VLDRW.32 Qd,[Rn]	Qd -> result	MVE
int32x4_t [__arm_]vldrwq_z_s32(int32_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRWT.32 Qd,[Rn]	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vldrwq_z_u32(uint32_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRWT.32 Qd,[Rn]	Qd -> result	MVE
float32x4_t [__arm_]vldrwq_z_f32(float32_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRWT.32 Qd,[Rn]	Qd -> result	MVE
int8x16_t [__arm_]vld1q[_s8](int8_t const * base)	base -> Rn	VLDRB.8 Qd,[Rn]	Qd -> result	MVE/NEON
int16x8_t [__arm_]vld1q[_s16](int16_t const * base)	base -> Rn	VLDRH.16 Qd,[Rn]	Qd -> result	MVE/NEON
int32x4_t [__arm_]vld1q[_s32](int32_t const * base)	base -> Rn	VLDRW.32 Qd,[Rn]	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vld1q[_u8](uint8_t const * base)	base -> Rn	VLDRB.8 Qd,[Rn]	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vld1q[_u16](uint16_t const * base)	base -> Rn	VLDRH.16 Qd,[Rn]	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vld1q[_u32](uint32_t const * base)	base -> Rn	VLDRW.32 Qd,[Rn]	Qd -> result	MVE/NEON
float16x8_t [__arm_]vld1q[_f16](float16_t const * base)	base -> Rn	VLDRH.16 Qd,[Rn]	Qd -> result	MVE/NEON
float32x4_t [__arm_]vld1q[_f32](float32_t const * base)	base -> Rn	VLDRW.32 Qd,[Rn]	Qd -> result	MVE/NEON
int8x16_t [__arm_]vld1q_z[_s8](int8_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRBT.8 Qd,[Rn]	Qd -> result	MVE
int16x8_t [__arm_]vld1q_z[_s16](int16_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRHT.16 Qd,[Rn]	Qd -> result	MVE
int32x4_t [__arm_]vld1q_z[_s32](int32_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRWT.32 Qd,[Rn]	Qd -> result	MVE
uint8x16_t [__arm_]vld1q_z[_u8](uint8_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRBT.8 Qd,[Rn]	Qd -> result	MVE
uint16x8_t [__arm_]vld1q_z[_u16](uint16_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRHT.16 Qd,[Rn]	Qd -> result	MVE
uint32x4_t [__arm_]vld1q_z[_u32](uint32_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRWT.32 Qd,[Rn]	Qd -> result	MVE
float16x8_t [__arm_]vld1q_z[_f16](float16_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRHT.16 Qd,[Rn]	Qd -> result	MVE
float32x4_t [__arm_]vld1q_z[_f32](float32_t const * base, mve_pred16_t p)	base -> Rn p -> Rp	VMSR P0,Rp VPST VLDRWT.32 Qd,[Rn]	Qd -> result	MVE
int16x8_t [__arm_]vldrhq_gather_offset[_s16](int16_t const * base, uint16x8_t offset)	base -> Rn offset -> Qm	VLDRH.U16 Qd,[Rn,Qm]	Qd -> result	MVE
int32x4_t [__arm_]vldrhq_gather_offset[_s32](int16_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRH.S32 Qd,[Rn,Qm]	Qd -> result	MVE
uint16x8_t [__arm_]vldrhq_gather_offset[_u16](uint16_t const * base, uint16x8_t offset)	base -> Rn offset -> Qm	VLDRH.U16 Qd,[Rn,Qm]	Qd -> result	MVE
uint32x4_t [__arm_]vldrhq_gather_offset[_u32](uint16_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRH.U32 Qd,[Rn,Qm]	Qd -> result	MVE
float16x8_t [__arm_]vldrhq_gather_offset[_f16](float16_t const * base, uint16x8_t offset)	base -> Rn offset -> Qm	VLDRH.F16 Qd,[Rn,Qm]	Qd -> result	MVE
int16x8_t [__arm_]vldrhq_gather_offset_z[_s16](int16_t const * base, uint16x8_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.U16 Qd,[Rn,Qm]	Qd -> result	MVE
int32x4_t [__arm_]vldrhq_gather_offset_z[_s32](int16_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.S32 Qd,[Rn,Qm]	Qd -> result	MVE
uint16x8_t [__arm_]vldrhq_gather_offset_z[_u16](uint16_t const * base, uint16x8_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.U16 Qd,[Rn,Qm]	Qd -> result	MVE
uint32x4_t [__arm_]vldrhq_gather_offset_z[_u32](uint16_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.U32 Qd,[Rn,Qm]	Qd -> result	MVE
float16x8_t [__arm_]vldrhq_gather_offset_z[_f16](float16_t const * base, uint16x8_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.F16 Qd,[Rn,Qm]	Qd -> result	MVE
int16x8_t [__arm_]vldrhq_gather_shifted_offset[_s16](int16_t const * base, uint16x8_t offset)	base -> Rn offset -> Qm	VLDRH.U16 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
int32x4_t [__arm_]vldrhq_gather_shifted_offset[_s32](int16_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRH.S32 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
uint16x8_t [__arm_]vldrhq_gather_shifted_offset[_u16](uint16_t const * base, uint16x8_t offset)	base -> Rn offset -> Qm	VLDRH.U16 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
uint32x4_t [__arm_]vldrhq_gather_shifted_offset[_u32](uint16_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRH.U32 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vldrhq_gather_shifted_offset[_f16](float16_t const * base, uint16x8_t offset)	base -> Rn offset -> Qm	VLDRH.F16 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
int16x8_t [__arm_]vldrhq_gather_shifted_offset_z[_s16](int16_t const * base, uint16x8_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.U16 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
int32x4_t [__arm_]vldrhq_gather_shifted_offset_z[_s32](int16_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.S32 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
uint16x8_t [__arm_]vldrhq_gather_shifted_offset_z[_u16](uint16_t const * base, uint16x8_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.U16 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
uint32x4_t [__arm_]vldrhq_gather_shifted_offset_z[_u32](uint16_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.U32 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
float16x8_t [__arm_]vldrhq_gather_shifted_offset_z[_f16](float16_t const * base, uint16x8_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRHT.F16 Qd,[Rn,Qm,UXTW #1]	Qd -> result	MVE
int8x16_t [__arm_]vldrbq_gather_offset[_s8](int8_t const * base, uint8x16_t offset)	base -> Rn offset -> Qm	VLDRB.U8 Qd,[Rn,Qm]	Qd -> result	MVE
int16x8_t [__arm_]vldrbq_gather_offset[_s16](int8_t const * base, uint16x8_t offset)	base -> Rn offset -> Qm	VLDRB.S16 Qd,[Rn,Qm]	Qd -> result	MVE
int32x4_t [__arm_]vldrbq_gather_offset[_s32](int8_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRB.S32 Qd,[Rn,Qm]	Qd -> result	MVE
uint8x16_t [__arm_]vldrbq_gather_offset[_u8](uint8_t const * base, uint8x16_t offset)	base -> Rn offset -> Qm	VLDRB.U8 Qd,[Rn,Qm]	Qd -> result	MVE
uint16x8_t [__arm_]vldrbq_gather_offset[_u16](uint8_t const * base, uint16x8_t offset)	base -> Rn offset -> Qm	VLDRB.U16 Qd,[Rn,Qm]	Qd -> result	MVE
uint32x4_t [__arm_]vldrbq_gather_offset[_u32](uint8_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRB.U32 Qd,[Rn,Qm]	Qd -> result	MVE
int8x16_t [__arm_]vldrbq_gather_offset_z[_s8](int8_t const * base, uint8x16_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRBT.U8 Qd,[Rn,Qm]	Qd -> result	MVE
int16x8_t [__arm_]vldrbq_gather_offset_z[_s16](int8_t const * base, uint16x8_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRBT.S16 Qd,[Rn,Qm]	Qd -> result	MVE
int32x4_t [__arm_]vldrbq_gather_offset_z[_s32](int8_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRBT.S32 Qd,[Rn,Qm]	Qd -> result	MVE
uint8x16_t [__arm_]vldrbq_gather_offset_z[_u8](uint8_t const * base, uint8x16_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRBT.U8 Qd,[Rn,Qm]	Qd -> result	MVE
uint16x8_t [__arm_]vldrbq_gather_offset_z[_u16](uint8_t const * base, uint16x8_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRBT.U16 Qd,[Rn,Qm]	Qd -> result	MVE
uint32x4_t [__arm_]vldrbq_gather_offset_z[_u32](uint8_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRBT.U32 Qd,[Rn,Qm]	Qd -> result	MVE
int32x4_t [__arm_]vldrwq_gather_offset[_s32](int32_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRW.U32 Qd,[Rn,Qm]	Qd -> result	MVE
uint32x4_t [__arm_]vldrwq_gather_offset[_u32](uint32_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRW.U32 Qd,[Rn,Qm]	Qd -> result	MVE
float32x4_t [__arm_]vldrwq_gather_offset[_f32](float32_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRW.U32 Qd,[Rn,Qm]	Qd -> result	MVE
int32x4_t [__arm_]vldrwq_gather_offset_z[_s32](int32_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Rn,Qm]	Qd -> result	MVE
uint32x4_t [__arm_]vldrwq_gather_offset_z[_u32](uint32_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Rn,Qm]	Qd -> result	MVE
float32x4_t [__arm_]vldrwq_gather_offset_z[_f32](float32_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Rn,Qm]	Qd -> result	MVE
int32x4_t [__arm_]vldrwq_gather_shifted_offset[_s32](int32_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRW.U32 Qd,[Rn,Qm,UXTW #2]	Qd -> result	MVE
uint32x4_t [__arm_]vldrwq_gather_shifted_offset[_u32](uint32_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRW.U32 Qd,[Rn,Qm,UXTW #2]	Qd -> result	MVE
float32x4_t [__arm_]vldrwq_gather_shifted_offset[_f32](float32_t const * base, uint32x4_t offset)	base -> Rn offset -> Qm	VLDRW.U32 Qd,[Rn,Qm,UXTW #2]	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vldrwq_gather_shifted_offset_z[_s32](int32_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Rn,Qm,UXTW #2]	Qd -> result	MVE
uint32x4_t [__arm_]vldrwq_gather_shifted_offset_z[_u32](uint32_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Rn,Qm,UXTW #2]	Qd -> result	MVE
float32x4_t [__arm_]vldrwq_gather_shifted_offset_z[_f32](float32_t const * base, uint32x4_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Rn,Qm,UXTW #2]	Qd -> result	MVE
int32x4_t [__arm_]vldrwq_gather_base_s32(uint32x4_t addr, const int offset)	addr -> Qn offset in +/- 4*[0..127]	VLDRW.U32 Qd,[Qn,#offset]	Qd -> result	MVE
uint32x4_t [__arm_]vldrwq_gather_base_u32(uint32x4_t addr, const int offset)	addr -> Qn offset in +/- 4*[0..127]	VLDRW.U32 Qd,[Qn,#offset]	Qd -> result	MVE
float32x4_t [__arm_]vldrwq_gather_base_f32(uint32x4_t addr, const int offset)	addr -> Qn offset in +/- 4*[0..127]	VLDRW.U32 Qd,[Qn,#offset]	Qd -> result	MVE
int32x4_t [__arm_]vldrwq_gather_base_z_s32(uint32x4_t addr, const int offset, mve_pred16_t p)	addr -> Qn offset in +/- 4*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Qn,#offset]	Qd -> result	MVE
uint32x4_t [__arm_]vldrwq_gather_base_z_u32(uint32x4_t addr, const int offset, mve_pred16_t p)	addr -> Qn offset in +/- 4*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Qn,#offset]	Qd -> result	MVE
float32x4_t [__arm_]vldrwq_gather_base_z_f32(uint32x4_t addr, const int offset, mve_pred16_t p)	addr -> Qn offset in +/- 4*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Qn,#offset]	Qd -> result	MVE
int32x4_t [__arm_]vldrwq_gather_base_wb_s32(uint32x4_t * addr, const int offset)	*addr -> Qn offset in +/- 4*[0..127]	VLDRW.U32 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
uint32x4_t [__arm_]vldrwq_gather_base_wb_u32(uint32x4_t * addr, const int offset)	*addr -> Qn offset in +/- 4*[0..127]	VLDRW.U32 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
float32x4_t [__arm_]vldrwq_gather_base_wb_f32(uint32x4_t * addr, const int offset)	*addr -> Qn offset in +/- 4*[0..127]	VLDRW.U32 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
int32x4_t [__arm_]vldrwq_gather_base_wb_z_s32(uint32x4_t * addr, const int offset, mve_pred16_t p)	*addr -> Qn offset in +/- 4*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
uint32x4_t [__arm_]vldrwq_gather_base_wb_z_u32(uint32x4_t * addr, const int offset, mve_pred16_t p)	*addr -> Qn offset in +/- 4*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
float32x4_t [__arm_]vldrwq_gather_base_wb_z_f32(uint32x4_t * addr, const int offset, mve_pred16_t p)	*addr -> Qn offset in +/- 4*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRWT.U32 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
int64x2_t [__arm_]vldrdq_gather_offset[_s64](int64_t const * base, uint64x2_t offset)	base -> Rn offset -> Qm	VLDRD.U64 Qd,[Rn,Qm]	Qd -> result	MVE
uint64x2_t [__arm_]vldrdq_gather_offset[_u64](uint64_t const * base, uint64x2_t offset)	base -> Rn offset -> Qm	VLDRD.U64 Qd,[Rn,Qm]	Qd -> result	MVE
int64x2_t [__arm_]vldrdq_gather_offset_z[_s64](int64_t const * base, uint64x2_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRDT.U64 Qd,[Rn,Qm]	Qd -> result	MVE
uint64x2_t [__arm_]vldrdq_gather_offset_z[_u64](uint64_t const * base, uint64x2_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRDT.U64 Qd,[Rn,Qm]	Qd -> result	MVE
int64x2_t [__arm_]vldrdq_gather_shifted_offset[_s64](int64_t const * base, uint64x2_t offset)	base -> Rn offset -> Qm	VLDRD.U64 Qd,[Rn,Qm,UXTW #3]	Qd -> result	MVE
uint64x2_t [__arm_]vldrdq_gather_shifted_offset[_u64](uint64_t const * base, uint64x2_t offset)	base -> Rn offset -> Qm	VLDRD.U64 Qd,[Rn,Qm,UXTW #3]	Qd -> result	MVE
int64x2_t [__arm_]vldrdq_gather_shifted_offset_z[_s64](int64_t const * base, uint64x2_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRDT.U64 Qd,[Rn,Qm,UXTW #3]	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint64x2_t [__arm_]vldrdq_gather_shifted_offset_z[_u64](uint64_t const * base, uint64x2_t offset, mve_pred16_t p)	base -> Rn offset -> Qm p -> Rp	VMSR P0,Rp VPST VLDRDT.U64 Qd,[Rn,Qm,UXTW #3]	Qd -> result	MVE
int64x2_t [__arm_]vldrdq_gather_base_s64(uint64x2_t addr, const int offset)	addr -> Qn offset in +/- 8*[0..127]	VLDRD.64 Qd,[Qn,#offset]	Qd -> result	MVE
uint64x2_t [__arm_]vldrdq_gather_base_u64(uint64x2_t addr, const int offset)	addr -> Qn offset in +/- 8*[0..127]	VLDRD.64 Qd,[Qn,#offset]	Qd -> result	MVE
int64x2_t [__arm_]vldrdq_gather_base_z_s64(uint64x2_t addr, const int offset, mve_pred16_t p)	addr -> Qn offset in +/- 8*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRDT.U64 Qd,[Qn,#offset]	Qd -> result	MVE
uint64x2_t [__arm_]vldrdq_gather_base_z_u64(uint64x2_t addr, const int offset, mve_pred16_t p)	addr -> Qn offset in +/- 8*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRDT.U64 Qd,[Qn,#offset]	Qd -> result	MVE
int64x2_t [__arm_]vldrdq_gather_base_wb_s64(uint64x2_t * addr, const int offset)	*addr -> Qn offset in +/- 8*[0..127]	VLDRD.64 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
uint64x2_t [__arm_]vldrdq_gather_base_wb_u64(uint64x2_t * addr, const int offset)	*addr -> Qn offset in +/- 8*[0..127]	VLDRD.64 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
int64x2_t [__arm_]vldrdq_gather_base_wb_z_s64(uint64x2_t * addr, const int offset, mve_pred16_t p)	*addr -> Qn offset in +/- 8*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRDT.U64 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
uint64x2_t [__arm_]vldrdq_gather_base_wb_z_u64(uint64x2_t * addr, const int offset, mve_pred16_t p)	*addr -> Qn offset in +/- 8*[0..127] p -> Rp	VMSR P0,Rp VPST VLDRDT.U64 Qd,[Qn,#offset]!	Qd -> result Qn -> *addr	MVE
void [__arm_]vst2q[_s8](int8_t * addr, int8x16x2_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2	VST20.8 {Qd - Qd2},[Rn] VST21.8 {Qd - Qd2},[Rn]	void -> result	MVE
void [__arm_]vst2q[_s16](int16_t * addr, int16x8x2_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2	VST20.16 {Qd - Qd2},[Rn] VST21.16 {Qd - Qd2},[Rn]	void -> result	MVE
void [__arm_]vst2q[_s32](int32_t * addr, int32x4x2_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2	VST20.32 {Qd - Qd2},[Rn] VST21.32 {Qd - Qd2},[Rn]	void -> result	MVE
void [__arm_]vst2q[_u8](uint8_t * addr, uint8x16x2_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2	VST20.8 {Qd - Qd2},[Rn] VST21.8 {Qd - Qd2},[Rn]	void -> result	MVE
void [__arm_]vst2q[_u16](uint16_t * addr, uint16x8x2_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2	VST20.16 {Qd - Qd2},[Rn] VST21.16 {Qd - Qd2},[Rn]	void -> result	MVE
void [__arm_]vst2q[_u32](uint32_t * addr, uint32x4x2_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2	VST20.32 {Qd - Qd2},[Rn] VST21.32 {Qd - Qd2},[Rn]	void -> result	MVE
void [__arm_]vst2q[_f16](float16_t * addr, float16x8x2_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2	VST20.16 {Qd - Qd2},[Rn] VST21.16 {Qd - Qd2},[Rn]	void -> result	MVE
void [__arm_]vst2q[_f32](float32_t * addr, float32x4x2_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2	VST20.32 {Qd - Qd2},[Rn] VST21.32 {Qd - Qd2},[Rn]	void -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
void [__arm_]vst4q[_s8](int8_t * addr, int8x16x4_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2 value.val[2] -> Qd3 value.val[3] -> Qd4	VST40.8 {Qd - Qd4},[Rn] VST41.8 {Qd - Qd4},[Rn] VST42.8 {Qd - Qd4},[Rn] VST43.8 {Qd - Qd4},[Rn]	void -> result	MVE
void [__arm_]vst4q[_s16](int16_t * addr, int16x8x4_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2 value.val[2] -> Qd3 value.val[3] -> Qd4	VST40.16 {Qd - Qd4},[Rn] VST41.16 {Qd - Qd4},[Rn] VST42.16 {Qd - Qd4},[Rn] VST43.16 {Qd - Qd4},[Rn]	void -> result	MVE
void [__arm_]vst4q[_s32](int32_t * addr, int32x4x4_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2 value.val[2] -> Qd3 value.val[3] -> Qd4	VST40.32 {Qd - Qd4},[Rn] VST41.32 {Qd - Qd4},[Rn] VST42.32 {Qd - Qd4},[Rn] VST43.32 {Qd - Qd4},[Rn]	void -> result	MVE
void [__arm_]vst4q[_u8](uint8_t * addr, uint8x16x4_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2 value.val[2] -> Qd3 value.val[3] -> Qd4	VST40.8 {Qd - Qd4},[Rn] VST41.8 {Qd - Qd4},[Rn] VST42.8 {Qd - Qd4},[Rn] VST43.8 {Qd - Qd4},[Rn]	void -> result	MVE
void [__arm_]vst4q[_u16](uint16_t * addr, uint16x8x4_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2 value.val[2] -> Qd3 value.val[3] -> Qd4	VST40.16 {Qd - Qd4},[Rn] VST41.16 {Qd - Qd4},[Rn] VST42.16 {Qd - Qd4},[Rn] VST43.16 {Qd - Qd4},[Rn]	void -> result	MVE
void [__arm_]vst4q[_u32](uint32_t * addr, uint32x4x4_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2 value.val[2] -> Qd3 value.val[3] -> Qd4	VST40.32 {Qd - Qd4},[Rn] VST41.32 {Qd - Qd4},[Rn] VST42.32 {Qd - Qd4},[Rn] VST43.32 {Qd - Qd4},[Rn]	void -> result	MVE
void [__arm_]vst4q[_f16](float16_t * addr, float16x8x4_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2 value.val[2] -> Qd3 value.val[3] -> Qd4	VST40.16 {Qd - Qd4},[Rn] VST41.16 {Qd - Qd4},[Rn] VST42.16 {Qd - Qd4},[Rn] VST43.16 {Qd - Qd4},[Rn]	void -> result	MVE
void [__arm_]vst4q[_f32](float32_t * addr, float32x4x4_t value)	addr -> Rn value.val[0] -> Qd value.val[1] -> Qd2 value.val[2] -> Qd3 value.val[3] -> Qd4	VST40.32 {Qd - Qd4},[Rn] VST41.32 {Qd - Qd4},[Rn] VST42.32 {Qd - Qd4},[Rn] VST43.32 {Qd - Qd4},[Rn]	void -> result	MVE
void [__arm_]vstrbq[_s8](int8_t * base, int8x16_t value)	base -> Rn value -> Qd	VSTRB.8 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq[_s16](int8_t * base, int16x8_t value)	base -> Rn value -> Qd	VSTRB.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq[_s32](int8_t * base, int32x4_t value)	base -> Rn value -> Qd	VSTRB.32 Qd,[Rn]	void -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
void [__arm_]vstrbq[_u8](uint8_t * base, uint8x16_t value)	base -> Rn value -> Qd	VSTRB.8 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq[_u16](uint8_t * base, uint16x8_t value)	base -> Rn value -> Qd	VSTRB.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq[_u32](uint8_t * base, uint32x4_t value)	base -> Rn value -> Qd	VSTRB.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq_p[_s8](int8_t * base, int8x16_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.8 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq_p[_s16](int8_t * base, int16x8_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq_p[_s32](int8_t * base, int32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq_p[_u8](uint8_t * base, uint8x16_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.8 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq_p[_u16](uint8_t * base, uint16x8_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq_p[_u32](uint8_t * base, uint32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq[_s16](int16_t * base, int16x8_t value)	base -> Rn value -> Qd	VSTRH.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq[_s32](int16_t * base, int32x4_t value)	base -> Rn value -> Qd	VSTRH.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq[_u16](uint16_t * base, uint16x8_t value)	base -> Rn value -> Qd	VSTRH.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq[_u32](uint16_t * base, uint32x4_t value)	base -> Rn value -> Qd	VSTRH.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq[_f16](float16_t * base, float16x8_t value)	base -> Rn value -> Qd	VSTRH.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq_p[_s16](int16_t * base, int16x8_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq_p[_s32](int16_t * base, int32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq_p[_u16](uint16_t * base, uint16x8_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq_p[_u32](uint16_t * base, uint32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrhq_p[_f16](float16_t * base, float16x8_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrwq[_s32](int32_t * base, int32x4_t value)	base -> Rn value -> Qd	VSTRW.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrwq[_u32](uint32_t * base, uint32x4_t value)	base -> Rn value -> Qd	VSTRW.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrwq[_f32](float32_t * base, float32x4_t value)	base -> Rn value -> Qd	VSTRW.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrwq_p[_s32](int32_t * base, int32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrwq_p[_u32](uint32_t * base, uint32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrwq_p[_f32](float32_t * base, float32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vst1q[_s8](int8_t * base, int8x16_t value)	base -> Rn value -> Qd	VSTRB.8 Qd,[Rn]	void -> result	MVE/NEON
void [__arm_]vst1q[_s16](int16_t * base, int16x8_t value)	base -> Rn value -> Qd	VSTRH.16 Qd,[Rn]	void -> result	MVE/NEON
void [__arm_]vst1q[_s32](int32_t * base, int32x4_t value)	base -> Rn value -> Qd	VSTRW.32 Qd,[Rn]	void -> result	MVE/NEON
void [__arm_]vst1q[_u8](uint8_t * base, uint8x16_t value)	base -> Rn value -> Qd	VSTRB.8 Qd,[Rn]	void -> result	MVE/NEON
void [__arm_]vst1q[_u16](uint16_t * base, uint16x8_t value)	base -> Rn value -> Qd	VSTRH.16 Qd,[Rn]	void -> result	MVE/NEON
void [__arm_]vst1q[_u32](uint32_t * base, uint32x4_t value)	base -> Rn value -> Qd	VSTRW.32 Qd,[Rn]	void -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
void [__arm_]vst1q[_f16](float16_t * base, float16x8_t value)	base -> Rn value -> Qd	VSTRH.16 Qd,[Rn]	void -> result	MVE/NEON
void [__arm_]vst1q[_f32](float32_t * base, float32x4_t value)	base -> Rn value -> Qd	VSTRW.32 Qd,[Rn]	void -> result	MVE/NEON
void [__arm_]vst1q_p[_s8](int8_t * base, int8x16_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.8 Qd,[Rn]	void -> result	MVE
void [__arm_]vst1q_p[_s16](int16_t * base, int16x8_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vst1q_p[_s32](int32_t * base, int32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vst1q_p[_u8](uint8_t * base, uint8x16_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.8 Qd,[Rn]	void -> result	MVE
void [__arm_]vst1q_p[_u16](uint16_t * base, uint16x8_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vst1q_p[_u32](uint32_t * base, uint32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vst1q_p[_f16](float16_t * base, float16x8_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn]	void -> result	MVE
