in_use	Opcode	Extended Name	Specific	variant
UNALLOCATED				
	BAD			
Branch,exception gener				
Compare _ Branch (imme				
	CBZ	32_bit		32_bit
	CBNZ	32_bit		32_bit
	CBZ	64_bit		64_bit
	CBNZ	64_bit		64_bit
Test & branch (immediate				
	TBZ			
	TBNZ			
Conditional branch (imme				
	B_cond			
Exception generation				
	SVC			
	HVC			
	SMC			
	BRK			
	HLT			
	DCPS1			
	DCPS2			
	DCPS3			
System				
	MSR	immediate	immediate	
	HINT			
	CLREX			
	DSB			
	DMB			
	ISB			
	SYS			
	MSR	register	register	
	SYSL			
	MRS			
Unconditional branch (reg				
	BR			
	BLR			
	RET			
	ERET			

1	in_use	Opcode	Extended Name	Specific	variant
40		DRPS			
41		**Unconditional branch (im**			
42		B			
43		BL			
44		**Loads and stores**			
45		**Load/store exclusive**			
46		STXRB			
47		STLXRB			
48		LDXRB			
49		LDAXRB			
50		STLRB			
51		LDARB			
52		STXRH			
53		STLXRH			
54		LDXRH			
55		LDAXRH			
56		STLRH			
57		LDARH			
58		STXR	32_bit		32_bit
59		STLXR	32_bit		32_bit
60		STXP	32_bit		32_bit
61		STLXP	32_bit		32_bit
62		LDXR	32_bit		32_bit
63		LDAXR	32_bit		32_bit
64		LDXP	32_bit		32_bit
65		LDAXP	32_bit		32_bit
66		STLR	32_bit		32_bit
67		LDAR	32_bit		32_bit
68		STXR	64_bit		64_bit
69		STLXR	64_bit		64_bit
70		STXP	64_bit		64_bit
71		STLXP	64_bit		64_bit
72		LDXR	64_bit		64_bit
73		LDAXR	64_bit		64_bit
74		LDXP	64_bit		64_bit
75		LDAXP	64_bit		64_bit
76		STLR	64_bit		64_bit
77		LDAR	64_bit		64_bit

1	in_use	Opcode	Extended Name	Specific		variant
78		**Load register (literal)**				
79		LDR	literal_32_bit	literal		32_bit
80		LDR	literal_SIMD_FP_32_bit	literal_SIMD_FP		32_bit
81		LDR	literal_64_bit	literal		64_bit
82		LDR	literal_SIMD_FP_64_bit	literal_SIMD_FP		64_bit
83		LDRSW	literal	literal		
84		LDR	literal_SIMD_FP_128_bit	literal_SIMD_FP		128_bit
85		PRFM	literal	literal		
86		**Load/store no-allocate pa**				
87		STNP	32_bit			32_bit
88		LDNP	32_bit			32_bit
89		STNP	SIMD_FP_32_bit	SIMD_FP		32_bit
90		LDNP	SIMD_FP_32_bit	SIMD_FP		32_bit
91		STNP	SIMD_FP_64_bit	SIMD_FP		64_bit
92		LDNP	SIMD_FP_64_bit	SIMD_FP		64_bit
93		STNP	64_bit			64_bit
94		LDNP	64_bit			64_bit
95		STNP	SIMD_FP_128_bit	SIMD_FP		128_bit
96		LDNP	SIMD_FP_128_bit	SIMD_FP		128_bit
97		**Load/store register pair (p**				
98		STP	1_32_bit		1	32_bit
99		LDP	1_32_bit		1	32_bit
100		STP	SIMD_FP_1_32_bit	SIMD_FP_1		32_bit
101		LDP	SIMD_FP_1_32_bit	SIMD_FP_1		32_bit
102		LDPSW	Post_index			Post_index
103		STP	SIMD_FP_1_64_bit	SIMD_FP_1		64_bit
104		LDP	SIMD_FP_1_64_bit	SIMD_FP_1		64_bit
105		STP	1_64_bit		1	64_bit
106		LDP	1_64_bit		1	64_bit
107		STP	SIMD_FP_1_128_bit	SIMD_FP_1		128_bit
108		LDP	SIMD_FP_1_128_bit	SIMD_FP_1		128_bit
109		**Load/store register pair (o**				
110		STP	2_32_bit		2	32_bit
111		LDP	2_32_bit		2	32_bit
112		STP	SIMD_FP_2_32_bit	SIMD_FP_2		32_bit

1	in_use	Opcode	Extended Name	Specific	variant
113		LDP	SIMD_FP_2_32_bit	SIMD_FP_2	32_bit
114		LDPSW	Signed_offset		Signed_offset
115		STP	SIMD_FP_2_64_bit	SIMD_FP_2	64_bit
116		LDP	SIMD_FP_2_64_bit	SIMD_FP_2	64_bit
117		STP	2_64_bit	2	64_bit
118		LDP	2_64_bit	2	64_bit
119		STP	SIMD_FP_2_128_bit	SIMD_FP_2	128_bit
120		LDP	SIMD_FP_2_128_bit	SIMD_FP_2	128_bit
121		**Load/store register pair (p**			
122		STP	3_32_bit	3	32_bit
123		LDP	3_32_bit	3	32_bit
124		STP	SIMD_FP_3_32_bit	SIMD_FP_3	32_bit
125		LDP	SIMD_FP_3_32_bit	SIMD_FP_3	32_bit
126		LDPSW	Pre_index		Pre_index
127		STP	SIMD_FP_3_64_bit	SIMD_FP_3	64_bit
128		LDP	SIMD_FP_3_64_bit	SIMD_FP_3	64_bit
129		STP	3_64_bit	3	64_bit
130		LDP	3_64_bit	3	64_bit
131		STP	SIMD_FP_3_128_bit	SIMD_FP_3	128_bit
132		LDP	SIMD_FP_3_128_bit	SIMD_FP_3	128_bit
133		**Load/store register (unsc**			
134		STURB			
135		LDURB			
136		LDURSB	64_bit		64_bit
137		LDURSB	32_bit		32_bit
138		STUR	SIMD_FP_8_bit	SIMD_FP	8_bit
139		LDUR	SIMD_FP_8_bit	SIMD_FP	8_bit
140		STUR	SIMD_FP_128_bit	SIMD_FP	128_bit
141		LDUR	SIMD_FP_128_bit	SIMD_FP	128_bit
142		STURH			
143		LDURH			
144		LDURSH	64_bit		64_bit
145		LDURSH	32_bit		32_bit
146		STUR	SIMD_FP_16_bit	SIMD_FP	16_bit
147		LDUR	SIMD_FP_16_bit	SIMD_FP	16_bit
148		STUR	32_bit		32_bit
149		LDUR	32_bit		32_bit
150		LDURSW			

1	in_use	Opcode	Extended Name	Specific	variant
151		STUR	SIMD_FP_32_bit	SIMD_FP	32_bit
152		LDUR	SIMD_FP_32_bit	SIMD_FP	32_bit
153		STUR	64_bit		64_bit
154		LDUR	64_bit		64_bit
155		PRFUM			
156		STUR	SIMD_FP_64_bit	SIMD_FP	64_bit
157		LDUR	SIMD_FP_64_bit	SIMD_FP	64_bit
158		**Load/store register (imme**			
159		STRB	immediate_Post_index	immediate	Post_index
160		LDRB	immediate_Post_index	immediate	Post_index
161		LDRSB	immediate_1_64_bit	immediate_1	64_bit
162		LDRSB	immediate_1_32_bit	immediate_1	32_bit
163		STR	immediate_SIMD_FP_1_8_bit	immediate_SIMD_FP_1	8_bit
164		LDR	immediate_SIMD_FP_1_8_bit	immediate_SIMD_FP_1	8_bit
165		STR	immediate_SIMD_FP_1_128_bit	immediate_SIMD_FP_1	128_bit
166		LDR	immediate_SIMD_FP_1_128_bit	immediate_SIMD_FP_1	128_bit
167		STRH	immediate_Post_index	immediate	Post_index
168		LDRH	immediate_Post_index	immediate	Post_index
169		LDRSH	immediate_1_64_bit	immediate_1	64_bit
170		LDRSH	immediate_1_32_bit	immediate_1	32_bit
171		STR	immediate_SIMD_FP_1_16_bit	immediate_SIMD_FP_1	16_bit
172		LDR	immediate_SIMD_FP_1_16_bit	immediate_SIMD_FP_1	16_bit
173		STR	immediate_1_32_bit	immediate_1	32_bit
174		LDR	immediate_1_32_bit	immediate_1	32_bit
175		LDRSW	immediate_Post_index	immediate	Post_index
176		STR	immediate_SIMD_FP_1_32_bit	immediate_SIMD_FP_1	32_bit
177		LDR	immediate_SIMD_FP_1_32_bit	immediate_SIMD_FP_1	32_bit
178		STR	immediate_1_64_bit	immediate_1	64_bit
179		LDR	immediate_1_64_bit	immediate_1	64_bit
180		STR	immediate_SIMD_FP_1_64_bit	immediate_SIMD_FP_1	64_bit
181		LDR	immediate_SIMD_FP_1_64_bit	immediate_SIMD_FP_1	64_bit
182		**Load/store register (unpri**			
183		STTRB			
184		LDTRB			
185		LDTRSB	64_bit		64_bit
186		LDTRSB	32_bit		32_bit
187		STTRH			
188		LDTRH			