void [__arm_]vst1q_p[_f32](float32_t * base, float32x4_t value, mve_pred16_t p)	base -> Rn value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset[_s8](int8_t * base, uint8x16_t offset, int8x16_t value)	base -> Rn offset -> Qm value -> Qd	VSTRB.8 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset[_s16](int8_t * base, uint16x8_t offset, int16x8_t value)	base -> Rn offset -> Qm value -> Qd	VSTRB.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset[_s32](int8_t * base, uint32x4_t offset, int32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRB.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset[_u8](uint8_t * base, uint8x16_t offset, uint8x16_t value)	base -> Rn offset -> Qm value -> Qd	VSTRB.8 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset[_u16](uint8_t * base, uint16x8_t offset, uint16x8_t value)	base -> Rn offset -> Qm value -> Qd	VSTRB.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset[_u32](uint8_t * base, uint32x4_t offset, uint32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRB.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset_p[_s8](int8_t * base, uint8x16_t offset, int8x16_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.8 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset_p[_s16](int8_t * base, uint16x8_t offset, int16x8_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset_p[_s32](int8_t * base, uint32x4_t offset, int32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset_p[_u8](uint8_t * base, uint8x16_t offset, uint8x16_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.8 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset_p[_u16](uint8_t * base, uint16x8_t offset, uint16x8_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrbq_scatter_offset_p[_u32](uint8_t * base, uint32x4_t offset, uint32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRBT.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset[_s16](int16_t * base, uint16x8_t offset, int16x8_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset[_s32](int16_t * base, uint32x4_t offset, int32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.32 Qd,[Rn,Qm]	void -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
void [__arm_]vstrhq_scatter_offset[_u16](uint16_t * base, uint16x8_t offset, uint16x8_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset[_u32](uint16_t * base, uint32x4_t offset, uint32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset[_f16](float16_t * base, uint16x8_t offset, float16x8_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset_p[_s16](int16_t * base, uint16x8_t offset, int16x8_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset_p[_s32](int16_t * base, uint32x4_t offset, int32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset_p[_u16](uint16_t * base, uint16x8_t offset, uint16x8_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset_p[_u32](uint16_t * base, uint32x4_t offset, uint32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_offset_p[_f16](float16_t * base, uint16x8_t offset, float16x8_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset[_s16](int16_t * base, uint16x8_t offset, int16x8_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.16 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset[_s32](int16_t * base, uint32x4_t offset, int32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.32 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset[_u16](uint16_t * base, uint16x8_t offset, uint16x8_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.16 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset[_u32](uint16_t * base, uint32x4_t offset, uint32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.32 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset[_f16](float16_t * base, uint16x8_t offset, float16x8_t value)	base -> Rn offset -> Qm value -> Qd	VSTRH.16 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset_p[_s16](int16_t * base, uint16x8_t offset, int16x8_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset_p[_s32](int16_t * base, uint32x4_t offset, int32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.32 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset_p[_u16](uint16_t * base, uint16x8_t offset, uint16x8_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset_p[_u32](uint16_t * base, uint32x4_t offset, uint32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.32 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrhq_scatter_shifted_offset_p[_f16](float16_t * base, uint16x8_t offset, float16x8_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRHT.16 Qd,[Rn,Qm,UXTW #1]	void -> result	MVE
void [__arm_]vstrwq_scatter_base[_s32](uint32x4_t addr, const int offset, int32x4_t value)	addr -> Qn offset in +/- 4*[0..127] value -> Qd	VSTRW.U32 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrwq_scatter_base[_u32](uint32x4_t addr, const int offset, uint32x4_t value)	addr -> Qn offset in +/- 4*[0..127] value -> Qd	VSTRW.U32 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrwq_scatter_base[_f32](uint32x4_t addr, const int offset, float32x4_t value)	addr -> Qn offset in +/- 4*[0..127] value -> Qd	VSTRW.U32 Qd,[Qn,#offset]	void -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
void [__arm_]vstrwq_scatter_base_p[_s32](uint32x4_t addr, const int offset, int32x4_t value, mve_pred16_t p)	addr -> Qn offset in +/- 4*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.U32 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrwq_scatter_base_p[_u32](uint32x4_t addr, const int offset, uint32x4_t value, mve_pred16_t p)	addr -> Qn offset in +/- 4*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.U32 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrwq_scatter_base_p[_f32](uint32x4_t addr, const int offset, float32x4_t value, mve_pred16_t p)	addr -> Qn offset in +/- 4*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.U32 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrwq_scatter_base_wb[_s32](uint32x4_t * addr, const int offset, int32x4_t value)	*addr -> Qn offset in +/- 4*[0..127] value -> Qd	VSTRW.U32 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrwq_scatter_base_wb[_u32](uint32x4_t * addr, const int offset, uint32x4_t value)	*addr -> Qn offset in +/- 4*[0..127] value -> Qd	VSTRW.U32 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrwq_scatter_base_wb[_f32](uint32x4_t * addr, const int offset, float32x4_t value)	*addr -> Qn offset in +/- 4*[0..127] value -> Qd	VSTRW.U32 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrwq_scatter_base_wb_p[_s32](uint32x4_t * addr, const int offset, int32x4_t value, mve_pred16_t p)	*addr -> Qn offset in +/- 4*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.U32 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrwq_scatter_base_wb_p[_u32](uint32x4_t * addr, const int offset, uint32x4_t value, mve_pred16_t p)	*addr -> Qn offset in +/- 4*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.U32 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrwq_scatter_base_wb_p[_f32](uint32x4_t * addr, const int offset, float32x4_t value, mve_pred16_t p)	*addr -> Qn offset in +/- 4*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.U32 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrwq_scatter_offset[_s32](int32_t * base, uint32x4_t offset, int32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRW.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrwq_scatter_offset[_u32](uint32_t * base, uint32x4_t offset, uint32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRW.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrwq_scatter_offset[_f32](float32_t * base, uint32x4_t offset, float32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRW.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrwq_scatter_offset_p[_s32](int32_t * base, uint32x4_t offset, int32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrwq_scatter_offset_p[_u32](uint32_t * base, uint32x4_t offset, uint32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrwq_scatter_offset_p[_f32](float32_t * base, uint32x4_t offset, float32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrwq_scatter_shifted_offset[_s32](int32_t * base, uint32x4_t offset, int32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRW.32 Qd,[Rn,Qm,UXTW #2]	void -> result	MVE
void [__arm_]vstrwq_scatter_shifted_offset[_u32](uint32_t * base, uint32x4_t offset, uint32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRW.32 Qd,[Rn,Qm,UXTW #2]	void -> result	MVE
void [__arm_]vstrwq_scatter_shifted_offset[_f32](float32_t * base, uint32x4_t offset, float32x4_t value)	base -> Rn offset -> Qm value -> Qd	VSTRW.32 Qd,[Rn,Qm,UXTW #2]	void -> result	MVE
void [__arm_]vstrwq_scatter_shifted_offset_p[_s32](int32_t * base, uint32x4_t offset, int32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn,Qm,UXTW #2]	void -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
void [__arm_]vstrwq_scatter_shifted_offset_p[_u32](uint32_t * base, uint32x4_t offset, uint32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn,Qm,UXTW #2]	void -> result	MVE
void [__arm_]vstrwq_scatter_shifted_offset_p[_f32](float32_t * base, uint32x4_t offset, float32x4_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRWT.32 Qd,[Rn,Qm,UXTW #2]	void -> result	MVE
void [__arm_]vstrdq_scatter_base[_s64](uint64x2_t addr, const int offset, int64x2_t value)	addr -> Qn offset in +/- 8*[0..127] value -> Qd	VSTRD.U64 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrdq_scatter_base[_u64](uint64x2_t addr, const int offset, uint64x2_t value)	addr -> Qn offset in +/- 8*[0..127] value -> Qd	VSTRD.U64 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrdq_scatter_base_p[_s64](uint64x2_t addr, const int offset, int64x2_t value, mve_pred16_t p)	addr -> Qn offset in +/- 8*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRDT.U64 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrdq_scatter_base_p[_u64](uint64x2_t addr, const int offset, uint64x2_t value, mve_pred16_t p)	addr -> Qn offset in +/- 8*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRDT.U64 Qd,[Qn,#offset]	void -> result	MVE
void [__arm_]vstrdq_scatter_base_wb[_s64](uint64x2_t * addr, const int offset, int64x2_t value)	*addr -> Qn offset in +/- 8*[0..127] value -> Qd	VSTRD.U64 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrdq_scatter_base_wb[_u64](uint64x2_t * addr, const int offset, uint64x2_t value)	*addr -> Qn offset in +/- 8*[0..127] value -> Qd	VSTRD.U64 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrdq_scatter_base_wb_p[_s64](uint64x2_t * addr, const int offset, int64x2_t value, mve_pred16_t p)	*addr -> Qn offset in +/- 8*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRDT.U64 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrdq_scatter_base_wb_p[_u64](uint64x2_t * addr, const int offset, uint64x2_t value, mve_pred16_t p)	*addr -> Qn offset in +/- 8*[0..127] value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRDT.U64 Qd,[Qn,#offset]!	void -> result	MVE
void [__arm_]vstrdq_scatter_offset[_s64](int64_t * base, uint64x2_t offset, int64x2_t value)	base -> Rn offset -> Qm value -> Qd	VSTRD.64 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrdq_scatter_offset[_u64](uint64_t * base, uint64x2_t offset, uint64x2_t value)	base -> Rn offset -> Qm value -> Qd	VSTRD.64 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrdq_scatter_offset_p[_s64](int64_t * base, uint64x2_t offset, int64x2_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRDT.64 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrdq_scatter_offset_p[_u64](uint64_t * base, uint64x2_t offset, uint64x2_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRDT.64 Qd,[Rn,Qm]	void -> result	MVE
void [__arm_]vstrdq_scatter_shifted_offset[_s64](int64_t * base, uint64x2_t offset, int64x2_t value)	base -> Rn offset -> Qm value -> Qd	VSTRD.64 Qd,[Rn,Qm,UXTW #3]	void -> result	MVE
void [__arm_]vstrdq_scatter_shifted_offset[_u64](uint64_t * base, uint64x2_t offset, uint64x2_t value)	base -> Rn offset -> Qm value -> Qd	VSTRD.64 Qd,[Rn,Qm,UXTW #3]	void -> result	MVE
void [__arm_]vstrdq_scatter_shifted_offset_p[_s64](int64_t * base, uint64x2_t offset, int64x2_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRDT.64 Qd,[Rn,Qm,UXTW #3]	void -> result	MVE
void [__arm_]vstrdq_scatter_shifted_offset_p[_u64](uint64_t * base, uint64x2_t offset, uint64x2_t value, mve_pred16_t p)	base -> Rn offset -> Qm value -> Qd p -> Rp	VMSR P0,Rp VPST VSTRDT.64 Qd,[Rn,Qm,UXTW #3]	void -> result	MVE
int64_t [__arm_]vaddlvaq[_s32](int64_t a, int32x4_t b)	a -> [RdaHi,RdaLo] b -> Qm	VADDLVA.S32 RdaLo,RdaHi,Qm	[RdaHi,RdaLo] -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint64_t [__arm_]vaddlvaq[_u32](uint64_t a, uint32x4_t b)	a -> [RdaHi,RdaLo] b -> Qm	VADDLVA.U32 RdaLo,RdaHi,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vaddlvaq_p[_s32](int64_t a, int32x4_t b, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qm p -> Rp	VMSR P0,Rp VPST VADDLVAT.S32 RdaLo,RdaHi,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vaddlvaq_p[_u32](uint64_t a, uint32x4_t b, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qm p -> Rp	VMSR P0,Rp VPST VADDLVAT.U32 RdaLo,RdaHi,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vaddlvq[_s32](int32x4_t a)	a -> Qm	VADDLV.S32 RdaLo,RdaHi,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vaddlvq[_u32](uint32x4_t a)	a -> Qm	VADDLV.U32 RdaLo,RdaHi,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vaddlvq_p[_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VADDLVT.S32 RdaLo,RdaHi,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vaddlvq_p[_u32](uint32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VADDLVT.U32 RdaLo,RdaHi,Qm	[RdaHi,RdaLo] -> result	MVE
int32_t [__arm_]vaddvaq[_s8](int32_t a, int8x16_t b)	a -> Rda b -> Qm	VADDVA.S8 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvaq[_s16](int32_t a, int16x8_t b)	a -> Rda b -> Qm	VADDVA.S16 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvaq[_s32](int32_t a, int32x4_t b)	a -> Rda b -> Qm	VADDVA.S32 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvaq[_u8](uint32_t a, uint8x16_t b)	a -> Rda b -> Qm	VADDVA.U8 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvaq[_u16](uint32_t a, uint16x8_t b)	a -> Rda b -> Qm	VADDVA.U16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvaq[_u32](uint32_t a, uint32x4_t b)	a -> Rda b -> Qm	VADDVA.U32 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvaq_p[_s8](int32_t a, int8x16_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VADDVAT.S8 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvaq_p[_s16](int32_t a, int16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VADDVAT.S16 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvaq_p[_s32](int32_t a, int32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VADDVAT.S32 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvaq_p[_u8](uint32_t a, uint8x16_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VADDVAT.U8 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvaq_p[_u16](uint32_t a, uint16x8_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VADDVAT.U16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvaq_p[_u32](uint32_t a, uint32x4_t b, mve_pred16_t p)	a -> Rda b -> Qm p -> Rp	VMSR P0,Rp VPST VADDVAT.U32 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvq[_s8](int8x16_t a)	a -> Qm	VADDV.S8 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvq[_s16](int16x8_t a)	a -> Qm	VADDV.S16 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvq[_s32](int32x4_t a)	a -> Qm	VADDV.S32 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvq[_u8](uint8x16_t a)	a -> Qm	VADDV.U8 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvq[_u16](uint16x8_t a)	a -> Qm	VADDV.U16 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvq[_u32](uint32x4_t a)	a -> Qm	VADDV.U32 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvq_p[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VADDVT.S8 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvq_p[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VADDVT.S16 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vaddvq_p[_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VADDVT.S32 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvq_p[_u8](uint8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VADDVT.U8 Rda,Qm	Rda -> result	MVE
uint32_t [__arm_]vaddvq_p[_u16](uint16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VADDVT.U16 Rda,Qm	Rda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32_t [__arm_]vaddvq_p[_u32](uint32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VADDVT.U32 Rda,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaq[_s8](int32_t add, int8x16_t m1, int8x16_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVA.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaq[_s16](int32_t add, int16x8_t m1, int16x8_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVA.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaq[_s32](int32_t add, int32x4_t m1, int32x4_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVA.S32 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavaq[_u8](uint32_t add, uint8x16_t m1, uint8x16_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVA.U8 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavaq[_u16](uint32_t add, uint16x8_t m1, uint16x8_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVA.U16 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavaq[_u32](uint32_t add, uint32x4_t m1, uint32x4_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVA.U32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaq_p[_s8](int32_t add, int8x16_t m1, int8x16_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAT.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaq_p[_s16](int32_t add, int16x8_t m1, int16x8_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAT.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaq_p[_s32](int32_t add, int32x4_t m1, int32x4_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAT.S32 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavaq_p[_u8](uint32_t add, uint8x16_t m1, uint8x16_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAT.U8 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavaq_p[_u16](uint32_t add, uint16x8_t m1, uint16x8_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAT.U16 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavaq_p[_u32](uint32_t add, uint32x4_t m1, uint32x4_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAT.U32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavq[_s8](int8x16_t m1, int8x16_t m2)	m1 -> Qn m2 -> Qm	VMLADAV.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavq[_s16](int16x8_t m1, int16x8_t m2)	m1 -> Qn m2 -> Qm	VMLADAV.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavq[_s32](int32x4_t m1, int32x4_t m2)	m1 -> Qn m2 -> Qm	VMLADAV.S32 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavq[_u8](uint8x16_t m1, uint8x16_t m2)	m1 -> Qn m2 -> Qm	VMLADAV.U8 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavq[_u16](uint16x8_t m1, uint16x8_t m2)	m1 -> Qn m2 -> Qm	VMLADAV.U16 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavq[_u32](uint32x4_t m1, uint32x4_t m2)	m1 -> Qn m2 -> Qm	VMLADAV.U32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavq_p[_s8](int8x16_t m1, int8x16_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVT.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavq_p[_s16](int16x8_t m1, int16x8_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVT.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavq_p[_s32](int32x4_t m1, int32x4_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVT.S32 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavq_p[_u8](uint8x16_t m1, uint8x16_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVT.U8 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavq_p[_u16](uint16x8_t m1, uint16x8_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVT.U16 Rda,Qn,Qm	Rda -> result	MVE
uint32_t [__arm_]vmladavq_p[_u32](uint32x4_t m1, uint32x4_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVT.U32 Rda,Qn,Qm	Rda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32_t [__arm_]vmladavaxq[_s8](int32_t add, int8x16_t m1, int8x16_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVAX.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaxq[_s16](int32_t add, int16x8_t m1, int16x8_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVAX.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaxq[_s32](int32_t add, int32x4_t m1, int32x4_t m2)	add -> Rda m1 -> Qn m2 -> Qm	VMLADAVAX.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaxq_p[_s8](int32_t add, int8x16_t m1, int8x16_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAXT.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaxq_p[_s16](int32_t add, int16x8_t m1, int16x8_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAXT.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavaxq_p[_s32](int32_t add, int32x4_t m1, int32x4_t m2, mve_pred16_t p)	add -> Rda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVAXT.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavxq[_s8](int8x16_t m1, int8x16_t m2)	m1 -> Qn m2 -> Qm	VMLADAVX.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavxq[_s16](int16x8_t m1, int16x8_t m2)	m1 -> Qn m2 -> Qm	VMLADAVX.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavxq[_s32](int32x4_t m1, int32x4_t m2)	m1 -> Qn m2 -> Qm	VMLADAVX.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavxq_p[_s8](int8x16_t m1, int8x16_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVXT.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavxq_p[_s16](int16x8_t m1, int16x8_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVXT.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmladavxq_p[_s32](int32x4_t m1, int32x4_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLADAVXT.S32 Rda,Qn,Qm	Rda -> result	MVE