1	in_use	Opcode	Extended Name	Specific	variant
189		LDTRSH	64_bit		64_bit
190		LDTRSH	32_bit		32_bit
191		STTR	32_bit		32_bit
192		LDTR	32_bit		32_bit
193		LDTRSW			
194		STTR	64_bit		64_bit
195		LDTR	64_bit		64_bit
196		**Load/store register (imme**			
197		STRB	immediate_Pre_index	immediate	Pre_index
198		LDRB	immediate_Pre_index	immediate	Pre_index
199		LDRSB	immediate_2_64_bit	immediate_2	64_bit
200		LDRSB	immediate_2_32_bit	immediate_2	32_bit
201		STR	immediate_SIMD_FP_2_8_bit	immediate_SIMD_FP_2	8_bit
202		LDR	immediate_SIMD_FP_2_8_bit	immediate_SIMD_FP_2	8_bit
203		STR	immediate_SIMD_FP_2_128_bit	immediate_SIMD_FP_2	128_bit
204		LDR	immediate_SIMD_FP_2_128_bit	immediate_SIMD_FP_2	128_bit
205		STRH	immediate_Pre_index	immediate	Pre_index
206		LDRH	immediate_Pre_index	immediate	Pre_index
207		LDRSH	immediate_2_64_bit	immediate_2	64_bit
208		LDRSH	immediate_2_32_bit	immediate_2	32_bit
209		STR	immediate_SIMD_FP_2_16_bit	immediate_SIMD_FP_2	16_bit
210		LDR	immediate_SIMD_FP_2_16_bit	immediate_SIMD_FP_2	16_bit
211		STR	immediate_2_32_bit	immediate_2	32_bit
212		LDR	immediate_2_32_bit	immediate_2	32_bit
213		LDRSW	immediate_Pre_index	immediate	Pre_index
214		STR	immediate_SIMD_FP_2_32_bit	immediate_SIMD_FP_2	32_bit
215		LDR	immediate_SIMD_FP_2_32_bit	immediate_SIMD_FP_2	32_bit
216		STR	immediate_2_64_bit	immediate_2	64_bit
217		LDR	immediate_2_64_bit	immediate_2	64_bit
218		STR	immediate_SIMD_FP_2_64_bit	immediate_SIMD_FP_2	64_bit
219		LDR	immediate_SIMD_FP_2_64_bit	immediate_SIMD_FP_2	64_bit
220		**Load/store register (regis**			
221		STRB	register	register	
222		LDRB	register	register	
223		LDRSB	register_64_bit	register	64_bit
224		LDRSB	register_32_bit	register	32_bit
225		STR	register_SIMD_FP_8_bit	register_SIMD_FP	8_bit
226		LDR	register_SIMD_FP_8_bit	register_SIMD_FP	8_bit

1	in_use	Opcode	Extended Name	Specific	variant
227		STR	register_SIMD_FP_128_bit	register_SIMD_FP	128_bit
228		LDR	register_SIMD_FP_128_bit	register_SIMD_FP	128_bit
229		STRH	register	register	
230		LDRH	register	register	
231		LDRSH	register_64_bit	register	64_bit
232		LDRSH	register_32_bit	register	32_bit
233		STR	register_SIMD_FP_16_bit	register_SIMD_FP	16_bit
234		LDR	register_SIMD_FP_16_bit	register_SIMD_FP	16_bit
235		STR	register_32_bit	register	32_bit
236		LDR	register_32_bit	register	32_bit
237		LDRSW	register	register	
238		STR	register_SIMD_FP_32_bit	register_SIMD_FP	32_bit
239		LDR	register_SIMD_FP_32_bit	register_SIMD_FP	32_bit
240		STR	register_64_bit	register	64_bit
241		LDR	register_64_bit	register	64_bit
242		PRFM	register	register	
243		STR	register_SIMD_FP_64_bit	register_SIMD_FP	64_bit
244		LDR	register_SIMD_FP_64_bit	register_SIMD_FP	64_bit
245		**Load/store register (unsig**			
246		STRB	immediate_Unsigned_offset	immediate	Unsigned_offs
247		LDRB	immediate_Unsigned_offset	immediate	Unsigned_offs
248		LDRSB	immediate_3_64_bit	immediate_3	64_bit
249		LDRSB	immediate_3_32_bit	immediate_3	32_bit
250		STR	immediate_SIMD_FP_8_bit	immediate_SIMD_FP	8_bit
251		LDR	immediate_SIMD_FP_8_bit	immediate_SIMD_FP	8_bit
252		STR	immediate_SIMD_FP_128_bit	immediate_SIMD_FP	128_bit
253		LDR	immediate_SIMD_FP_128_bit	immediate_SIMD_FP	128_bit
254		STRH	immediate_Unsigned_offset	immediate	Unsigned_offs
255		LDRH	immediate_Unsigned_offset	immediate	Unsigned_offs
256		LDRSH	immediate_3_64_bit	immediate_3	64_bit
257		LDRSH	immediate_3_32_bit	immediate_3	32_bit
258		STR	immediate_SIMD_FP_16_bit	immediate_SIMD_FP	16_bit
259		LDR	immediate_SIMD_FP_16_bit	immediate_SIMD_FP	16_bit
260		STR	immediate_3_32_bit	immediate_3	32_bit
261		LDR	immediate_3_32_bit	immediate_3	32_bit
262		LDRSW	immediate_Unsigned_offset	immediate	Unsigned_offs
263		STR	immediate_SIMD_FP_32_bit	immediate_SIMD_FP	32_bit
264		LDR	immediate_SIMD_FP_32_bit	immediate_SIMD_FP	32_bit