int64_t [__arm_]vmlaldavaq[_s16](int64_t add, int16x8_t m1, int16x8_t m2)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm	VMLALDAVA.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavaq[_s32](int64_t add, int32x4_t m1, int32x4_t m2)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm	VMLALDAVA.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vmlaldavaq[_u16](uint64_t add, uint16x8_t m1, uint16x8_t m2)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm	VMLALDAVA.U16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vmlaldavaq[_u32](uint64_t add, uint32x4_t m1, uint32x4_t m2)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm	VMLALDAVA.U32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavaq_p[_s16](int64_t add, int16x8_t m1, int16x8_t m2, mve_pred16_t p)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVAT.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavaq_p[_s32](int64_t add, int32x4_t m1, int32x4_t m2, mve_pred16_t p)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVAT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vmlaldavaq_p[_u16](uint64_t add, uint16x8_t m1, uint16x8_t m2, mve_pred16_t p)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVAT.U16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vmlaldavaq_p[_u32](uint64_t add, uint32x4_t m1, uint32x4_t m2, mve_pred16_t p)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVAT.U32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavq[_s16](int16x8_t m1, int16x8_t m2)	m1 -> Qn m2 -> Qm	VMLALDAV.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavq[_s32](int32x4_t m1, int32x4_t m2)	m1 -> Qn m2 -> Qm	VMLALDAV.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint64_t [__arm_]vmlaldavq[_u16](uint16x8_t m1, uint16x8_t m2)	m1 -> Qn m2 -> Qm	VMLALDAV.U16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vmlaldavq[_u32](uint32x4_t m1, uint32x4_t m2)	m1 -> Qn m2 -> Qm	VMLALDAV.U32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavq_p[_s16](int16x8_t m1, int16x8_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVT.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavq_p[_s32](int32x4_t m1, int32x4_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vmlaldavq_p[_u16](uint16x8_t m1, uint16x8_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVT.U16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vmlaldavq_p[_u32](uint32x4_t m1, uint32x4_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVT.U32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavaxq[_s16](int64_t add, int16x8_t m1, int16x8_t m2)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm	VMLALDAVAX.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavaxq[_s32](int64_t add, int32x4_t m1, int32x4_t m2)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm	VMLALDAVAX.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavaxq_p[_s16](int64_t add, int16x8_t m1, int16x8_t m2, mve_pred16_t p)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVAXT.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavaxq_p[_s32](int64_t add, int32x4_t m1, int32x4_t m2, mve_pred16_t p)	add -> [RdaHi,RdaLo] m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVAXT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavxq[_s16](int16x8_t m1, int16x8_t m2)	m1 -> Qn m2 -> Qm	VMLALDAVX.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavxq[_s32](int32x4_t m1, int32x4_t m2)	m1 -> Qn m2 -> Qm	VMLALDAVX.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavxq_p[_s16](int16x8_t m1, int16x8_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVXT.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlaldavxq_p[_s32](int32x4_t m1, int32x4_t m2, mve_pred16_t p)	m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VMLALDAVXT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int8x16_t [__arm_]vmlaq[_n_s8](int8x16_t add, int8x16_t m1, int8_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VMLA.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vmlaq[_n_s16](int16x8_t add, int16x8_t m1, int16_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VMLA.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vmlaq[_n_s32](int32x4_t add, int32x4_t m1, int32_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VMLA.S32 Qda,Qn,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vmlaq[_n_u8](uint8x16_t add, uint8x16_t m1, uint8_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VMLA.U8 Qda,Qn,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vmlaq[_n_u16](uint16x8_t add, uint16x8_t m1, uint16_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VMLA.U16 Qda,Qn,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vmlaq[_n_u32](uint32x4_t add, uint32x4_t m1, uint32_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VMLA.U32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vmlaq_m[_n_s8](int8x16_t add, int8x16_t m1, int8_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VMLAT.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vmlaq_m[_n_s16](int16x8_t add, int16x8_t m1, int16_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VMLAT.S16 Qda,Qn,Rm	Qda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vmlaq_m[_n_s32](int32x4_t add, int32x4_t m1, int32_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VMLAT.S32 Qda,Qn,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vmlaq_m[_n_u8](uint8x16_t add, uint8x16_t m1, uint8_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VMLAT.U8 Qda,Qn,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vmlaq_m[_n_u16](uint16x8_t add, uint16x8_t m1, uint16_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VMLAT.U16 Qda,Qn,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vmlaq_m[_n_u32](uint32x4_t add, uint32x4_t m1, uint32_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VMLAT.U32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vmlasq[_n_s8](int8x16_t m1, int8x16_t m2, int8_t add)	m1 -> Qda m2 -> Qn add -> Rm	VMLAS.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vmlasq[_n_s16](int16x8_t m1, int16x8_t m2, int16_t add)	m1 -> Qda m2 -> Qn add -> Rm	VMLAS.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vmlasq[_n_s32](int32x4_t m1, int32x4_t m2, int32_t add)	m1 -> Qda m2 -> Qn add -> Rm	VMLAS.S32 Qda,Qn,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vmlasq[_n_u8](uint8x16_t m1, uint8x16_t m2, uint8_t add)	m1 -> Qda m2 -> Qn add -> Rm	VMLAS.U8 Qda,Qn,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vmlasq[_n_u16](uint16x8_t m1, uint16x8_t m2, uint16_t add)	m1 -> Qda m2 -> Qn add -> Rm	VMLAS.U16 Qda,Qn,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vmlasq[_n_u32](uint32x4_t m1, uint32x4_t m2, uint32_t add)	m1 -> Qda m2 -> Qn add -> Rm	VMLAS.U32 Qda,Qn,Rm	Qda -> result	MVE
int8x16_t [__arm_]vmlasq_m[_n_s8](int8x16_t m1, int8x16_t m2, int8_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VMLAST.S8 Qda,Qn,Rm	Qda -> result	MVE
int16x8_t [__arm_]vmlasq_m[_n_s16](int16x8_t m1, int16x8_t m2, int16_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VMLAST.S16 Qda,Qn,Rm	Qda -> result	MVE
int32x4_t [__arm_]vmlasq_m[_n_s32](int32x4_t m1, int32x4_t m2, int32_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VMLAST.S32 Qda,Qn,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vmlasq_m[_n_u8](uint8x16_t m1, uint8x16_t m2, uint8_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VMLAST.U8 Qda,Qn,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vmlasq_m[_n_u16](uint16x8_t m1, uint16x8_t m2, uint16_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VMLAST.U16 Qda,Qn,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vmlasq_m[_n_u32](uint32x4_t m1, uint32x4_t m2, uint32_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VMLAST.U32 Qda,Qn,Rm	Qda -> result	MVE
int32_t [__arm_]vmlsdavaq[_s8](int32_t a, int8x16_t b, int8x16_t c)	a -> Rda b -> Qn c -> Qm	VMLSDAVA.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaq[_s16](int32_t a, int16x8_t b, int16x8_t c)	a -> Rda b -> Qn c -> Qm	VMLSDAVA.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaq[_s32](int32_t a, int32x4_t b, int32x4_t c)	a -> Rda b -> Qn c -> Qm	VMLSDAVA.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaq_p[_s8](int32_t a, int8x16_t b, int8x16_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVAT.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaq_p[_s16](int32_t a, int16x8_t b, int16x8_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVAT.S16 Rda,Qn,Qm	Rda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32_t [__arm_]vmlsdavaq_p[_s32](int32_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVAT.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VMLSDAV.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMLSDAV.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMLSDAV.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavq_p[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVT.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavq_p[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVT.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavq_p[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVT.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaxq[_s8](int32_t a, int8x16_t b, int8x16_t c)	a -> Rda b -> Qn c -> Qm	VMLSDAVAX.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaxq[_s16](int32_t a, int16x8_t b, int16x8_t c)	a -> Rda b -> Qn c -> Qm	VMLSDAVAX.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaxq[_s32](int32_t a, int32x4_t b, int32x4_t c)	a -> Rda b -> Qn c -> Qm	VMLSDAVAX.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaxq_p[_s8](int32_t a, int8x16_t b, int8x16_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVAXT.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaxq_p[_s16](int32_t a, int16x8_t b, int16x8_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVAXT.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavaxq_p[_s32](int32_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> Rda b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVAXT.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavxq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VMLSDAVX.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavxq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMLSDAVX.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavxq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMLSDAVX.S32 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavxq_p[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVXT.S8 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavxq_p[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVXT.S16 Rda,Qn,Qm	Rda -> result	MVE
int32_t [__arm_]vmlsdavxq_p[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSDAVXT.S32 Rda,Qn,Qm	Rda -> result	MVE
int64_t [__arm_]vmlsldavaq[_s16](int64_t a, int16x8_t b, int16x8_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VMLSLDAVA.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavaq[_s32](int64_t a, int32x4_t b, int32x4_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VMLSLDAVA.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavaq_p[_s16](int64_t a, int16x8_t b, int16x8_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSLDAVAT.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavaq_p[_s32](int64_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSLDAVAT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMLSLDAV.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMLSLDAV.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int64_t [__arm_]vmlsldavq_p[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSLDAVT.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavq_p[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSLDAVT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavaxq[_s16](int64_t a, int16x8_t b, int16x8_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VMLSLDAVAX.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavaxq[_s32](int64_t a, int32x4_t b, int32x4_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VMLSLDAVAX.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavaxq_p[_s16](int64_t a, int16x8_t b, int16x8_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSLDAVAXT.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavaxq_p[_s32](int64_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VMLSLDAVAXT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavxq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VMLSLDAVX.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavxq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VMLSLDAVX.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavxq_p[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSLDAVXT.S16 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vmlsldavxq_p[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VMLSLDAVXT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int8x16_t [__arm_]vhaddq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VHADD.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vhaddq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VHADD.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vhaddq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VHADD.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vhaddq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VHADD.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vhaddq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VHADD.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vhaddq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VHADD.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vhaddq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VHADD.S8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vhaddq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VHADD.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vhaddq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VHADD.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vhaddq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VHADD.U8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vhaddq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VHADD.U16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vhaddq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VHADD.U32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vhaddq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vhaddq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vhaddq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.S32 Qd,Qn,Rm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint8x16_t [__arm_]vhaddq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, uint8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vhaddq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, uint16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vhaddq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, uint32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vhaddq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vhaddq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vhaddq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vhaddq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vhaddq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vhaddq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vhaddq_x[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vhaddq_x[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vhaddq_x[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vhaddq_x[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vhaddq_x[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vhaddq_x[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHADDT.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vhaddq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vhaddq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vhaddq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vhaddq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vhaddq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vhaddq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHADDT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vhcaddq_rot90[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VHCADD.S8 Qd,Qn,Qm,#90	Qd -> result	MVE
int16x8_t [__arm_]vhcaddq_rot90[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VHCADD.S16 Qd,Qn,Qm,#90	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vhcaddq_rot90[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VHCADD.S32 Qd,Qn,Qm,#90	Qd -> result	MVE
int8x16_t [__arm_]vhcaddq_rot90_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S8 Qd,Qn,Qm,#90	Qd -> result	MVE
int16x8_t [__arm_]vhcaddq_rot90_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S16 Qd,Qn,Qm,#90	Qd -> result	MVE
int32x4_t [__arm_]vhcaddq_rot90_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S32 Qd,Qn,Qm,#90	Qd -> result	MVE
int8x16_t [__arm_]vhcaddq_rot90_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S8 Qd,Qn,Qm,#90	Qd -> result	MVE
int16x8_t [__arm_]vhcaddq_rot90_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S16 Qd,Qn,Qm,#90	Qd -> result	MVE
int32x4_t [__arm_]vhcaddq_rot90_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S32 Qd,Qn,Qm,#90	Qd -> result	MVE
int8x16_t [__arm_]vhcaddq_rot270[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VHCADD.S8 Qd,Qn,Qm,#270	Qd -> result	MVE
int16x8_t [__arm_]vhcaddq_rot270[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VHCADD.S16 Qd,Qn,Qm,#270	Qd -> result	MVE
int32x4_t [__arm_]vhcaddq_rot270[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VHCADD.S32 Qd,Qn,Qm,#270	Qd -> result	MVE
int8x16_t [__arm_]vhcaddq_rot270_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S8 Qd,Qn,Qm,#270	Qd -> result	MVE
int16x8_t [__arm_]vhcaddq_rot270_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S16 Qd,Qn,Qm,#270	Qd -> result	MVE
int32x4_t [__arm_]vhcaddq_rot270_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S32 Qd,Qn,Qm,#270	Qd -> result	MVE
int8x16_t [__arm_]vhcaddq_rot270_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S8 Qd,Qn,Qm,#270	Qd -> result	MVE
int16x8_t [__arm_]vhcaddq_rot270_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S16 Qd,Qn,Qm,#270	Qd -> result	MVE
int32x4_t [__arm_]vhcaddq_rot270_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHCADDT.S32 Qd,Qn,Qm,#270	Qd -> result	MVE
int8x16_t [__arm_]vhsubq[_n_s8](int8x16_t a, int8_t b)	a -> Qn b -> Rm	VHSUB.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vhsubq[_n_s16](int16x8_t a, int16_t b)	a -> Qn b -> Rm	VHSUB.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vhsubq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VHSUB.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vhsubq[_n_u8](uint8x16_t a, uint8_t b)	a -> Qn b -> Rm	VHSUB.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vhsubq[_n_u16](uint16x8_t a, uint16_t b)	a -> Qn b -> Rm	VHSUB.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vhsubq[_n_u32](uint32x4_t a, uint32_t b)	a -> Qn b -> Rm	VHSUB.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vhsubq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VHSUB.S8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vhsubq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VHSUB.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vhsubq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VHSUB.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vhsubq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VHSUB.U8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vhsubq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VHSUB.U16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vhsubq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VHSUB.U32 Qd,Qn,Qm	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vhsubq_m[_n_s8](int8x16_t inactive, int8x16_t a, int8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vhsubq_m[_n_s16](int16x8_t inactive, int16x8_t a, int16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vhsubq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vhsubq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, uint8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vhsubq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, uint16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vhsubq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, uint32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vhsubq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vhsubq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vhsubq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vhsubq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vhsubq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vhsubq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vhsubq_x[_n_s8](int8x16_t a, int8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.S8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vhsubq_x[_n_s16](int16x8_t a, int16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.S16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vhsubq_x[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.S32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vhsubq_x[_n_u8](uint8x16_t a, uint8_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.U8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vhsubq_x[_n_u16](uint16x8_t a, uint16_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.U16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vhsubq_x[_n_u32](uint32x4_t a, uint32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VHSUBT.U32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vhsubq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vhsubq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vhsubq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.S32 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint8x16_t [__arm_]vhsubq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vhsubq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vhsubq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VHSUBT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrhaddq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VRHADD.S8 Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vrhaddq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VRHADD.S16 Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vrhaddq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VRHADD.S32 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vrhaddq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VRHADD.U8 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vrhaddq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VRHADD.U16 Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vrhaddq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VRHADD.U32 Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vrhaddq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrhaddq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vrhaddq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrhaddq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vrhaddq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vrhaddq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrhaddq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrhaddq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vrhaddq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrhaddq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vrhaddq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vrhaddq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRHADDT.U32 Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vfmaq[_n_f16](float16x8_t add, float16x8_t m1, float16_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VFMA.F16 Qda,Qn,Rm	Qda -> result	MVE/NEON