1	in_use	Opcode	Extended Name	Specific	variant
265		STR	immediate_3_64_bit	immediate_3	64_bit
266		LDR	immediate_3_64_bit	immediate_3	64_bit
267		PRFM	immediate	immediate	
268		STR	immediate_SIMD_FP_64_bit	immediate_SIMD_FP	64_bit
269		LDR	immediate_SIMD_FP_64_bit	immediate_SIMD_FP	64_bit
270		**Data processing – Imme**			
271		**PC-rel. addressing**			
272		ADR			
273		ADRP			
274		**Add/subtract (immediate)**			
275		ADD	immediate_32_bit	immediate	32_bit
276		ADDS	immediate_32_bit	immediate	32_bit
277		SUB	immediate_32_bit	immediate	32_bit
278		SUBS	immediate_32_bit	immediate	32_bit
279		ADD	immediate_64_bit	immediate	64_bit
280		ADDS	immediate_64_bit	immediate	64_bit
281		SUB	immediate_64_bit	immediate	64_bit
282		SUBS	immediate_64_bit	immediate	64_bit
283		**Logical (immediate)**			
284		AND	immediate_32_bit	immediate	32_bit
285		ORR	immediate_32_bit	immediate	32_bit
286		EOR	immediate_32_bit	immediate	32_bit
287		ANDS	immediate_32_bit	immediate	32_bit
288		AND	immediate_64_bit	immediate	64_bit
289		ORR	immediate_64_bit	immediate	64_bit
290		EOR	immediate_64_bit	immediate	64_bit
291		ANDS	immediate_64_bit	immediate	64_bit
292		**Move wide (immediate)**			
293		MOVN	32_bit		32_bit
294		MOVZ	32_bit		32_bit
295		MOVK	32_bit		32_bit
296		MOVN	64_bit		64_bit
297		MOVZ	64_bit		64_bit
298		MOVK	64_bit		64_bit
299		**Bitfield**			
300		SBFM	32_bit		32_bit
301		BFM	32_bit		32_bit
302		UBFM	32_bit		32_bit

1	in_use	Opcode	Extended Name	Specific	variant
303		SBFM	64_bit		64_bit
304		BFM	64_bit		64_bit
305		UBFM	64_bit		64_bit
306		**Extract**			
307		EXTR	32_bit		32_bit
308		EXTR	64_bit		64_bit
309		**Data Processing – regis**			
310		**Logical (shifted register)**			
311		AND	shifted_register_32_bit	shifted_register	32_bit
312		BIC	shifted_register_32_bit	shifted_register	32_bit
313		ORR	shifted_register_32_bit	shifted_register	32_bit
314		ORN	shifted_register_32_bit	shifted_register	32_bit
315		EOR	shifted_register_32_bit	shifted_register	32_bit
316		EON	shifted_register_32_bit	shifted_register	32_bit
317		ANDS	shifted_register_32_bit	shifted_register	32_bit
318		BICS	shifted_register_32_bit	shifted_register	32_bit
319		AND	shifted_register_64_bit	shifted_register	64_bit
320		BIC	shifted_register_64_bit	shifted_register	64_bit
321		ORR	shifted_register_64_bit	shifted_register	64_bit
322		ORN	shifted_register_64_bit	shifted_register	64_bit
323		EOR	shifted_register_64_bit	shifted_register	64_bit
324		EON	shifted_register_64_bit	shifted_register	64_bit
325		ANDS	shifted_register_64_bit	shifted_register	64_bit
326		BICS	shifted_register_64_bit	shifted_register	64_bit
327		**Add/subtract (shifted regi**			
328		ADD	shifted_register_32_bit	shifted_register	32_bit
329		ADDS	shifted_register_32_bit	shifted_register	32_bit
330		SUB	shifted_register_32_bit	shifted_register	32_bit
331		SUBS	shifted_register_32_bit	shifted_register	32_bit
332		ADD	shifted_register_64_bit	shifted_register	64_bit
333		ADDS	shifted_register_64_bit	shifted_register	64_bit
334		SUB	shifted_register_64_bit	shifted_register	64_bit
335		SUBS	shifted_register_64_bit	shifted_register	64_bit
336		**Add/subtract (extended re**			
337		ADD	extended_register_32_bit	extended_register	32_bit
338		ADDS	extended_register_32_bit	extended_register	32_bit
339		SUB	extended_register_32_bit	extended_register	32_bit
340		SUBS	extended_register_32_bit	extended_register	32_bit

1	in_use	Opcode	Extended Name	Specific	variant
341		ADD	extended_register_64_bit	extended_register	64_bit
342		ADDS	extended_register_64_bit	extended_register	64_bit
343		SUB	extended_register_64_bit	extended_register	64_bit
344		SUBS	extended_register_64_bit	extended_register	64_bit
345		**Add/subtract (with carry)**			
346		ADC	32_bit		32_bit
347		ADCS	32_bit		32_bit
348		SBC	32_bit		32_bit
349		SBCS	32_bit		32_bit
350		ADC	64_bit		64_bit
351		ADCS	64_bit		64_bit
352		SBC	64_bit		64_bit
353		SBCS	64_bit		64_bit
354		**Conditional compare (reg**			
355		CCMN	register_32_bit	register	32_bit
356		CCMN	register_64_bit	register	64_bit
357		CCMP	register_32_bit	register	32_bit
358		CCMP	register_64_bit	register	64_bit
359		**Conditional compare (imr**			
360		CCMN	immediate_32_bit	immediate	32_bit
361		CCMN	immediate_64_bit	immediate	64_bit
362		CCMP	immediate_32_bit	immediate	32_bit
363		CCMP	immediate_64_bit	immediate	64_bit
364		**Conditional select**			
365		CSEL	32_bit		32_bit
366		CSINC	32_bit		32_bit
367		CSINV	32_bit		32_bit
368		CSNEG	32_bit		32_bit
369		CSEL	64_bit		64_bit
370		CSINC	64_bit		64_bit
371		CSINV	64_bit		64_bit
372		CSNEG	64_bit		64_bit
373		**Data-processing (3 sourc**			
374		MADD	32_bit		32_bit
375		MADD	64_bit		64_bit
376		SMADDL			
377		UMADDL			
378		MSUB	32_bit		32_bit

1	in_use	Opcode	Extended Name	Specific	variant
379		MSUB	64_bit		64_bit
380		SMSUBL			
381		UMSUBL			
382		SMULH			
383		UMULH			
384		**Data-processing (2 sourc**			
385		CRC32X			
386		CRC32CX			
387		CRC32B			
388		CRC32CB			
389		CRC32H			
390		CRC32CH			
391		CRC32W			
392		CRC32CW			
393		UDIV	32_bit		32_bit
394		UDIV	64_bit		64_bit
395		SDIV	32_bit		32_bit
396		SDIV	64_bit		64_bit
397		LSLV	32_bit		32_bit
398		LSLV	64_bit		64_bit
399		LSRV	32_bit		32_bit
400		LSRV	64_bit		64_bit
401		ASRV	32_bit		32_bit
402		ASRV	64_bit		64_bit
403		RORV	32_bit		32_bit
404		RORV	64_bit		64_bit
405		**Data-processing (1 sourc**			
406		RBIT	32_bit		32_bit
407		RBIT	64_bit		64_bit
408		CLZ	32_bit		32_bit
409		CLZ	64_bit		64_bit
410		CLS	32_bit		32_bit
411		CLS	64_bit		64_bit
412		REV	32_bit		32_bit
413		REV	64_bit		64_bit
414		REV16	64_bit		64_bit
415		REV16	32_bit		32_bit
416		REV32			