float32x4_t [__arm_]vfmaq[_n_f32](float32x4_t add, float32x4_t m1, float32_t m2)	add -> Qda m1 -> Qn m2 -> Rm	VFMA.F32 Qda,Qn,Rm	Qda -> result	MVE/NEON
float16x8_t [__arm_]vfmaq_m[_n_f16](float16x8_t add, float16x8_t m1, float16_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VFMAT.F16 Qda,Qn,Rm	Qda -> result	MVE
float32x4_t [__arm_]vfmaq_m[_n_f32](float32x4_t add, float32x4_t m1, float32_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Rm p -> Rp	VMSR P0,Rp VPST VFMAT.F32 Qda,Qn,Rm	Qda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vfmaq[_f16](float16x8_t add, float16x8_t m1, float16x8_t m2)	add -> Qda m1 -> Qn m2 -> Qm	VFMA.F16 Qda,Qn,Qm	Qda -> result	MVE/NEON
float32x4_t [__arm_]vfmaq[_f32](float32x4_t add, float32x4_t m1, float32x4_t m2)	add -> Qda m1 -> Qn m2 -> Qm	VFMA.F32 Qda,Qn,Qm	Qda -> result	MVE/NEON
float16x8_t [__arm_]vfmaq_m[_f16](float16x8_t add, float16x8_t m1, float16x8_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VFMAT.F16 Qda,Qn,Qm	Qda -> result	MVE
float32x4_t [__arm_]vfmaq_m[_f32](float32x4_t add, float32x4_t m1, float32x4_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VFMAT.F32 Qda,Qn,Qm	Qda -> result	MVE
float16x8_t [__arm_]vfmasq[_n_f16](float16x8_t m1, float16x8_t m2, float16_t add)	m1 -> Qda m2 -> Qn add -> Rm	VFMAS.F16 Qda,Qn,Rm	Qda -> result	MVE
float32x4_t [__arm_]vfmasq[_n_f32](float32x4_t m1, float32x4_t m2, float32_t add)	m1 -> Qda m2 -> Qn add -> Rm	VFMAS.F32 Qda,Qn,Rm	Qda -> result	MVE
float16x8_t [__arm_]vfmasq_m[_n_f16](float16x8_t m1, float16x8_t m2, float16_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VFMAST.F16 Qda,Qn,Rm	Qda -> result	MVE
float32x4_t [__arm_]vfmasq_m[_n_f32](float32x4_t m1, float32x4_t m2, float32_t add, mve_pred16_t p)	m1 -> Qda m2 -> Qn add -> Rm p -> Rp	VMSR P0,Rp VPST VFMAST.F32 Qda,Qn,Rm	Qda -> result	MVE
float16x8_t [__arm_]vfmsq[_f16](float16x8_t add, float16x8_t m1, float16x8_t m2)	add -> Qda m1 -> Qn m2 -> Qm	VFMS.F16 Qda,Qn,Qm	Qda -> result	MVE/NEON
float32x4_t [__arm_]vfmsq[_f32](float32x4_t add, float32x4_t m1, float32x4_t m2)	add -> Qda m1 -> Qn m2 -> Qm	VFMS.F32 Qda,Qn,Qm	Qda -> result	MVE/NEON
float16x8_t [__arm_]vfmsq_m[_f16](float16x8_t add, float16x8_t m1, float16x8_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VFMST.F16 Qda,Qn,Qm	Qda -> result	MVE
float32x4_t [__arm_]vfmsq_m[_f32](float32x4_t add, float32x4_t m1, float32x4_t m2, mve_pred16_t p)	add -> Qda m1 -> Qn m2 -> Qm p -> Rp	VMSR P0,Rp VPST VFMST.F32 Qda,Qn,Qm	Qda -> result	MVE
int64_t [__arm_]vrmlaldavhaq[_s32](int64_t a, int32x4_t b, int32x4_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VRMLALDAVHA.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vrmlaldavhaq[_u32](uint64_t a, uint32x4_t b, uint32x4_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VRMLALDAVHA.U32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlaldavhaq_p[_s32](int64_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VRMLALDAVHAT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vrmlaldavhaq_p[_u32](uint64_t a, uint32x4_t b, uint32x4_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VRMLALDAVHAT.U32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlaldavhq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VRMLALDAVH.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vrmlaldavhq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VRMLALDAVH.U32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlaldavhq_p[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMLALDAVHT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]vrmlaldavhq_p[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMLALDAVHT.U32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlaldavhaxq[_s32](int64_t a, int32x4_t b, int32x4_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VRMLALDAVHAX.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int64_t [__arm_]vrmlaldavhaxq_p[_s32](int64_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VRMLALDAVHAXT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlaldavhxq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VRMLALDAVHX.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlaldavhxq_p[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMLALDAVHXT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlsldavhaq[_s32](int64_t a, int32x4_t b, int32x4_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VRMLSLDAVHA.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlsldavhaq_p[_s32](int64_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VRMLSLDAVHAT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlsldavhq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VRMLSLDAVH.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlsldavhq_p[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMLSLDAVHT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlsldavhaxq[_s32](int64_t a, int32x4_t b, int32x4_t c)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm	VRMLSLDAVHAX.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlsldavhaxq_p[_s32](int64_t a, int32x4_t b, int32x4_t c, mve_pred16_t p)	a -> [RdaHi,RdaLo] b -> Qn c -> Qm p -> Rp	VMSR P0,Rp VPST VRMLSLDAVHAXT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlsldavhxq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VRMLSLDAVHX.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]vrmlsldavhxq_p[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMLSLDAVHXT.S32 RdaLo,RdaHi,Qn,Qm	[RdaHi,RdaLo] -> result	MVE
int8x16_t [__arm_]vrmulhq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VRMULH.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrmulhq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VRMULH.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vrmulhq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VRMULH.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrmulhq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VRMULH.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vrmulhq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VRMULH.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vrmulhq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VRMULH.U32 Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrmulhq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrmulhq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vrmulhq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrmulhq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vrmulhq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vrmulhq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.U32 Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vrmulhq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.S8 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrmulhq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.S16 Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vrmulhq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.S32 Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrmulhq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.U8 Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vrmulhq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.U16 Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vrmulhq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VRMULHT.U32 Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtaq_s16_f16(float16x8_t a)	a -> Qm	VCVTA.S16.F16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vcvtaq_s32_f32(float32x4_t a)	a -> Qm	VCVTA.S32.F32 Qd,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vcvtaq_u16_f16(float16x8_t a)	a -> Qm	VCVTA.U16.F16 Qd,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vcvtaq_u32_f32(float32x4_t a)	a -> Qm	VCVTA.U32.F32 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vcvtaq_m[_s16_f16](int16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTAT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtaq_m[_s32_f32](int32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTAT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtaq_m[_u16_f16](uint16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTAT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtaq_m[_u32_f32](uint32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTAT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtaq_x_s16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTAT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtaq_x_s32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTAT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtaq_x_u16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTAT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtaq_x_u32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTAT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtnq_s16_f16(float16x8_t a)	a -> Qm	VCVTN.S16.F16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vcvtnq_s32_f32(float32x4_t a)	a -> Qm	VCVTN.S32.F32 Qd,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vcvtnq_u16_f16(float16x8_t a)	a -> Qm	VCVTN.U16.F16 Qd,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vcvtnq_u32_f32(float32x4_t a)	a -> Qm	VCVTN.U32.F32 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vcvtnq_m[_s16_f16](int16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTNT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtnq_m[_s32_f32](int32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTNT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtnq_m[_u16_f16](uint16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTNT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtnq_m[_u32_f32](uint32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTNT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtnq_x_s16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTNT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtnq_x_s32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTNT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtnq_x_u16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTNT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtnq_x_u32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTNT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtpq_s16_f16(float16x8_t a)	a -> Qm	VCVTP.S16.F16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vcvtpq_s32_f32(float32x4_t a)	a -> Qm	VCVTP.S32.F32 Qd,Qm	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vcvtpq_u16_f16(float16x8_t a)	a -> Qm	VCVTP.U16.F16 Qd,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vcvtpq_u32_f32(float32x4_t a)	a -> Qm	VCVTP.U32.F32 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vcvtpq_m[_s16_f16](int16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTPT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtpq_m[_s32_f32](int32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTPT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtpq_m[_u16_f16](uint16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTPT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtpq_m[_u32_f32](uint32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTPT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtpq_x_s16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTPT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtpq_x_s32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTPT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtpq_x_u16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTPT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtpq_x_u32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTPT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtmq_s16_f16(float16x8_t a)	a -> Qm	VCVTM.S16.F16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vcvtmq_s32_f32(float32x4_t a)	a -> Qm	VCVTM.S32.F32 Qd,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vcvtmq_u16_f16(float16x8_t a)	a -> Qm	VCVTM.U16.F16 Qd,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vcvtmq_u32_f32(float32x4_t a)	a -> Qm	VCVTM.U32.F32 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vcvtmq_m[_s16_f16](int16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTMT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtmq_m[_s32_f32](int32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTMT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtmq_m[_u16_f16](uint16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTMT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtmq_m[_u32_f32](uint32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTMT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtmq_x_s16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTMT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtmq_x_s32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTMT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtmq_x_u16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTMT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtmq_x_u32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTMT.U32.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvtbq_f16_f32(float16x8_t a, float32x4_t b)	a -> Qd b -> Qm	VCVTB.F16.F32 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvtbq_f32_f16(float16x8_t a)	a -> Qm	VCVTB.F32.F16 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvtbq_m_f16_f32(float16x8_t a, float32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VCVTBT.F16.F32 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvtbq_m_f32_f16(float32x4_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTBT.F32.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvtbq_x_f32_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTBT.F32.F16 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvttq_f16_f32(float16x8_t a, float32x4_t b)	a -> Qd b -> Qm	VCVTT.F16.F32 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvttq_f32_f16(float16x8_t a)	a -> Qm	VCVTT.F32.F16 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvttq_m_f16_f32(float16x8_t a, float32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VCVTTT.F16.F32 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvttq_m_f32_f16(float32x4_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTTT.F32.F16 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float32x4_t [__arm_]vcvttq_x_f32_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTTT.F32.F16 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvtq[_f16_s16](int16x8_t a)	a -> Qm	VCVT.F16.S16 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vcvtq[_f16_u16](uint16x8_t a)	a -> Qm	VCVT.F16.U16 Qd,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vcvtq[_f32_s32](int32x4_t a)	a -> Qm	VCVT.F32.S32 Qd,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vcvtq[_f32_u32](uint32x4_t a)	a -> Qm	VCVT.F32.U32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vcvtq_m[_f16_s16](float16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.F16.S16 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvtq_m[_f16_u16](float16x8_t inactive, uint16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.F16.U16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvtq_m[_f32_s32](float32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.F32.S32 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvtq_m[_f32_u32](float32x4_t inactive, uint32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.F32.U32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvtq_x[_f16_u16](uint16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.F16.U16 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvtq_x[_f16_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.F16.S16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvtq_x[_f32_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.F32.S32 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vcvtq_x[_f32_u32](uint32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.F32.U32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vcvtq_n[_f16_s16](int16x8_t a, const int imm6)	a -> Qm 1 <= imm6 <= 16	VCVT.F16.S16 Qd,Qm,imm6	Qd -> result	MVE/NEON
float16x8_t [__arm_]vcvtq_n[_f16_u16](uint16x8_t a, const int imm6)	a -> Qm 1 <= imm6 <= 16	VCVT.F16.U16 Qd,Qm,imm6	Qd -> result	MVE/NEON
float32x4_t [__arm_]vcvtq_n[_f32_s32](int32x4_t a, const int imm6)	a -> Qm 1 <= imm6 <= 32	VCVT.F32.S32 Qd,Qm,imm6	Qd -> result	MVE/NEON
float32x4_t [__arm_]vcvtq_n[_f32_u32](uint32x4_t a, const int imm6)	a -> Qm 1 <= imm6 <= 32	VCVT.F32.U32 Qd,Qm,imm6	Qd -> result	MVE/NEON
float16x8_t [__arm_]vcvtq_m_n[_f16_s16](float16x8_t inactive, int16x8_t a, const int imm6, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm6 <= 16 p -> Rp	VMSR P0,Rp VPST VCVTT.F16.S16 Qd,Qm,imm6	Qd -> result	MVE
float16x8_t [__arm_]vcvtq_m_n[_f16_u16](float16x8_t inactive, uint16x8_t a, const int imm6, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm6 <= 16 p -> Rp	VMSR P0,Rp VPST VCVTT.F16.U16 Qd,Qm,imm6	Qd -> result	MVE
float32x4_t [__arm_]vcvtq_m_n[_f32_s32](float32x4_t inactive, int32x4_t a, const int imm6, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm6 <= 32 p -> Rp	VMSR P0,Rp VPST VCVTT.F32.S32 Qd,Qm,imm6	Qd -> result	MVE
float32x4_t [__arm_]vcvtq_m_n[_f32_u32](float32x4_t inactive, uint32x4_t a, const int imm6, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm6 <= 32 p -> Rp	VMSR P0,Rp VPST VCVTT.F32.U32 Qd,Qm,imm6	Qd -> result	MVE
float16x8_t [__arm_]vcvtq_x_n[_f16_s16](int16x8_t a, const int imm6, mve_pred16_t p)	a -> Qm 1 <= imm6 <= 16 p -> Rp	VMSR P0,Rp VPST VCVTT.F16.S16 Qd,Qm,imm6	Qd -> result	MVE
float16x8_t [__arm_]vcvtq_x_n[_f16_u16](uint16x8_t a, const int imm6, mve_pred16_t p)	a -> Qm 1 <= imm6 <= 16 p -> Rp	VMSR P0,Rp VPST VCVTT.F16.U16 Qd,Qm,imm6	Qd -> result	MVE
float32x4_t [__arm_]vcvtq_x_n[_f32_s32](int32x4_t a, const int imm6, mve_pred16_t p)	a -> Qm 1 <= imm6 <= 32 p -> Rp	VMSR P0,Rp VPST VCVTT.F32.S32 Qd,Qm,imm6	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float32x4_t [__arm_]vcvtq_x_n[_f32_u32](uint32x4_t a, const int imm6, mve_pred16_t p)	a -> Qm 1 <= imm6 <= 32 p -> Rp	VMSR P0,Rp VPST VCVTT.F32.U32 Qd,Qm,imm6	Qd -> result	MVE
int16x8_t [__arm_]vcvtq_s16_f16(float16x8_t a)	a -> Qm	VCVT.S16.F16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vcvtq_s32_f32(float32x4_t a)	a -> Qm	VCVT.S32.F32 Qd,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vcvtq_u16_f16(float16x8_t a)	a -> Qm	VCVT.U16.F16 Qd,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vcvtq_u32_f32(float32x4_t a)	a -> Qm	VCVT.U32.F32 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vcvtq_m[_s16_f16](int16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtq_m[_s32_f32](int32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtq_m[_u16_f16](uint16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtq_m[_u32_f32](uint32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtq_x_s16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.S16.F16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vcvtq_x_s32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.S32.F32 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vcvtq_x_u16_f16(float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.U16.F16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vcvtq_x_u32_f32(float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VCVTT.U32.F32 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vcvtq_n_s16_f16(float16x8_t a, const int imm6)	a -> Qm 1 <= imm6 <= 16	VCVT.S16.F16 Qd,Qm,imm6	Qd -> result	MVE/NEON
int32x4_t [__arm_]vcvtq_n_s32_f32(float32x4_t a, const int imm6)	a -> Qm 1 <= imm6 <= 32	VCVT.S32.F32 Qd,Qm,imm6	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vcvtq_n_u16_f16(float16x8_t a, const int imm6)	a -> Qm 1 <= imm6 <= 16	VCVT.U16.F16 Qd,Qm,imm6	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vcvtq_n_u32_f32(float32x4_t a, const int imm6)	a -> Qm 1 <= imm6 <= 32	VCVT.U32.F32 Qd,Qm,imm6	Qd -> result	MVE/NEON
int16x8_t [__arm_]vcvtq_m_n[_s16_f16](int16x8_t inactive, float16x8_t a, const int imm6, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm6 <= 16 p -> Rp	VMSR P0,Rp VPST VCVTT.S16.F16 Qd,Qm,imm6	Qd -> result	MVE
int32x4_t [__arm_]vcvtq_m_n[_s32_f32](int32x4_t inactive, float32x4_t a, const int imm6, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm6 <= 32 p -> Rp	VMSR P0,Rp VPST VCVTT.S32.F32 Qd,Qm,imm6	Qd -> result	MVE
uint16x8_t [__arm_]vcvtq_m_n[_u16_f16](uint16x8_t inactive, float16x8_t a, const int imm6, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm6 <= 16 p -> Rp	VMSR P0,Rp VPST VCVTT.U16.F16 Qd,Qm,imm6	Qd -> result	MVE
uint32x4_t [__arm_]vcvtq_m_n[_u32_f32](uint32x4_t inactive, float32x4_t a, const int imm6, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm6 <= 32 p -> Rp	VMSR P0,Rp VPST VCVTT.U32.F32 Qd,Qm,imm6	Qd -> result	MVE
int16x8_t [__arm_]vcvtq_x_n_s16_f16(float16x8_t a, const int imm6, mve_pred16_t p)	a -> Qm 1 <= imm6 <= 16 p -> Rp	VMSR P0,Rp VPST VCVTT.S16.F16 Qd,Qm,imm6	Qd -> result	MVE
int32x4_t [__arm_]vcvtq_x_n_s32_f32(float32x4_t a, const int imm6, mve_pred16_t p)	a -> Qm 1 <= imm6 <= 32 p -> Rp	VMSR P0,Rp VPST VCVTT.S32.F32 Qd,Qm,imm6	Qd -> result	MVE
uint16x8_t [__arm_]vcvtq_x_n_u16_f16(float16x8_t a, const int imm6, mve_pred16_t p)	a -> Qm 1 <= imm6 <= 16 p -> Rp	VMSR P0,Rp VPST VCVTT.U16.F16 Qd,Qm,imm6	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vcvtq_x_n_u32_f32(float32x4_t a, const int imm6, mve_pred16_t p)	a -> Qm 1 <= imm6 <= 32 p -> Rp	VMSR P0,Rp VPST VCVTT.U32.F32 Qd,Qm,imm6	Qd -> result	MVE
float16x8_t [__arm_]vrndq[_f16](float16x8_t a)	a -> Qm	VRINTZ.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndq[_f32](float32x4_t a)	a -> Qm	VRINTZ.F32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vrndq_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTZT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndq_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTZT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndq_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTZT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndq_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTZT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndnq[_f16](float16x8_t a)	a -> Qm	VRINTN.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndnq[_f32](float32x4_t a)	a -> Qm	VRINTN.F32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vrndnq_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTNT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndnq_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTNT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndnq_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTNT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndnq_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTNT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndmq[_f16](float16x8_t a)	a -> Qm	VRINTM.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndmq[_f32](float32x4_t a)	a -> Qm	VRINTM.F32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vrndmq_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTMT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndmq_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTMT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndmq_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTMT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndmq_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTMT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndpq[_f16](float16x8_t a)	a -> Qm	VRINTP.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndpq[_f32](float32x4_t a)	a -> Qm	VRINTP.F32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vrndpq_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTPT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndpq_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTPT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndpq_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTPT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndpq_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTPT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndaq[_f16](float16x8_t a)	a -> Qm	VRINTA.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndaq[_f32](float32x4_t a)	a -> Qm	VRINTA.F32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vrndaq_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTAT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndaq_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTAT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndaq_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTAT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndaq_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTAT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndxq[_f16](float16x8_t a)	a -> Qm	VRINTX.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndxq[_f32](float32x4_t a)	a -> Qm	VRINTX.F32 Qd,Qm	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vrndxq_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTXT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndxq_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTXT.F32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrndxq_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTXT.F16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrndxq_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VRINTXT.F32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vandq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VAND Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vandq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VAND Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vandq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VAND Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vandq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VAND Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vandq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VAND Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vandq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VAND Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vandq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VAND Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vandq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VAND Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vandq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vandq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vandq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vandq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vandq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vandq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vandq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vandq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vandq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vandq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vandq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vandq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vandq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vandq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vandq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vandq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VANDT Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vbicq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VBIC Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vbicq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VBIC Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vbicq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VBIC Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vbicq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VBIC Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vbicq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VBIC Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vbicq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VBIC Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vbicq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VBIC Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vbicq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VBIC Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vbicq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vbicq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vbicq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vbicq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vbicq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vbicq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vbicq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vbicq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vbicq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vbicq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vbicq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vbicq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vbicq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vbicq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vbicq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vbicq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VBICT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vbicq[_n_s16](int16x8_t a, const int16_t imm)	a -> Qda imm in AdvSIMDExpandImm	VBIC.I16 Qda,#imm	Qda -> result	MVE
int32x4_t [__arm_]vbicq[_n_s32](int32x4_t a, const int32_t imm)	a -> Qda imm in AdvSIMDExpandImm	VBIC.I32 Qda,#imm	Qda -> result	MVE
uint16x8_t [__arm_]vbicq[_n_u16](uint16x8_t a, const uint16_t imm)	a -> Qda imm in AdvSIMDExpandImm	VBIC.I16 Qda,#imm	Qda -> result	MVE
uint32x4_t [__arm_]vbicq[_n_u32](uint32x4_t a, const uint32_t imm)	a -> Qda imm in AdvSIMDExpandImm	VBIC.I32 Qda,#imm	Qda -> result	MVE
int16x8_t [__arm_]vbicq_m_n[_s16](int16x8_t a, const int16_t imm, mve_pred16_t p)	a -> Qda imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VBICT.I16 Qda,#imm	Qda -> result	MVE
int32x4_t [__arm_]vbicq_m_n[_s32](int32x4_t a, const int32_t imm, mve_pred16_t p)	a -> Qda imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VBICT.I32 Qda,#imm	Qda -> result	MVE
uint16x8_t [__arm_]vbicq_m_n[_u16](uint16x8_t a, const uint16_t imm, mve_pred16_t p)	a -> Qda imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VBICT.I16 Qda,#imm	Qda -> result	MVE
uint32x4_t [__arm_]vbicq_m_n[_u32](uint32x4_t a, const uint32_t imm, mve_pred16_t p)	a -> Qda imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VBICT.I32 Qda,#imm	Qda -> result	MVE
int8x16_t [__arm_]vbrsrq[_n_s8](int8x16_t a, int32_t b)	a -> Qn b -> Rm	VBRSR.8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vbrsrq[_n_s16](int16x8_t a, int32_t b)	a -> Qn b -> Rm	VBRSR.16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vbrsrq[_n_s32](int32x4_t a, int32_t b)	a -> Qn b -> Rm	VBRSR.32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vbrsrq[_n_u8](uint8x16_t a, int32_t b)	a -> Qn b -> Rm	VBRSR.8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vbrsrq[_n_u16](uint16x8_t a, int32_t b)	a -> Qn b -> Rm	VBRSR.16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vbrsrq[_n_u32](uint32x4_t a, int32_t b)	a -> Qn b -> Rm	VBRSR.32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vbrsrq[_n_f16](float16x8_t a, int32_t b)	a -> Qn b -> Rm	VBRSR.16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vbrsrq[_n_f32](float32x4_t a, int32_t b)	a -> Qn b -> Rm	VBRSR.32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vbrsrq_m[_n_s8](int8x16_t inactive, int8x16_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vbrsrq_m[_n_s16](int16x8_t inactive, int16x8_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vbrsrq_m[_n_s32](int32x4_t inactive, int32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vbrsrq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.8 Qd,Qn,Rm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vbrsrq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vbrsrq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vbrsrq_m[_n_f16](float16x8_t inactive, float16x8_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vbrsrq_m[_n_f32](float32x4_t inactive, float32x4_t a, int32_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]vbrsrq_x[_n_s8](int8x16_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.8 Qd,Qn,Rm	Qd -> result	MVE
int16x8_t [__arm_]vbrsrq_x[_n_s16](int16x8_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.16 Qd,Qn,Rm	Qd -> result	MVE
int32x4_t [__arm_]vbrsrq_x[_n_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.32 Qd,Qn,Rm	Qd -> result	MVE
uint8x16_t [__arm_]vbrsrq_x[_n_u8](uint8x16_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.8 Qd,Qn,Rm	Qd -> result	MVE
uint16x8_t [__arm_]vbrsrq_x[_n_u16](uint16x8_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.16 Qd,Qn,Rm	Qd -> result	MVE
uint32x4_t [__arm_]vbrsrq_x[_n_u32](uint32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.32 Qd,Qn,Rm	Qd -> result	MVE
float16x8_t [__arm_]vbrsrq_x[_n_f16](float16x8_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.16 Qd,Qn,Rm	Qd -> result	MVE
float32x4_t [__arm_]vbrsrq_x[_n_f32](float32x4_t a, int32_t b, mve_pred16_t p)	a -> Qn b -> Rm p -> Rp	VMSR P0,Rp VPST VBRSRT.32 Qd,Qn,Rm	Qd -> result	MVE
int8x16_t [__arm_]veorq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VEOR Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]veorq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VEOR Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]veorq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VEOR Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]veorq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VEOR Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]veorq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VEOR Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]veorq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VEOR Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]veorq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VEOR Qd,Qn,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]veorq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VEOR Qd,Qn,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]veorq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]veorq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]veorq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]veorq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]veorq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]veorq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]veorq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]veorq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]veorq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]veorq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]veorq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]veorq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]veorq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]veorq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]veorq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]veorq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VEORT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovlbq[_s8](int8x16_t a)	a -> Qm	VMOVLB.S8 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmovlbq[_s16](int16x8_t a)	a -> Qm	VMOVLB.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovlbq[_u8](uint8x16_t a)	a -> Qm	VMOVLB.U8 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmovlbq[_u16](uint16x8_t a)	a -> Qm	VMOVLB.U16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovlbq_m[_s8](int16x8_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLBT.S8 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmovlbq_m[_s16](int32x4_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLBT.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovlbq_m[_u8](uint16x8_t inactive, uint8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLBT.U8 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmovlbq_m[_u16](uint32x4_t inactive, uint16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLBT.U16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovlbq_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLBT.S8 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmovlbq_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLBT.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovlbq_x[_u8](uint8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLBT.U8 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmovlbq_x[_u16](uint16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLBT.U16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovltq[_s8](int8x16_t a)	a -> Qm	VMOVLT.S8 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmovltq[_s16](int16x8_t a)	a -> Qm	VMOVLT.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovltq[_u8](uint8x16_t a)	a -> Qm	VMOVLT.U8 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmovltq[_u16](uint16x8_t a)	a -> Qm	VMOVLT.U16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovltq_m[_s8](int16x8_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLTT.S8 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmovltq_m[_s16](int32x4_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLTT.S16 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vmovltq_m[_u8](uint16x8_t inactive, uint8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLTT.U8 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmovltq_m[_u16](uint32x4_t inactive, uint16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLTT.U16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovltq_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLTT.S8 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmovltq_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLTT.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovltq_x[_u8](uint8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLTT.U8 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmovltq_x[_u16](uint16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMOVLTT.U16 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmovnbq[_s16](int8x16_t a, int16x8_t b)	a -> Qd b -> Qm	VMOVNB.I16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovnbq[_s32](int16x8_t a, int32x4_t b)	a -> Qd b -> Qm	VMOVNB.I32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmovnbq[_u16](uint8x16_t a, uint16x8_t b)	a -> Qd b -> Qm	VMOVNB.I16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovnbq[_u32](uint16x8_t a, uint32x4_t b)	a -> Qd b -> Qm	VMOVNB.I32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmovnbq_m[_s16](int8x16_t a, int16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VMOVNBT.I16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovnbq_m[_s32](int16x8_t a, int32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VMOVNBT.I32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmovnbq_m[_u16](uint8x16_t a, uint16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VMOVNBT.I16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovnbq_m[_u32](uint16x8_t a, uint32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VMOVNBT.I32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmovntq[_s16](int8x16_t a, int16x8_t b)	a -> Qd b -> Qm	VMOVNT.I16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovntq[_s32](int16x8_t a, int32x4_t b)	a -> Qd b -> Qm	VMOVNT.I32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmovntq[_u16](uint8x16_t a, uint16x8_t b)	a -> Qd b -> Qm	VMOVNT.I16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovntq[_u32](uint16x8_t a, uint32x4_t b)	a -> Qd b -> Qm	VMOVNT.I32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmovntq_m[_s16](int8x16_t a, int16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VMOVNTT.I16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmovntq_m[_s32](int16x8_t a, int32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VMOVNTT.I32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmovntq_m[_u16](uint8x16_t a, uint16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VMOVNTT.I16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmovntq_m[_u32](uint16x8_t a, uint32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VMOVNTT.I32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmvnq[_s8](int8x16_t a)	a -> Qm	VMVN Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vmvnq[_s16](int16x8_t a)	a -> Qm	VMVN Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vmvnq[_s32](int32x4_t a)	a -> Qm	VMVN Qd,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vmvnq[_u8](uint8x16_t a)	a -> Qm	VMVN Qd,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vmvnq[_u16](uint16x8_t a)	a -> Qm	VMVN Qd,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vmvnq[_u32](uint32x4_t a)	a -> Qm	VMVN Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vmvnq_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmvnq_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmvnq_m[_s32](int32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmvnq_m[_u8](uint8x16_t inactive, uint8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vmvnq_m[_u16](uint16x8_t inactive, uint16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmvnq_m[_u32](uint32x4_t inactive, uint32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vmvnq_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmvnq_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vmvnq_x[_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vmvnq_x[_u8](uint8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vmvnq_x[_u16](uint16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vmvnq_x[_u32](uint32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VMVNT Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vmvnq_n_s16(const int16_t imm)	imm in AdvSIMDExpandImm	VMVN.I16 Qd,#imm	Qd -> result	MVE
int32x4_t [__arm_]vmvnq_n_s32(const int32_t imm)	imm in AdvSIMDExpandImm	VMVN.I32 Qd,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vmvnq_n_u16(const uint16_t imm)	imm in AdvSIMDExpandImm	VMVN.I16 Qd,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vmvnq_n_u32(const uint32_t imm)	imm in AdvSIMDExpandImm	VMVN.I32 Qd,#imm	Qd -> result	MVE
int16x8_t [__arm_]vmvnq_m[_n_s16](int16x8_t inactive, const int16_t imm, mve_pred16_t p)	inactive -> Qd imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VMVNT.I16 Qd,#imm	Qd -> result	MVE
int32x4_t [__arm_]vmvnq_m[_n_s32](int32x4_t inactive, const int32_t imm, mve_pred16_t p)	inactive -> Qd imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VMVNT.I32 Qd,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vmvnq_m[_n_u16](uint16x8_t inactive, const uint16_t imm, mve_pred16_t p)	inactive -> Qd imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VMVNT.I16 Qd,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vmvnq_m[_n_u32](uint32x4_t inactive, const uint32_t imm, mve_pred16_t p)	inactive -> Qd imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VMVNT.I32 Qd,#imm	Qd -> result	MVE
int16x8_t [__arm_]vmvnq_x_n_s16(const int16_t imm, mve_pred16_t p)	imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VMVNT.I16 Qd,#imm	Qd -> result	MVE
int32x4_t [__arm_]vmvnq_x_n_s32(const int32_t imm, mve_pred16_t p)	imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VMVNT.I32 Qd,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vmvnq_x_n_u16(const uint16_t imm, mve_pred16_t p)	imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VMVNT.I16 Qd,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vmvnq_x_n_u32(const uint32_t imm, mve_pred16_t p)	imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VMVNT.I32 Qd,#imm	Qd -> result	MVE
mve_pred16_t [__arm_]vpnot(mve_pred16_t a)	a -> Rp	VMSR P0,Rp VPNOT VMRS Rt,P0	Rt -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vpselq[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vpselq[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vpselq[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
int64x2_t [__arm_]vpselq[_s64](int64x2_t a, int64x2_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vpselq[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vpselq[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vpselq[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
uint64x2_t [__arm_]vpselq[_u64](uint64x2_t a, uint64x2_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vpselq[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vpselq[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPSEL Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vornq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VORN Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vornq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VORN Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vornq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VORN Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vornq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VORN Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vornq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VORN Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vornq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VORN Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vornq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VORN Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vornq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VORN Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vornq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vornq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vornq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vornq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vornq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vornq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vornq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vornq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vornq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vornq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vornq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vornq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vornq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vornq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vornq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vornq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORNT Qd,Qn,Qm	Qd -> result	MVE