1	in_use	Opcode	Extended Name	Specific	variant
417	//	**Data Processing – SIMD**			
418	//	**Floating-point<->fixed-po**			
419	//	SCVTF	scalar_fixed_point_32_bit_to_single_pre	scalar_fixed_point	32_bit_to_sin
420	//	UCVTF	scalar_fixed_point_32_bit_to_single_pre	scalar_fixed_point	32_bit_to_sin
421	//	FCVTZS	scalar_fixed_point_Single_precision_to_	scalar_fixed_point	Single_precisi
422	//	FCVTZU	scalar_fixed_point_Single_precision_to_	scalar_fixed_point	Single_precisi
423	//	SCVTF	scalar_fixed_point_32_bit_to_double_pre	scalar_fixed_point	32_bit_to_dou
424	//	UCVTF	scalar_fixed_point_32_bit_to_double_pre	scalar_fixed_point	32_bit_to_dou
425	//	FCVTZS	scalar_fixed_point_Double_precision_to_	scalar_fixed_point	Double_precis
426	//	FCVTZU	scalar_fixed_point_Double_precision_to_	scalar_fixed_point	Double_precis
427	//	SCVTF	scalar_fixed_point_64_bit_to_single_pre	scalar_fixed_point	64_bit_to_sin
428	//	UCVTF	scalar_fixed_point_64_bit_to_single_pre	scalar_fixed_point	64_bit_to_sin
429	//	FCVTZS	scalar_fixed_point_Single_precision_to_	scalar_fixed_point	Single_precisi
430	//	FCVTZU	scalar_fixed_point_Single_precision_to_	scalar_fixed_point	Single_precisi
431	//	SCVTF	scalar_fixed_point_64_bit_to_double_pre	scalar_fixed_point	64_bit_to_dou
432	//	UCVTF	scalar_fixed_point_64_bit_to_double_pre	scalar_fixed_point	64_bit_to_dou
433	//	FCVTZS	scalar_fixed_point_Double_precision_to_	scalar_fixed_point	Double_precis
434	//	FCVTZU	scalar_fixed_point_Double_precision_to_	scalar_fixed_point	Double_precis
435	//	**Floating-point conditional**			
436	//	FCCMP	Single_precision		Single_precisi
437	//	FCCMPE	Single_precision		Single_precisi
438	//	FCCMP	Double_precision		Double_precis
439	//	FCCMPE	Double_precision		Double_precis
440	//	**Floating-point data-proce**			
441	//	FMUL	scalar_Single_precision	scalar	Single_precisi
442	//	FDIV	scalar_Single_precision	scalar	Single_precisi
443	//	FADD	scalar_Single_precision	scalar	Single_precisi
444	//	FSUB	scalar_Single_precision	scalar	Single_precisi
445	//	FMAX	scalar_Single_precision	scalar	Single_precisi
446	//	FMIN	scalar_Single_precision	scalar	Single_precisi
447	//	FMAXNM	scalar_Single_precision	scalar	Single_precisi
448	//	FMINNM	scalar_Single_precision	scalar	Single_precisi
449	//	FNMUL	Single_precision		Single_precisi
450	//	FMUL	scalar_Double_precision	scalar	Double_precis

1	in_use	Opcode	Extended Name	Specific	variant
451	//	FDIV	scalar_Double_precision	scalar	Double_precis
452	//	FADD	scalar_Double_precision	scalar	Double_precis
453	//	FSUB	scalar_Double_precision	scalar	Double_precis
454	//	FMAX	scalar_Double_precision	scalar	Double_precis
455	//	FMIN	scalar_Double_precision	scalar	Double_precis
456	//	FMAXNM	scalar_Double_precision	scalar	Double_precis
457	//	FMINNM	scalar_Double_precision	scalar	Double_precis
458	//	FNMUL	Double_precision		Double_precis
459	//	**Floating-point conditional**			
460	//	FCSEL	Single_precision		Single_precisi
461	//	FCSEL	Double_precision		Double_precis
462	//	**Floating-point immediate**			
463	//	FMOV	scalar_immediate_Single_precision	scalar_immediate	Single_precisi
464	//	FMOV	scalar_immediate_Double_precision	scalar_immediate	Double_precis
465	//	**Floating-point compare**			
466	//	FCMP	Single_precision		Single_precisi
467	//	FCMP	Single_precision_zero		Single_precisi
468	//	FCMPE	Single_precision		Single_precisi
469	//	FCMPE	Single_precision_zero		Single_precisi
470	//	FCMP	Double_precision		Double_precis
471	//	FCMP	Double_precision_zero		Double_precis
472	//	FCMPE	Double_precision		Double_precis
473	//	FCMPE	Double_precision_zero		Double_precis
474	//	**Floating-point data-proce**			
475	//	FMOV	register_Single_precision	register	Single_precisi
476	//	FABS	scalar_Single_precision	scalar	Single_precisi
477	//	FNEG	scalar_Single_precision	scalar	Single_precisi
478	//	FSQRT	scalar_Single_precision	scalar	Single_precisi
479	//	FCVT	Single_precision_to_double_precision		Single_precisi
480	//	FCVT	Single_precision_to_half_precision		Single_precisi
481	//	FRINTN	scalar_Single_precision	scalar	Single_precisi
482	//	FRINTP	scalar_Single_precision	scalar	Single_precisi
483	//	FRINTM	scalar_Single_precision	scalar	Single_precisi
484	//	FRINTZ	scalar_Single_precision	scalar	Single_precisi