float16x8_t [__arm_]vorrq[_f16](float16x8_t a, float16x8_t b)	a -> Qn b -> Qm	VORR Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vorrq[_f32](float32x4_t a, float32x4_t b)	a -> Qn b -> Qm	VORR Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vorrq[_s8](int8x16_t a, int8x16_t b)	a -> Qn b -> Qm	VORR Qd,Qn,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vorrq[_s16](int16x8_t a, int16x8_t b)	a -> Qn b -> Qm	VORR Qd,Qn,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vorrq[_s32](int32x4_t a, int32x4_t b)	a -> Qn b -> Qm	VORR Qd,Qn,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vorrq[_u8](uint8x16_t a, uint8x16_t b)	a -> Qn b -> Qm	VORR Qd,Qn,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vorrq[_u16](uint16x8_t a, uint16x8_t b)	a -> Qn b -> Qm	VORR Qd,Qn,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vorrq[_u32](uint32x4_t a, uint32x4_t b)	a -> Qn b -> Qm	VORR Qd,Qn,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vorrq_m[_f16](float16x8_t inactive, float16x8_t a, float16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vorrq_m[_f32](float32x4_t inactive, float32x4_t a, float32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vorrq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vorrq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vorrq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vorrq_m[_u8](uint8x16_t inactive, uint8x16_t a, uint8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vorrq_m[_u16](uint16x8_t inactive, uint16x8_t a, uint16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vorrq_m[_u32](uint32x4_t inactive, uint32x4_t a, uint32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
float16x8_t [__arm_]vorrq_x[_f16](float16x8_t a, float16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
float32x4_t [__arm_]vorrq_x[_f32](float32x4_t a, float32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
int8x16_t [__arm_]vorrq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vorrq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
int32x4_t [__arm_]vorrq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vorrq_x[_u8](uint8x16_t a, uint8x16_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vorrq_x[_u16](uint16x8_t a, uint16x8_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vorrq_x[_u32](uint32x4_t a, uint32x4_t b, mve_pred16_t p)	a -> Qn b -> Qm p -> Rp	VMSR P0,Rp VPST VORRT Qd,Qn,Qm	Qd -> result	MVE
int16x8_t [__arm_]vorrq[_n_s16](int16x8_t a, const int16_t imm)	a -> Qda imm in AdvSIMDExpandImm	VORR.I16 Qda,#imm	Qda -> result	MVE
int32x4_t [__arm_]vorrq[_n_s32](int32x4_t a, const int32_t imm)	a -> Qda imm in AdvSIMDExpandImm	VORR.I32 Qda,#imm	Qda -> result	MVE
uint16x8_t [__arm_]vorrq[_n_u16](uint16x8_t a, const uint16_t imm)	a -> Qda imm in AdvSIMDExpandImm	VORR.I16 Qda,#imm	Qda -> result	MVE
uint32x4_t [__arm_]vorrq[_n_u32](uint32x4_t a, const uint32_t imm)	a -> Qda imm in AdvSIMDExpandImm	VORR.I32 Qda,#imm	Qda -> result	MVE
int16x8_t [__arm_]vorrq_m_n[_s16](int16x8_t a, const int16_t imm, mve_pred16_t p)	a -> Qda imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VORRT.I16 Qda,#imm	Qda -> result	MVE
int32x4_t [__arm_]vorrq_m_n[_s32](int32x4_t a, const int32_t imm, mve_pred16_t p)	a -> Qda imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VORRT.I32 Qda,#imm	Qda -> result	MVE
uint16x8_t [__arm_]vorrq_m_n[_u16](uint16x8_t a, const uint16_t imm, mve_pred16_t p)	a -> Qda imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VORRT.I16 Qda,#imm	Qda -> result	MVE
uint32x4_t [__arm_]vorrq_m_n[_u32](uint32x4_t a, const uint32_t imm, mve_pred16_t p)	a -> Qda imm in AdvSIMDExpandImm p -> Rp	VMSR P0,Rp VPST VORRT.I32 Qda,#imm	Qda -> result	MVE
int8x16_t [__arm_]vqmovnbq[_s16](int8x16_t a, int16x8_t b)	a -> Qd b -> Qm	VQMOVNB.S16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqmovnbq[_s32](int16x8_t a, int32x4_t b)	a -> Qd b -> Qm	VQMOVNB.S32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqmovnbq[_u16](uint8x16_t a, uint16x8_t b)	a -> Qd b -> Qm	VQMOVNB.U16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqmovnbq[_u32](uint16x8_t a, uint32x4_t b)	a -> Qd b -> Qm	VQMOVNB.U32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqmovnbq_m[_s16](int8x16_t a, int16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVNBT.S16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqmovnbq_m[_s32](int16x8_t a, int32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVNBT.S32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqmovnbq_m[_u16](uint8x16_t a, uint16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVNBT.U16 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vqmovnbq_m[_u32](uint16x8_t a, uint32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVNBT.U32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqmovntq[_s16](int8x16_t a, int16x8_t b)	a -> Qd b -> Qm	VQMOVNT.S16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqmovntq[_s32](int16x8_t a, int32x4_t b)	a -> Qd b -> Qm	VQMOVNT.S32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqmovntq[_u16](uint8x16_t a, uint16x8_t b)	a -> Qd b -> Qm	VQMOVNT.U16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqmovntq[_u32](uint16x8_t a, uint32x4_t b)	a -> Qd b -> Qm	VQMOVNT.U32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqmovntq_m[_s16](int8x16_t a, int16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVNTT.S16 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vqmovntq_m[_s32](int16x8_t a, int32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVNTT.S32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqmovntq_m[_u16](uint8x16_t a, uint16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVNTT.U16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqmovntq_m[_u32](uint16x8_t a, uint32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVNTT.U32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqmovunbq[_s16](uint8x16_t a, int16x8_t b)	a -> Qd b -> Qm	VQMOVUNB.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqmovunbq[_s32](uint16x8_t a, int32x4_t b)	a -> Qd b -> Qm	VQMOVUNB.S32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqmovunbq_m[_s16](uint8x16_t a, int16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVUNBT.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqmovunbq_m[_s32](uint16x8_t a, int32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVUNBT.S32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqmovuntq[_s16](uint8x16_t a, int16x8_t b)	a -> Qd b -> Qm	VQMOVUNT.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqmovuntq[_s32](uint16x8_t a, int32x4_t b)	a -> Qd b -> Qm	VQMOVUNT.S32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vqmovuntq_m[_s16](uint8x16_t a, int16x8_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVUNTT.S16 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vqmovuntq_m[_s32](uint16x8_t a, int32x4_t b, mve_pred16_t p)	a -> Qd b -> Qm p -> Rp	VMSR P0,Rp VPST VQMOVUNTT.S32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vqrshlq[_n_s8](int8x16_t a, int32_t b)	a -> Qda b -> Rm	VQRSHL.S8 Qda,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqrshlq[_n_s16](int16x8_t a, int32_t b)	a -> Qda b -> Rm	VQRSHL.S16 Qda,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqrshlq[_n_s32](int32x4_t a, int32_t b)	a -> Qda b -> Rm	VQRSHL.S32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vqrshlq[_n_u8](uint8x16_t a, int32_t b)	a -> Qda b -> Rm	VQRSHL.U8 Qda,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vqrshlq[_n_u16](uint16x8_t a, int32_t b)	a -> Qda b -> Rm	VQRSHL.U16 Qda,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vqrshlq[_n_u32](uint32x4_t a, int32_t b)	a -> Qda b -> Rm	VQRSHL.U32 Qda,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqrshlq_m_n[_s8](int8x16_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQRSHLT.S8 Qda,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqrshlq_m_n[_s16](int16x8_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQRSHLT.S16 Qda,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqrshlq_m_n[_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQRSHLT.S32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vqrshlq_m_n[_u8](uint8x16_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQRSHLT.U8 Qda,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vqrshlq_m_n[_u16](uint16x8_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQRSHLT.U16 Qda,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vqrshlq_m_n[_u32](uint32x4_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQRSHLT.U32 Qda,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqrshlq[_s8](int8x16_t a, int8x16_t b)	a -> Qm b -> Qn	VQRSHL.S8 Qd,Qm,Qn	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int16x8_t [__arm_]vqrshlq[_s16](int16x8_t a, int16x8_t b)	a -> Qm b -> Qn	VQRSHL.S16 Qd,Qm,Qn	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqrshlq[_s32](int32x4_t a, int32x4_t b)	a -> Qm b -> Qn	VQRSHL.S32 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vqrshlq[_u8](uint8x16_t a, int8x16_t b)	a -> Qm b -> Qn	VQRSHL.U8 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vqrshlq[_u16](uint16x8_t a, int16x8_t b)	a -> Qm b -> Qn	VQRSHL.U16 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vqrshlq[_u32](uint32x4_t a, int32x4_t b)	a -> Qm b -> Qn	VQRSHL.U32 Qd,Qm,Qn	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqrshlq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQRSHLT.S8 Qd,Qm,Qn	Qd -> result	MVE
int16x8_t [__arm_]vqrshlq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQRSHLT.S16 Qd,Qm,Qn	Qd -> result	MVE
int32x4_t [__arm_]vqrshlq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQRSHLT.S32 Qd,Qm,Qn	Qd -> result	MVE
uint8x16_t [__arm_]vqrshlq_m[_u8](uint8x16_t inactive, uint8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQRSHLT.U8 Qd,Qm,Qn	Qd -> result	MVE
uint16x8_t [__arm_]vqrshlq_m[_u16](uint16x8_t inactive, uint16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQRSHLT.U16 Qd,Qm,Qn	Qd -> result	MVE
uint32x4_t [__arm_]vqrshlq_m[_u32](uint32x4_t inactive, uint32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQRSHLT.U32 Qd,Qm,Qn	Qd -> result	MVE
int8x16_t [__arm_]vqrshrnbq[_n_s16](int8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQRSHRNB.S16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqrshrnbq[_n_s32](int16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQRSHRNB.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqrshrnbq[_n_u16](uint8x16_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQRSHRNB.U16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqrshrnbq[_n_u32](uint16x8_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQRSHRNB.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqrshrnbq_m[_n_s16](int8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQRSHRNBT.S16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqrshrnbq_m[_n_s32](int16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQRSHRNBT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqrshrnbq_m[_n_u16](uint8x16_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQRSHRNBT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqrshrnbq_m[_n_u32](uint16x8_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQRSHRNBT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqrshrntq[_n_s16](int8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQRSHRNT.S16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqrshrntq[_n_s32](int16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQRSHRNT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqrshrntq[_n_u16](uint8x16_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQRSHRNT.U16 Qd,Qm,#imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vqrshrntq[_n_u32](uint16x8_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQRSHRNT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqrshrntq_m[_n_s16](int8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQRSHRNTT.S16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqrshrntq_m[_n_s32](int16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQRSHRNTT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqrshrntq_m[_n_u16](uint8x16_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQRSHRNTT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqrshrntq_m[_n_u32](uint16x8_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQRSHRNTT.U32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqrshrunbq[_n_s16](uint8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQRSHRUNB.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqrshrunbq[_n_s32](uint16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQRSHRUNB.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqrshrunbq_m[_n_s16](uint8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQRSHRUNBT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqrshrunbq_m[_n_s32](uint16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQRSHRUNBT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqrshruntq[_n_s16](uint8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQRSHRUNT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqrshruntq[_n_s32](uint16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQRSHRUNT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqrshruntq_m[_n_s16](uint8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQRSHRUNTT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqrshruntq_m[_n_s32](uint16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQRSHRUNTT.S32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqshlq[_s8](int8x16_t a, int8x16_t b)	a -> Qm b -> Qn	VQSHL.S8 Qd,Qm,Qn	Qd -> result	MVE/NEON
int16x8_t [__arm_]vqshlq[_s16](int16x8_t a, int16x8_t b)	a -> Qm b -> Qn	VQSHL.S16 Qd,Qm,Qn	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqshlq[_s32](int32x4_t a, int32x4_t b)	a -> Qm b -> Qn	VQSHL.S32 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vqshlq[_u8](uint8x16_t a, int8x16_t b)	a -> Qm b -> Qn	VQSHL.U8 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vqshlq[_u16](uint16x8_t a, int16x8_t b)	a -> Qm b -> Qn	VQSHL.U16 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vqshlq[_u32](uint32x4_t a, int32x4_t b)	a -> Qm b -> Qn	VQSHL.U32 Qd,Qm,Qn	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqshlq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQSHLT.S8 Qd,Qm,Qn	Qd -> result	MVE
int16x8_t [__arm_]vqshlq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQSHLT.S16 Qd,Qm,Qn	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int32x4_t [__arm_]vqshlq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQSHLT.S32 Qd,Qm,Qn	Qd -> result	MVE
uint8x16_t [__arm_]vqshlq_m[_u8](uint8x16_t inactive, uint8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQSHLT.U8 Qd,Qm,Qn	Qd -> result	MVE
uint16x8_t [__arm_]vqshlq_m[_u16](uint16x8_t inactive, uint16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQSHLT.U16 Qd,Qm,Qn	Qd -> result	MVE
uint32x4_t [__arm_]vqshlq_m[_u32](uint32x4_t inactive, uint32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VQSHLT.U32 Qd,Qm,Qn	Qd -> result	MVE
int8x16_t [__arm_]vqshlq_n[_s8](int8x16_t a, const int imm)	a -> Qm 0 <= imm <= 7	VQSHL.S8 Qd,Qm,#imm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vqshlq_n[_s16](int16x8_t a, const int imm)	a -> Qm 0 <= imm <= 15	VQSHL.S16 Qd,Qm,#imm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vqshlq_n[_s32](int32x4_t a, const int imm)	a -> Qm 0 <= imm <= 31	VQSHL.S32 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vqshlq_n[_u8](uint8x16_t a, const int imm)	a -> Qm 0 <= imm <= 7	VQSHL.U8 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vqshlq_n[_u16](uint16x8_t a, const int imm)	a -> Qm 0 <= imm <= 15	VQSHL.U16 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vqshlq_n[_u32](uint32x4_t a, const int imm)	a -> Qm 0 <= imm <= 31	VQSHL.U32 Qd,Qm,#imm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vqshlq_m_n[_s8](int8x16_t inactive, int8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VQSHLT.S8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqshlq_m_n[_s16](int16x8_t inactive, int16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VQSHLT.S16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vqshlq_m_n[_s32](int32x4_t inactive, int32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VQSHLT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshlq_m_n[_u8](uint8x16_t inactive, uint8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VQSHLT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshlq_m_n[_u16](uint16x8_t inactive, uint16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VQSHLT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vqshlq_m_n[_u32](uint32x4_t inactive, uint32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VQSHLT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqshlq_r[_s8](int8x16_t a, int32_t b)	a -> Qda b -> Rm	VQSHL.S8 Qda,Rm	Qda -> result	MVE
int16x8_t [__arm_]vqshlq_r[_s16](int16x8_t a, int32_t b)	a -> Qda b -> Rm	VQSHL.S16 Qda,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqshlq_r[_s32](int32x4_t a, int32_t b)	a -> Qda b -> Rm	VQSHL.S32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vqshlq_r[_u8](uint8x16_t a, int32_t b)	a -> Qda b -> Rm	VQSHL.U8 Qda,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vqshlq_r[_u16](uint16x8_t a, int32_t b)	a -> Qda b -> Rm	VQSHL.U16 Qda,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vqshlq_r[_u32](uint32x4_t a, int32_t b)	a -> Qda b -> Rm	VQSHL.U32 Qda,Rm	Qda -> result	MVE
int8x16_t [__arm_]vqshlq_m_r[_s8](int8x16_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQSHLT.S8 Qda,Rm	Qda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int16x8_t [__arm_]vqshlq_m_r[_s16](int16x8_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQSHLT.S16 Qda,Rm	Qda -> result	MVE
int32x4_t [__arm_]vqshlq_m_r[_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQSHLT.S32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vqshlq_m_r[_u8](uint8x16_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQSHLT.U8 Qda,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vqshlq_m_r[_u16](uint16x8_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQSHLT.U16 Qda,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vqshlq_m_r[_u32](uint32x4_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VQSHLT.U32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vqshluq[_n_s8](int8x16_t a, const int imm)	a -> Qm 0 <= imm <= 7	VQSHLU.S8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshluq[_n_s16](int16x8_t a, const int imm)	a -> Qm 0 <= imm <= 15	VQSHLU.S16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vqshluq[_n_s32](int32x4_t a, const int imm)	a -> Qm 0 <= imm <= 31	VQSHLU.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshluq_m[_n_s8](uint8x16_t inactive, int8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VQSHLUT.S8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshluq_m[_n_s16](uint16x8_t inactive, int16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VQSHLUT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vqshluq_m[_n_s32](uint32x4_t inactive, int32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VQSHLUT.S32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqshrnbq[_n_s16](int8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQSHRNB.S16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqshrnbq[_n_s32](int16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQSHRNB.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshrnbq[_n_u16](uint8x16_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQSHRNB.U16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshrnbq[_n_u32](uint16x8_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQSHRNB.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqshrnbq_m[_n_s16](int8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQSHRNBT.S16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqshrnbq_m[_n_s32](int16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQSHRNBT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshrnbq_m[_n_u16](uint8x16_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQSHRNBT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshrnbq_m[_n_u32](uint16x8_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQSHRNBT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqshrntq[_n_s16](int8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQSHRNT.S16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqshrntq[_n_s32](int16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQSHRNT.S32 Qd,Qm,#imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint8x16_t [__arm_]vqshrntq[_n_u16](uint8x16_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQSHRNT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshrntq[_n_u32](uint16x8_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQSHRNT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vqshrntq_m[_n_s16](int8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQSHRNTT.S16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vqshrntq_m[_n_s32](int16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQSHRNTT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshrntq_m[_n_u16](uint8x16_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQSHRNTT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshrntq_m[_n_u32](uint16x8_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQSHRNTT.U32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshrunbq[_n_s16](uint8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQSHRUNB.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshrunbq[_n_s32](uint16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQSHRUNB.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshrunbq_m[_n_s16](uint8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQSHRUNBT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshrunbq_m[_n_s32](uint16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQSHRUNBT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshruntq[_n_s16](uint8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VQSHRUNT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshruntq[_n_s32](uint16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VQSHRUNT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vqshruntq_m[_n_s16](uint8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VQSHRUNTT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vqshruntq_m[_n_s32](uint16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VQSHRUNTT.S32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vrev16q[_s8](int8x16_t a)	a -> Qm	VREV16.8 Qd,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vrev16q[_u8](uint8x16_t a)	a -> Qm	VREV16.8 Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vrev16q_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV16T.8 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrev16q_m[_u8](uint8x16_t inactive, uint8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV16T.8 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrev16q_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV16T.8 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrev16q_x[_u8](uint8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV16T.8 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrev32q[_s8](int8x16_t a)	a -> Qm	VREV32.8 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vrev32q[_s16](int16x8_t a)	a -> Qm	VREV32.16 Qd,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vrev32q[_u8](uint8x16_t a)	a -> Qm	VREV32.8 Qd,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vrev32q[_u16](uint16x8_t a)	a -> Qm	VREV32.16 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vrev32q[_f16](float16x8_t a)	a -> Qm	VREV32.16 Qd,Qm	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vrev32q_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrev32q_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.16 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrev32q_m[_u8](uint8x16_t inactive, uint8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.8 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vrev32q_m[_u16](uint16x8_t inactive, uint16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.16 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrev32q_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.16 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrev32q_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrev32q_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.16 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrev32q_x[_u8](uint8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.8 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vrev32q_x[_u16](uint16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.16 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrev32q_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV32T.16 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrev64q[_s8](int8x16_t a)	a -> Qm	VREV64.8 Qd,Qm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vrev64q[_s16](int16x8_t a)	a -> Qm	VREV64.16 Qd,Qm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vrev64q[_s32](int32x4_t a)	a -> Qm	VREV64.32 Qd,Qm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vrev64q[_u8](uint8x16_t a)	a -> Qm	VREV64.8 Qd,Qm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vrev64q[_u16](uint16x8_t a)	a -> Qm	VREV64.16 Qd,Qm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vrev64q[_u32](uint32x4_t a)	a -> Qm	VREV64.32 Qd,Qm	Qd -> result	MVE/NEON
float16x8_t [__arm_]vrev64q[_f16](float16x8_t a)	a -> Qm	VREV64.16 Qd,Qm	Qd -> result	MVE/NEON
float32x4_t [__arm_]vrev64q[_f32](float32x4_t a)	a -> Qm	VREV64.32 Qd,Qm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vrev64q_m[_s8](int8x16_t inactive, int8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrev64q_m[_s16](int16x8_t inactive, int16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vrev64q_m[_s32](int32x4_t inactive, int32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrev64q_m[_u8](uint8x16_t inactive, uint8x16_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.8 Qd,Qm	Qd -> result	MVE
uint16x8_t [__arm_]vrev64q_m[_u16](uint16x8_t inactive, uint16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vrev64q_m[_u32](uint32x4_t inactive, uint32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrev64q_m[_f16](float16x8_t inactive, float16x8_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrev64q_m[_f32](float32x4_t inactive, float32x4_t a, mve_pred16_t p)	inactive -> Qd a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrev64q_x[_s8](int8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.8 Qd,Qm	Qd -> result	MVE
int16x8_t [__arm_]vrev64q_x[_s16](int16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.16 Qd,Qm	Qd -> result	MVE
int32x4_t [__arm_]vrev64q_x[_s32](int32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.32 Qd,Qm	Qd -> result	MVE
uint8x16_t [__arm_]vrev64q_x[_u8](uint8x16_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.8 Qd,Qm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vrev64q_x[_u16](uint16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.16 Qd,Qm	Qd -> result	MVE
uint32x4_t [__arm_]vrev64q_x[_u32](uint32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.32 Qd,Qm	Qd -> result	MVE
float16x8_t [__arm_]vrev64q_x[_f16](float16x8_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.16 Qd,Qm	Qd -> result	MVE
float32x4_t [__arm_]vrev64q_x[_f32](float32x4_t a, mve_pred16_t p)	a -> Qm p -> Rp	VMSR P0,Rp VPST VREV64T.32 Qd,Qm	Qd -> result	MVE
int8x16_t [__arm_]vrshlq[_n_s8](int8x16_t a, int32_t b)	a -> Qda b -> Rm	VRSHL.S8 Qda,Rm	Qda -> result	MVE
int16x8_t [__arm_]vrshlq[_n_s16](int16x8_t a, int32_t b)	a -> Qda b -> Rm	VRSHL.S16 Qda,Rm	Qda -> result	MVE
int32x4_t [__arm_]vrshlq[_n_s32](int32x4_t a, int32_t b)	a -> Qda b -> Rm	VRSHL.S32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vrshlq[_n_u8](uint8x16_t a, int32_t b)	a -> Qda b -> Rm	VRSHL.U8 Qda,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vrshlq[_n_u16](uint16x8_t a, int32_t b)	a -> Qda b -> Rm	VRSHL.U16 Qda,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vrshlq[_n_u32](uint32x4_t a, int32_t b)	a -> Qda b -> Rm	VRSHL.U32 Qda,Rm	Qda -> result	MVE
int8x16_t [__arm_]vrshlq_m_n[_s8](int8x16_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VRSHLT.S8 Qda,Rm	Qda -> result	MVE
int16x8_t [__arm_]vrshlq_m_n[_s16](int16x8_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VRSHLT.S16 Qda,Rm	Qda -> result	MVE
int32x4_t [__arm_]vrshlq_m_n[_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VRSHLT.S32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vrshlq_m_n[_u8](uint8x16_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VRSHLT.U8 Qda,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vrshlq_m_n[_u16](uint16x8_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VRSHLT.U16 Qda,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vrshlq_m_n[_u32](uint32x4_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VRSHLT.U32 Qda,Rm	Qda -> result	MVE
int8x16_t [__arm_]vrshlq[_s8](int8x16_t a, int8x16_t b)	a -> Qm b -> Qn	VRSHL.S8 Qd,Qm,Qn	Qd -> result	MVE/NEON
int16x8_t [__arm_]vrshlq[_s16](int16x8_t a, int16x8_t b)	a -> Qm b -> Qn	VRSHL.S16 Qd,Qm,Qn	Qd -> result	MVE/NEON
int32x4_t [__arm_]vrshlq[_s32](int32x4_t a, int32x4_t b)	a -> Qm b -> Qn	VRSHL.S32 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vrshlq[_u8](uint8x16_t a, int8x16_t b)	a -> Qm b -> Qn	VRSHL.U8 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vrshlq[_u16](uint16x8_t a, int16x8_t b)	a -> Qm b -> Qn	VRSHL.U16 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vrshlq[_u32](uint32x4_t a, int32x4_t b)	a -> Qm b -> Qn	VRSHL.U32 Qd,Qm,Qn	Qd -> result	MVE/NEON
int8x16_t [__arm_]vrshlq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.S8 Qd,Qm,Qn	Qd -> result	MVE
int16x8_t [__arm_]vrshlq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.S16 Qd,Qm,Qn	Qd -> result	MVE
int32x4_t [__arm_]vrshlq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.S32 Qd,Qm,Qn	Qd -> result	MVE
uint8x16_t [__arm_]vrshlq_m[_u8](uint8x16_t inactive, uint8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.U8 Qd,Qm,Qn	Qd -> result	MVE
uint16x8_t [__arm_]vrshlq_m[_u16](uint16x8_t inactive, uint16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.U16 Qd,Qm,Qn	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vrshlq_m[_u32](uint32x4_t inactive, uint32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.U32 Qd,Qm,Qn	Qd -> result	MVE
int8x16_t [__arm_]vrshlq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.S8 Qd,Qm,Qn	Qd -> result	MVE
int16x8_t [__arm_]vrshlq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.S16 Qd,Qm,Qn	Qd -> result	MVE
int32x4_t [__arm_]vrshlq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.S32 Qd,Qm,Qn	Qd -> result	MVE
uint8x16_t [__arm_]vrshlq_x[_u8](uint8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.U8 Qd,Qm,Qn	Qd -> result	MVE
uint16x8_t [__arm_]vrshlq_x[_u16](uint16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.U16 Qd,Qm,Qn	Qd -> result	MVE
uint32x4_t [__arm_]vrshlq_x[_u32](uint32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VRSHLT.U32 Qd,Qm,Qn	Qd -> result	MVE
int8x16_t [__arm_]vshlcq[_s8](int8x16_t a, uint32_t * b, const int imm)	a -> Qda *b -> Rdm 1 <= imm <= 32	VSHLC Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
int16x8_t [__arm_]vshlcq[_s16](int16x8_t a, uint32_t * b, const int imm)	a -> Qda *b -> Rdm 1 <= imm <= 32	VSHLC Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
int32x4_t [__arm_]vshlcq[_s32](int32x4_t a, uint32_t * b, const int imm)	a -> Qda *b -> Rdm 1 <= imm <= 32	VSHLC Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
uint8x16_t [__arm_]vshlcq[_u8](uint8x16_t a, uint32_t * b, const int imm)	a -> Qda *b -> Rdm 1 <= imm <= 32	VSHLC Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
uint16x8_t [__arm_]vshlcq[_u16](uint16x8_t a, uint32_t * b, const int imm)	a -> Qda *b -> Rdm 1 <= imm <= 32	VSHLC Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
uint32x4_t [__arm_]vshlcq[_u32](uint32x4_t a, uint32_t * b, const int imm)	a -> Qda *b -> Rdm 1 <= imm <= 32	VSHLC Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
int8x16_t [__arm_]vshlcq_m[_s8](int8x16_t a, uint32_t * b, const int imm, mve_pred16_t p)	a -> Qda *b -> Rdm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHLCT Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
int16x8_t [__arm_]vshlcq_m[_s16](int16x8_t a, uint32_t * b, const int imm, mve_pred16_t p)	a -> Qda *b -> Rdm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHLCT Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
int32x4_t [__arm_]vshlcq_m[_s32](int32x4_t a, uint32_t * b, const int imm, mve_pred16_t p)	a -> Qda *b -> Rdm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHLCT Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
uint8x16_t [__arm_]vshlcq_m[_u8](uint8x16_t a, uint32_t * b, const int imm, mve_pred16_t p)	a -> Qda *b -> Rdm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHLCT Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
uint16x8_t [__arm_]vshlcq_m[_u16](uint16x8_t a, uint32_t * b, const int imm, mve_pred16_t p)	a -> Qda *b -> Rdm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHLCT Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE
uint32x4_t [__arm_]vshlcq_m[_u32](uint32x4_t a, uint32_t * b, const int imm, mve_pred16_t p)	a -> Qda *b -> Rdm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHLCT Qda,Rdm,#imm	Qda -> result Rdm -> *b	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int16x8_t [__arm_]vshllbq[_n_s8](int8x16_t a, const int imm)	a -> Qm 1 <= imm <= 8	VSHLLB.S8 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshllbq[_n_s16](int16x8_t a, const int imm)	a -> Qm 1 <= imm <= 16	VSHLLB.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshllbq[_n_u8](uint8x16_t a, const int imm)	a -> Qm 1 <= imm <= 8	VSHLLB.U8 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshllbq[_n_u16](uint16x8_t a, const int imm)	a -> Qm 1 <= imm <= 16	VSHLLB.U16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshllbq_m[_n_s8](int16x8_t inactive, int8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHLLBT.S8 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshllbq_m[_n_s16](int32x4_t inactive, int16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHLLBT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshllbq_m[_n_u8](uint16x8_t inactive, uint8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHLLBT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshllbq_m[_n_u16](uint32x4_t inactive, uint16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHLLBT.U16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshllbq_x[_n_s8](int8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHLLBT.S8 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshllbq_x[_n_s16](int16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHLLBT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshllbq_x[_n_u8](uint8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHLLBT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshllbq_x[_n_u16](uint16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHLLBT.U16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshlltq[_n_s8](int8x16_t a, const int imm)	a -> Qm 1 <= imm <= 8	VSHLLT.S8 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshlltq[_n_s16](int16x8_t a, const int imm)	a -> Qm 1 <= imm <= 16	VSHLLT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshlltq[_n_u8](uint8x16_t a, const int imm)	a -> Qm 1 <= imm <= 8	VSHLLT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshlltq[_n_u16](uint16x8_t a, const int imm)	a -> Qm 1 <= imm <= 16	VSHLLT.U16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshlltq_m[_n_s8](int16x8_t inactive, int8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHLLTT.S8 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshlltq_m[_n_s16](int32x4_t inactive, int16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHLLTT.S16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshlltq_m[_n_u8](uint16x8_t inactive, uint8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHLLTT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshlltq_m[_n_u16](uint32x4_t inactive, uint16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHLLTT.U16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshlltq_x[_n_s8](int8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHLLTT.S8 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshlltq_x[_n_s16](int16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHLLTT.S16 Qd,Qm,#imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vshlltq_x[_n_u8](uint8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHLLTT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshlltq_x[_n_u16](uint16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHLLTT.U16 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshlq[_s8](int8x16_t a, int8x16_t b)	a -> Qm b -> Qn	VSHL.S8 Qd,Qm,Qn	Qd -> result	MVE/NEON
int16x8_t [__arm_]vshlq[_s16](int16x8_t a, int16x8_t b)	a -> Qm b -> Qn	VSHL.S16 Qd,Qm,Qn	Qd -> result	MVE/NEON
int32x4_t [__arm_]vshlq[_s32](int32x4_t a, int32x4_t b)	a -> Qm b -> Qn	VSHL.S32 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vshlq[_u8](uint8x16_t a, int8x16_t b)	a -> Qm b -> Qn	VSHL.U8 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vshlq[_u16](uint16x8_t a, int16x8_t b)	a -> Qm b -> Qn	VSHL.U16 Qd,Qm,Qn	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vshlq[_u32](uint32x4_t a, int32x4_t b)	a -> Qm b -> Qn	VSHL.U32 Qd,Qm,Qn	Qd -> result	MVE/NEON
int8x16_t [__arm_]vshlq_m[_s8](int8x16_t inactive, int8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.S8 Qd,Qm,Qn	Qd -> result	MVE
int16x8_t [__arm_]vshlq_m[_s16](int16x8_t inactive, int16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.S16 Qd,Qm,Qn	Qd -> result	MVE
int32x4_t [__arm_]vshlq_m[_s32](int32x4_t inactive, int32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.S32 Qd,Qm,Qn	Qd -> result	MVE
uint8x16_t [__arm_]vshlq_m[_u8](uint8x16_t inactive, uint8x16_t a, int8x16_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.U8 Qd,Qm,Qn	Qd -> result	MVE
uint16x8_t [__arm_]vshlq_m[_u16](uint16x8_t inactive, uint16x8_t a, int16x8_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.U16 Qd,Qm,Qn	Qd -> result	MVE
uint32x4_t [__arm_]vshlq_m[_u32](uint32x4_t inactive, uint32x4_t a, int32x4_t b, mve_pred16_t p)	inactive -> Qd a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.U32 Qd,Qm,Qn	Qd -> result	MVE
int8x16_t [__arm_]vshlq_x[_s8](int8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.S8 Qd,Qm,Qn	Qd -> result	MVE
int16x8_t [__arm_]vshlq_x[_s16](int16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.S16 Qd,Qm,Qn	Qd -> result	MVE
int32x4_t [__arm_]vshlq_x[_s32](int32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.S32 Qd,Qm,Qn	Qd -> result	MVE
uint8x16_t [__arm_]vshlq_x[_u8](uint8x16_t a, int8x16_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.U8 Qd,Qm,Qn	Qd -> result	MVE
uint16x8_t [__arm_]vshlq_x[_u16](uint16x8_t a, int16x8_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.U16 Qd,Qm,Qn	Qd -> result	MVE
uint32x4_t [__arm_]vshlq_x[_u32](uint32x4_t a, int32x4_t b, mve_pred16_t p)	a -> Qm b -> Qn p -> Rp	VMSR P0,Rp VPST VSHLT.U32 Qd,Qm,Qn	Qd -> result	MVE
int8x16_t [__arm_]vshlq_n[_s8](int8x16_t a, const int imm)	a -> Qm 0 <= imm <= 7	VSHL.S8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshlq_n[_s16](int16x8_t a, const int imm)	a -> Qm 0 <= imm <= 15	VSHL.S16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshlq_n[_s32](int32x4_t a, const int imm)	a -> Qm 0 <= imm <= 31	VSHL.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vshlq_n[_u8](uint8x16_t a, const int imm)	a -> Qm 0 <= imm <= 7	VSHL.U8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshlq_n[_u16](uint16x8_t a, const int imm)	a -> Qm 0 <= imm <= 15	VSHL.U16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshlq_n[_u32](uint32x4_t a, const int imm)	a -> Qm 0 <= imm <= 31	VSHL.U32 Qd,Qm,#imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vshlq_m_n[_s8](int8x16_t inactive, int8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VSHLT.S8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshlq_m_n[_s16](int16x8_t inactive, int16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VSHLT.S16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshlq_m_n[_s32](int32x4_t inactive, int32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VSHLT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vshlq_m_n[_u8](uint8x16_t inactive, uint8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VSHLT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshlq_m_n[_u16](uint16x8_t inactive, uint16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VSHLT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshlq_m_n[_u32](uint32x4_t inactive, uint32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VSHLT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshlq_x_n[_s8](int8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VSHLT.S8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshlq_x_n[_s16](int16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VSHLT.S16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshlq_x_n[_s32](int32x4_t a, const int imm, mve_pred16_t p)	a -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VSHLT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vshlq_x_n[_u8](uint8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VSHLT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshlq_x_n[_u16](uint16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VSHLT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshlq_x_n[_u32](uint32x4_t a, const int imm, mve_pred16_t p)	a -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VSHLT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshlq_r[_s8](int8x16_t a, int32_t b)	a -> Qda b -> Rm	VSHL.S8 Qda,Rm	Qda -> result	MVE
int16x8_t [__arm_]vshlq_r[_s16](int16x8_t a, int32_t b)	a -> Qda b -> Rm	VSHL.S16 Qda,Rm	Qda -> result	MVE
int32x4_t [__arm_]vshlq_r[_s32](int32x4_t a, int32_t b)	a -> Qda b -> Rm	VSHL.S32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vshlq_r[_u8](uint8x16_t a, int32_t b)	a -> Qda b -> Rm	VSHL.U8 Qda,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vshlq_r[_u16](uint16x8_t a, int32_t b)	a -> Qda b -> Rm	VSHL.U16 Qda,Rm	Qda -> result	MVE
uint32x4_t [__arm_]vshlq_r[_u32](uint32x4_t a, int32_t b)	a -> Qda b -> Rm	VSHL.U32 Qda,Rm	Qda -> result	MVE
int8x16_t [__arm_]vshlq_m_r[_s8](int8x16_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VSHLT.S8 Qda,Rm	Qda -> result	MVE
int16x8_t [__arm_]vshlq_m_r[_s16](int16x8_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VSHLT.S16 Qda,Rm	Qda -> result	MVE
int32x4_t [__arm_]vshlq_m_r[_s32](int32x4_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VSHLT.S32 Qda,Rm	Qda -> result	MVE
uint8x16_t [__arm_]vshlq_m_r[_u8](uint8x16_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VSHLT.U8 Qda,Rm	Qda -> result	MVE
uint16x8_t [__arm_]vshlq_m_r[_u16](uint16x8_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VSHLT.U16 Qda,Rm	Qda -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32x4_t [__arm_]vshlq_m_r[_u32](uint32x4_t a, int32_t b, mve_pred16_t p)	a -> Qda b -> Rm p -> Rp	VMSR P0,Rp VPST VSHLT.U32 Qda,Rm	Qda -> result	MVE