1	in_use	Opcode	Extended Name	Specific	variant
485	//	FRINTA	scalar_Single_precision	scalar	Single_precisi
486	//	FRINTX	scalar_Single_precision	scalar	Single_precisi
487	//	FRINTI	scalar_Single_precision	scalar	Single_precisi
488	//	FMOV	register_Double_precision	register	Double_precis
489	//	FABS	scalar_Double_precision	scalar	Double_precis
490	//	FNEG	scalar_Double_precision	scalar	Double_precis
491	//	FSQRT	scalar_Double_precision	scalar	Double_precis
492	//	FCVT	Double_precision_to_single_precision		Double_precis
493	//	FCVT	Double_precision_to_half_precision		Double_precis
494	//	FRINTN	scalar_Double_precision	scalar	Double_precis
495	//	FRINTP	scalar_Double_precision	scalar	Double_precis
496	//	FRINTM	scalar_Double_precision	scalar	Double_precis
497	//	FRINTZ	scalar_Double_precision	scalar	Double_precis
498	//	FRINTA	scalar_Double_precision	scalar	Double_precis
499	//	FRINTX	scalar_Double_precision	scalar	Double_precis
500	//	FRINTI	scalar_Double_precision	scalar	Double_precis
501	//	FCVT	Half_precision_to_single_precision		Half_precisior
502	//	FCVT	Half_precision_to_double_precision		Half_precisior
503	//	**Floating-point<->integer c**			
504	//	FCVTNS	scalar_Single_precision_to_32_bit	scalar	Single_precisi
505	//	FCVTNU	scalar_Single_precision_to_32_bit	scalar	Single_precisi
506	//	SCVTF	scalar_integer_32_bit_to_single_precisio	scalar_integer	32_bit_to_sing
507	//	UCVTF	scalar_integer_32_bit_to_single_precisio	scalar_integer	32_bit_to_sing
508	//	FCVTAS	scalar_Single_precision_to_32_bit	scalar	Single_precisi
509	//	FCVTAU	scalar_Single_precision_to_32_bit	scalar	Single_precisi
510	//	FMOV	general_Single_precision_to_32_bit	general	Single_precisi
511	//	FMOV	general_32_bit_to_single_precision	general	32_bit_to_sing
512	//	FCVTPS	scalar_Single_precision_to_32_bit	scalar	Single_precisi
513	//	FCVTPU	scalar_Single_precision_to_32_bit	scalar	Single_precisi
514	//	FCVTMS	scalar_Single_precision_to_32_bit	scalar	Single_precisi
515	//	FCVTMU	scalar_Single_precision_to_32_bit	scalar	Single_precisi
516	//	FCVTZS	scalar_integer_Single_precision_to_32_b	scalar_integer	Single_precisi
517	//	FCVTZU	scalar_integer_Single_precision_to_32_b	scalar_integer	Single_precisi
518	//	FCVTNS	scalar_Double_precision_to_32_bit	scalar	Double_precis