int8x16_t [__arm_]vrshrnbq[_n_s16](int8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VRSHRNB.I16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vrshrnbq[_n_s32](int16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VRSHRNB.I32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vrshrnbq[_n_u16](uint8x16_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VRSHRNB.I16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vrshrnbq[_n_u32](uint16x8_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VRSHRNB.I32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vrshrnbq_m[_n_s16](int8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VRSHRNBT.I16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vrshrnbq_m[_n_s32](int16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VRSHRNBT.I32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vrshrnbq_m[_n_u16](uint8x16_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VRSHRNBT.I16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vrshrnbq_m[_n_u32](uint16x8_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VRSHRNBT.I32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vrshrntq[_n_s16](int8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VRSHRNT.I16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vrshrntq[_n_s32](int16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VRSHRNT.I32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vrshrntq[_n_u16](uint8x16_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VRSHRNT.I16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vrshrntq[_n_u32](uint16x8_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VRSHRNT.I32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vrshrntq_m[_n_s16](int8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VRSHRNTT.I16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vrshrntq_m[_n_s32](int16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VRSHRNTT.I32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vrshrntq_m[_n_u16](uint8x16_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VRSHRNTT.I16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vrshrntq_m[_n_u32](uint16x8_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VRSHRNTT.I32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vrshrq[_n_s8](int8x16_t a, const int imm)	a -> Qm 1 <= imm <= 8	VRSHR.S8 Qd,Qm,#imm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vrshrq[_n_s16](int16x8_t a, const int imm)	a -> Qm 1 <= imm <= 16	VRSHR.S16 Qd,Qm,#imm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vrshrq[_n_s32](int32x4_t a, const int imm)	a -> Qm 1 <= imm <= 32	VRSHR.S32 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vrshrq[_n_u8](uint8x16_t a, const int imm)	a -> Qm 1 <= imm <= 8	VRSHR.U8 Qd,Qm,#imm	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vrshrq[_n_u16](uint16x8_t a, const int imm)	a -> Qm 1 <= imm <= 16	VRSHR.U16 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vrshrq[_n_u32](uint32x4_t a, const int imm)	a -> Qm 1 <= imm <= 32	VRSHR.U32 Qd,Qm,#imm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vrshrq_m[_n_s8](int8x16_t inactive, int8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VRSHRT.S8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vrshrq_m[_n_s16](int16x8_t inactive, int16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VRSHRT.S16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vrshrq_m[_n_s32](int32x4_t inactive, int32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VRSHRT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vrshrq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VRSHRT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vrshrq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VRSHRT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vrshrq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VRSHRT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vrshrq_x[_n_s8](int8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VRSHRT.S8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vrshrq_x[_n_s16](int16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VRSHRT.S16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vrshrq_x[_n_s32](int32x4_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VRSHRT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vrshrq_x[_n_u8](uint8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VRSHRT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vrshrq_x[_n_u16](uint16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VRSHRT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vrshrq_x[_n_u32](uint32x4_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VRSHRT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshrnbq[_n_s16](int8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VSHRNB.I16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshrnbq[_n_s32](int16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VSHRNB.I32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vshrnbq[_n_u16](uint8x16_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VSHRNB.I16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshrnbq[_n_u32](uint16x8_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VSHRNB.I32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshrnbq_m[_n_s16](int8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHRNBT.I16 Qd,Qm,#imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int16x8_t [__arm_]vshrnbq_m[_n_s32](int16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHRNBT.I32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vshrnbq_m[_n_u16](uint8x16_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHRNBT.I16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshrnbq_m[_n_u32](uint16x8_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHRNBT.I32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshrntq[_n_s16](int8x16_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VSHRNT.I16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshrntq[_n_s32](int16x8_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VSHRNT.I32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vshrntq[_n_u16](uint8x16_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VSHRNT.I16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshrntq[_n_u32](uint16x8_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VSHRNT.I32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshrntq_m[_n_s16](int8x16_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHRNTT.I16 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshrntq_m[_n_s32](int16x8_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHRNTT.I32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vshrntq_m[_n_u16](uint8x16_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHRNTT.I16 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshrntq_m[_n_u32](uint16x8_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHRNTT.I32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshrq[_n_s8](int8x16_t a, const int imm)	a -> Qm 1 <= imm <= 8	VSHR.S8 Qd,Qm,#imm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vshrq[_n_s16](int16x8_t a, const int imm)	a -> Qm 1 <= imm <= 16	VSHR.S16 Qd,Qm,#imm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vshrq[_n_s32](int32x4_t a, const int imm)	a -> Qm 1 <= imm <= 32	VSHR.S32 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vshrq[_n_u8](uint8x16_t a, const int imm)	a -> Qm 1 <= imm <= 8	VSHR.U8 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vshrq[_n_u16](uint16x8_t a, const int imm)	a -> Qm 1 <= imm <= 16	VSHR.U16 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vshrq[_n_u32](uint32x4_t a, const int imm)	a -> Qm 1 <= imm <= 32	VSHR.U32 Qd,Qm,#imm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vshrq_m[_n_s8](int8x16_t inactive, int8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHRT.S8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshrq_m[_n_s16](int16x8_t inactive, int16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHRT.S16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshrq_m[_n_s32](int32x4_t inactive, int32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHRT.S32 Qd,Qm,#imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint8x16_t [__arm_]vshrq_m[_n_u8](uint8x16_t inactive, uint8x16_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHRT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshrq_m[_n_u16](uint16x8_t inactive, uint16x8_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHRT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshrq_m[_n_u32](uint32x4_t inactive, uint32x4_t a, const int imm, mve_pred16_t p)	inactive -> Qd a -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHRT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vshrq_x[_n_s8](int8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHRT.S8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vshrq_x[_n_s16](int16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHRT.S16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vshrq_x[_n_s32](int32x4_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHRT.S32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vshrq_x[_n_u8](uint8x16_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSHRT.U8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vshrq_x[_n_u16](uint16x8_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSHRT.U16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vshrq_x[_n_u32](uint32x4_t a, const int imm, mve_pred16_t p)	a -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSHRT.U32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vsliq[_n_s8](int8x16_t a, int8x16_t b, const int imm)	a -> Qd b -> Qm 0 <= imm <= 7	VSLI.8 Qd,Qm,#imm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vsliq[_n_s16](int16x8_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 0 <= imm <= 15	VSLI.16 Qd,Qm,#imm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vsliq[_n_s32](int32x4_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 0 <= imm <= 31	VSLI.32 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vsliq[_n_u8](uint8x16_t a, uint8x16_t b, const int imm)	a -> Qd b -> Qm 0 <= imm <= 7	VSLI.8 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vsliq[_n_u16](uint16x8_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 0 <= imm <= 15	VSLI.16 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vsliq[_n_u32](uint32x4_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 0 <= imm <= 31	VSLI.32 Qd,Qm,#imm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vsliq_m[_n_s8](int8x16_t a, int8x16_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VSLIT.8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vsliq_m[_n_s16](int16x8_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VSLIT.16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vsliq_m[_n_s32](int32x4_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VSLIT.32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vsliq_m[_n_u8](uint8x16_t a, uint8x16_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 0 <= imm <= 7 p -> Rp	VMSR P0,Rp VPST VSLIT.8 Qd,Qm,#imm	Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint16x8_t [__arm_]vsliq_m[_n_u16](uint16x8_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 0 <= imm <= 15 p -> Rp	VMSR P0,Rp VPST VSLIT.16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vsliq_m[_n_u32](uint32x4_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 0 <= imm <= 31 p -> Rp	VMSR P0,Rp VPST VSLIT.32 Qd,Qm,#imm	Qd -> result	MVE
int8x16_t [__arm_]vsriq[_n_s8](int8x16_t a, int8x16_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VSRI.8 Qd,Qm,#imm	Qd -> result	MVE/NEON
int16x8_t [__arm_]vsriq[_n_s16](int16x8_t a, int16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VSRI.16 Qd,Qm,#imm	Qd -> result	MVE/NEON
int32x4_t [__arm_]vsriq[_n_s32](int32x4_t a, int32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 32	VSRI.32 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vsriq[_n_u8](uint8x16_t a, uint8x16_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 8	VSRI.8 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vsriq[_n_u16](uint16x8_t a, uint16x8_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 16	VSRI.16 Qd,Qm,#imm	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vsriq[_n_u32](uint32x4_t a, uint32x4_t b, const int imm)	a -> Qd b -> Qm 1 <= imm <= 32	VSRI.32 Qd,Qm,#imm	Qd -> result	MVE/NEON
int8x16_t [__arm_]vsriq_m[_n_s8](int8x16_t a, int8x16_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSRIT.8 Qd,Qm,#imm	Qd -> result	MVE
int16x8_t [__arm_]vsriq_m[_n_s16](int16x8_t a, int16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSRIT.16 Qd,Qm,#imm	Qd -> result	MVE
int32x4_t [__arm_]vsriq_m[_n_s32](int32x4_t a, int32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSRIT.32 Qd,Qm,#imm	Qd -> result	MVE
uint8x16_t [__arm_]vsriq_m[_n_u8](uint8x16_t a, uint8x16_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 8 p -> Rp	VMSR P0,Rp VPST VSRIT.8 Qd,Qm,#imm	Qd -> result	MVE
uint16x8_t [__arm_]vsriq_m[_n_u16](uint16x8_t a, uint16x8_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 16 p -> Rp	VMSR P0,Rp VPST VSRIT.16 Qd,Qm,#imm	Qd -> result	MVE
uint32x4_t [__arm_]vsriq_m[_n_u32](uint32x4_t a, uint32x4_t b, const int imm, mve_pred16_t p)	a -> Qd b -> Qm 1 <= imm <= 32 p -> Rp	VMSR P0,Rp VPST VSRIT.32 Qd,Qm,#imm	Qd -> result	MVE
float16_t [__arm_]vgetq_lane[_f16](float16x8_t a, const int idx)	a -> Qn 0 <= idx <= 7	VMOV.U16 Rt,Qn[idx]	Rt -> result	MVE/NEON
float32_t [__arm_]vgetq_lane[_f32](float32x4_t a, const int idx)	a -> Qn 0 <= idx <= 3	VMOV.32 Rt,Qn[idx]	Rt -> result	MVE/NEON
int8_t [__arm_]vgetq_lane[_s8](int8x16_t a, const int idx)	a -> Qn 0 <= idx <= 15	VMOV.S8 Rt,Qn[idx]	Rt -> result	MVE/NEON
int16_t [__arm_]vgetq_lane[_s16](int16x8_t a, const int idx)	a -> Qn 0 <= idx <= 7	VMOV.S16 Rt,Qn[idx]	Rt -> result	MVE/NEON
int32_t [__arm_]vgetq_lane[_s32](int32x4_t a, const int idx)	a -> Qn 0 <= idx <= 3	VMOV.32 Rt,Qn[idx]	Rt -> result	MVE/NEON
int64_t [__arm_]vgetq_lane[_s64](int64x2_t a, const int idx)	a -> Qn 0 <= idx <= 1	VMOV Rt1,Rt2,D(2*n+idx)	[Rt1,Rt2] -> result	MVE/NEON
uint8_t [__arm_]vgetq_lane[_u8](uint8x16_t a, const int idx)	a -> Qn 0 <= idx <= 15	VMOV.U8 Rt,Qn[idx]	Rt -> result	MVE/NEON
uint16_t [__arm_]vgetq_lane[_u16](uint16x8_t a, const int idx)	a -> Qn 0 <= idx <= 7	VMOV.U16 Rt,Qn[idx]	Rt -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint32_t [__arm_]vgetq_lane[_u32](uint32x4_t a, const int idx)	a -> Qn 0 <= idx <= 3	VMOV.32 Rt,Qn[idx]	Rt -> result	MVE/NEON
uint64_t [__arm_]vgetq_lane[_u64](uint64x2_t a, const int idx)	a -> Qn 0 <= idx <= 1	VMOV Rt1,Rt2,D(2*n+idx)	[Rt1,Rt2] -> result	MVE/NEON
float16x8_t [__arm_]vsetq_lane[_f16](float16_t a, float16x8_t b, const int idx)	a -> Rt b -> Qd 0 <= idx <= 7	VMOV.16 Qd[idx],Rt	Qd -> result	MVE/NEON
float32x4_t [__arm_]vsetq_lane[_f32](float32_t a, float32x4_t b, const int idx)	a -> Rt b -> Qd 0 <= idx <= 3	VMOV.32 Qd[idx],Rt	Qd -> result	MVE/NEON
int8x16_t [__arm_]vsetq_lane[_s8](int8_t a, int8x16_t b, const int idx)	a -> Rt b -> Qd 0 <= idx <= 15	VMOV.8 Qd[idx],Rt	Qd -> result	MVE/NEON
int16x8_t [__arm_]vsetq_lane[_s16](int16_t a, int16x8_t b, const int idx)	a -> Rt b -> Qd 0 <= idx <= 7	VMOV.16 Qd[idx],Rt	Qd -> result	MVE/NEON
int32x4_t [__arm_]vsetq_lane[_s32](int32_t a, int32x4_t b, const int idx)	a -> Rt b -> Qd 0 <= idx <= 3	VMOV.32 Qd[idx],Rt	Qd -> result	MVE/NEON
int64x2_t [__arm_]vsetq_lane[_s64](int64_t a, int64x2_t b, const int idx)	a -> [Rt1,Rt2] b -> Qd 0 <= idx <= 1	VMOV D(2*d+idx),Rt1,Rt2	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vsetq_lane[_u8](uint8_t a, uint8x16_t b, const int idx)	a -> Rt b -> Qd 0 <= idx <= 15	VMOV.8 Qd[idx],Rt	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vsetq_lane[_u16](uint16_t a, uint16x8_t b, const int idx)	a -> Rt b -> Qd 0 <= idx <= 7	VMOV.16 Qd[idx],Rt	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vsetq_lane[_u32](uint32_t a, uint32x4_t b, const int idx)	a -> Rt b -> Qd 0 <= idx <= 3	VMOV.32 Qd[idx],Rt	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vsetq_lane[_u64](uint64_t a, uint64x2_t b, const int idx)	a -> [Rt1,Rt2] b -> Qd 0 <= idx <= 1	VMOV D(2*d+idx),Rt1,Rt2	Qd -> result	MVE/NEON
mve_pred16_t [__arm_]vctp8q(uint32_t a)	a -> Rn	VCTP.8 Rn VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vctp16q(uint32_t a)	a -> Rn	VCTP.16 Rn VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vctp32q(uint32_t a)	a -> Rn	VCTP.32 Rn VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vctp64q(uint32_t a)	a -> Rn	VCTP.64 Rn VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vctp8q_m(uint32_t a, mve_pred16_t p)	a -> Rn p -> Rp	VMSR P0,Rp VPST VCTPT.8 Rn VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vctp16q_m(uint32_t a, mve_pred16_t p)	a -> Rn p -> Rp	VMSR P0,Rp VPST VCTPT.16 Rn VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vctp32q_m(uint32_t a, mve_pred16_t p)	a -> Rn p -> Rp	VMSR P0,Rp VPST VCTPT.32 Rn VMRS Rd,P0	Rd -> result	MVE
mve_pred16_t [__arm_]vctp64q_m(uint32_t a, mve_pred16_t p)	a -> Rn p -> Rp	VMSR P0,Rp VPST VCTPT.64 Rn VMRS Rd,P0	Rd -> result	MVE
int8x16_t [__arm_]vuninitializedq_s8(void)			Qd -> result	MVE
int16x8_t [__arm_]vuninitializedq_s16(void)			Qd -> result	MVE
int32x4_t [__arm_]vuninitializedq_s32(void)			Qd -> result	MVE
int64x2_t [__arm_]vuninitializedq_s64(void)			Qd -> result	MVE
uint8x16_t [__arm_]vuninitializedq_u8(void)			Qd -> result	MVE
uint16x8_t [__arm_]vuninitializedq_u16(void)			Qd -> result	MVE
uint32x4_t [__arm_]vuninitializedq_u32(void)			Qd -> result	MVE
uint64x2_t [__arm_]vuninitializedq_u64(void)			Qd -> result	MVE
float16x8_t [__arm_]vuninitializedq_f16(void)			Qd -> result	MVE
float32x4_t [__arm_]vuninitializedq_f32(void)			Qd -> result	MVE
int8x16_t [__arm_]vuninitializedq(int8x16_t t)	t -> Do Not Evaluate		Qd -> result	MVE
int16x8_t [__arm_]vuninitializedq(int16x8_t t)	t -> Do Not Evaluate		Qd -> result	MVE
int32x4_t [__arm_]vuninitializedq(int32x4_t t)	t -> Do Not Evaluate		Qd -> result	MVE
int64x2_t [__arm_]vuninitializedq(int64x2_t t)	t -> Do Not Evaluate		Qd -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
uint8x16_t [__arm_]vuninitializedq(uint8x16_t t)	t -> Do Not Evaluate		Qd -> result	MVE
uint16x8_t [__arm_]vuninitializedq(uint16x8_t t)	t -> Do Not Evaluate		Qd -> result	MVE
uint32x4_t [__arm_]vuninitializedq(uint32x4_t t)	t -> Do Not Evaluate		Qd -> result	MVE
uint64x2_t [__arm_]vuninitializedq(uint64x2_t t)	t -> Do Not Evaluate		Qd -> result	MVE
float16x8_t [__arm_]vuninitializedq(float16x8_t t)	t -> Do Not Evaluate		Qd -> result	MVE
float32x4_t [__arm_]vuninitializedq(float32x4_t t)	t -> Do Not Evaluate		Qd -> result	MVE
int16x8_t [__arm_]vreinterpretq_s16[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_s8](int8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int8x16_t [__arm_]vreinterpretq_s8[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_s16](int16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int8x16_t [__arm_]vreinterpretq_s8[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int16x8_t [__arm_]vreinterpretq_s16[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_s32](int32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int8x16_t [__arm_]vreinterpretq_s8[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int16x8_t [__arm_]vreinterpretq_s16[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_f32](float32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int8x16_t [__arm_]vreinterpretq_s8[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int16x8_t [__arm_]vreinterpretq_s16[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_u8](uint8x16_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int8x16_t [__arm_]vreinterpretq_s8[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int16x8_t [__arm_]vreinterpretq_s16[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_u16](uint16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int8x16_t [__arm_]vreinterpretq_s8[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int16x8_t [__arm_]vreinterpretq_s16[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_u32](uint32x4_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int8x16_t [__arm_]vreinterpretq_s8[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int16x8_t [__arm_]vreinterpretq_s16[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_u64](uint64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int8x16_t [__arm_]vreinterpretq_s8[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int16x8_t [__arm_]vreinterpretq_s16[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float16x8_t [__arm_]vreinterpretq_f16[_s64](int64x2_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int8x16_t [__arm_]vreinterpretq_s8[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int16x8_t [__arm_]vreinterpretq_s16[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int32x4_t [__arm_]vreinterpretq_s32[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
float32x4_t [__arm_]vreinterpretq_f32[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint8x16_t [__arm_]vreinterpretq_u8[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint16x8_t [__arm_]vreinterpretq_u16[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint32x4_t [__arm_]vreinterpretq_u32[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64x2_t [__arm_]vreinterpretq_u64[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
int64x2_t [__arm_]vreinterpretq_s64[_f16](float16x8_t a)	a -> Qd	NOP	Qd -> result	MVE/NEON
uint64_t [__arm_]lsll(uint64_t value, int32_t shift)	value -> [RdaHi,RdaLo] shift -> Rm	LSLL RdaLo,RdaHi,Rm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]asrl(int64_t value, int32_t shift)	value -> [RdaHi,RdaLo] shift -> Rm	ASRL RdaLo,RdaHi,Rm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]uqrshll(uint64_t value, int32_t shift)	value -> [RdaHi,RdaLo] shift -> Rm	UQRSHLL RdaLo,RdaHi,#64,Rm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]uqrshll_sat48(uint64_t value, int32_t shift)	value -> [RdaHi,RdaLo] shift -> Rm	UQRSHLL RdaLo,RdaHi,#48,Rm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]sqrshrl(int64_t value, int32_t shift)	value -> [RdaHi,RdaLo] shift -> Rm	SQRSHRL RdaLo,RdaHi,#64,Rm	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]sqrshrl_sat48(int64_t value, int32_t shift)	value -> [RdaHi,RdaLo] shift -> Rm	SQRSHRL RdaLo,RdaHi,#48,Rm	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]uqshll(uint64_t value, const int shift)	value -> [RdaHi,RdaLo] 1 <= shift <= 32	UQSHLL RdaLo,RdaHi,#shift	[RdaHi,RdaLo] -> result	MVE
uint64_t [__arm_]urshrl(uint64_t value, const int shift)	value -> [RdaHi,RdaLo] 1 <= shift <= 32	URSHRL RdaLo,RdaHi,#shift	[RdaHi,RdaLo] -> result	MVE
int64_t [__arm_]srshrl(int64_t value, const int shift)	value -> [RdaHi,RdaLo] 1 <= shift <= 32	SRSHRL RdaLo,RdaHi,#shift	[RdaHi,RdaLo] -> result	MVE

Intrinsic	Argument Preparation	Instruction	Result	Supported Architectures
int64_t [__arm_]sqshll(int64_t value, const int shift)	value -> [RdaHi,RdaLo] 1 <= shift <= 32	SQSHLL RdaLo,RdaHi,#shift	[RdaHi,RdaLo] -> result	MVE
uint32_t [__arm_]uqrshl(uint32_t value, int32_t shift)	value -> Rda shift -> Rm	UQRSHL Rda,Rm	Rda -> result	MVE
int32_t [__arm_]sqrshr(int32_t value, int32_t shift)	value -> Rda shift -> Rm	SQRSHR Rda,Rm	Rda -> result	MVE
uint32_t [__arm_]uqshl(uint32_t value, const int shift)	value -> Rda 1 <= shift <= 32	UQSHL Rda,#shift	Rda -> result	MVE
uint32_t [__arm_]urshr(uint32_t value, const int shift)	value -> Rda 1 <= shift <= 32	URSHR Rda,#shift	Rda -> result	MVE
int32_t [__arm_]sqshl(int32_t value, const int shift)	value -> Rda 1 <= shift <= 32	SQSHL Rda,#shift	Rda -> result	MVE
int32_t [__arm_]srshr(int32_t value, const int shift)	value -> Rda 1 <= shift <= 32	SRSHR Rda,#shift	Rda -> result	MVE