1	in_use	Opcode	Extended Name	Specific	variant
519	//	FCVTNU	scalar_Double_precision_to_32_bit	scalar	Double_precis
520	//	SCVTF	scalar_integer_32_bit_to_double_precisi	scalar_integer	32_bit_to_dou
521	//	UCVTF	scalar_integer_32_bit_to_double_precisi	scalar_integer	32_bit_to_dou
522	//	FCVTAS	scalar_Double_precision_to_32_bit	scalar	Double_precis
523	//	FCVTAU	scalar_Double_precision_to_32_bit	scalar	Double_precis
524	//	FCVTPS	scalar_Double_precision_to_32_bit	scalar	Double_precis
525	//	FCVTPU	scalar_Double_precision_to_32_bit	scalar	Double_precis
526	//	FCVTMS	scalar_Double_precision_to_32_bit	scalar	Double_precis
527	//	FCVTMU	scalar_Double_precision_to_32_bit	scalar	Double_precis
528	//	FCVTZS	scalar_integer_Double_precision_to_32_	scalar_integer	Double_precis
529	//	FCVTZU	scalar_integer_Double_precision_to_32_	scalar_integer	Double_precis
530	//	FCVTNS	scalar_Single_precision_to_64_bit	scalar	Single_precisi
531	//	FCVTNU	scalar_Single_precision_to_64_bit	scalar	Single_precisi
532	//	SCVTF	scalar_integer_64_bit_to_single_precisio	scalar_integer	64_bit_to_sing
533	//	UCVTF	scalar_integer_64_bit_to_single_precisio	scalar_integer	64_bit_to_sing
534	//	FCVTAS	scalar_Single_precision_to_64_bit	scalar	Single_precisi
535	//	FCVTAU	scalar_Single_precision_to_64_bit	scalar	Single_precisi
536	//	FCVTPS	scalar_Single_precision_to_64_bit	scalar	Single_precisi
537	//	FCVTPU	scalar_Single_precision_to_64_bit	scalar	Single_precisi
538	//	FCVTMS	scalar_Single_precision_to_64_bit	scalar	Single_precisi
539	//	FCVTMU	scalar_Single_precision_to_64_bit	scalar	Single_precisi
540	//	FCVTZS	scalar_integer_Single_precision_to_64_b	scalar_integer	Single_precisi
541	//	FCVTZU	scalar_integer_Single_precision_to_64_b	scalar_integer	Single_precisi
542	//	FCVTNS	scalar_Double_precision_to_64_bit	scalar	Double_precis
543	//	FCVTNU	scalar_Double_precision_to_64_bit	scalar	Double_precis
544	//	SCVTF	scalar_integer_64_bit_to_double_precisi	scalar_integer	64_bit_to_dou
545	//	UCVTF	scalar_integer_64_bit_to_double_precisi	scalar_integer	64_bit_to_dou
546	//	FCVTAS	scalar_Double_precision_to_64_bit	scalar	Double_precis
547	//	FCVTAU	scalar_Double_precision_to_64_bit	scalar	Double_precis
548	//	FMOV	general_Double_precision_to_64_bit	general	Double_precis
549	//	FMOV	general_64_bit_to_double_precision	general	64_bit_to_dou
550	//	FCVTPS	scalar_Double_precision_to_64_bit	scalar	Double_precis
551	//	FCVTPU	scalar_Double_precision_to_64_bit	scalar	Double_precis
552	//	FCVTMS	scalar_Double_precision_to_64_bit	scalar	Double_precis

1	in_use	Opcode	Extended Name	Specific	variant
553	//	FCVTMU	scalar_Double_precision_to_64_bit	scalar	Double_precis
554	//	FCVTZS	scalar_integer_Double_precision_to_64_	scalar_integer	Double_precis
555	//	FCVTZU	scalar_integer_Double_precision_to_64_	scalar_integer	Double_precis
556	//	FMOV	general_Top_half_of_128_bit_to_64_bit	general	Top_half_of_1
557	//	FMOV	general_64_bit_to_top_half_of_128_bit	general	64_bit_to_top
558	//	**Floating-point data-proce**			
559	//	FMADD	Single_precision		Single_precisi
560	//	FMSUB	Single_precision		Single_precisi
561	//	FNMADD	Single_precision		Single_precisi
562	//	FNMSUB	Single_precision		Single_precisi
563	//	FMADD	Double_precision		Double_precis
564	//	FMSUB	Double_precision		Double_precis
565	//	FNMADD	Double_precision		Double_precis
566	//	FNMSUB	Double_precision		Double_precis
567	//	**AdvSIMD scalar three san**			
568	//	SQADD	Scalar		Scalar
569	//	SQSUB	Scalar		Scalar
570	//	CMGT	register_Scalar	register	Scalar
571	//	CMGE	register_Scalar	register	Scalar
572	//	SSHL	Scalar		Scalar
573	//	SQSHL	register_Scalar	register	Scalar
574	//	SRSHL	Scalar		Scalar
575	//	SQRSHL	Scalar		Scalar
576	//	ADD	vector_Scalar	vector	Scalar
577	//	CMTST	Scalar		Scalar
578	//	SQDMULH	vector_Scalar	vector	Scalar
579	//	FMULX	Scalar		Scalar
580	//	FCMEQ	register_Scalar	register	Scalar
581	//	FRECPS	Scalar		Scalar
582	//	FRSQRTS	Scalar		Scalar
583	//	UQADD	Scalar		Scalar
584	//	UQSUB	Scalar		Scalar
585	//	CMHI	register_Scalar	register	Scalar
586	//	CMHS	register_Scalar	register	Scalar

1	in_use	Opcode	Extended Name	Specific	variant
587	//	USHL	Scalar		Scalar
588	//	UQSHL	register_Scalar	register	Scalar
589	//	URSHL	Scalar		Scalar
590	//	UQRSHL	Scalar		Scalar
591	//	SUB	vector_Scalar	vector	Scalar
592	//	CMEQ	register_Scalar	register	Scalar
593	//	SQRDMULH	vector_Scalar	vector	Scalar
594	//	FCMGE	register_Scalar	register	Scalar
595	//	FACGE	Scalar		Scalar
596	//	FABD	Scalar		Scalar
597	//	FCMGT	register_Scalar	register	Scalar
598	//	FACGT	Scalar		Scalar
599	//	**AdvSIMD scalar three diff**			
600	//	SQDMLAL	vector_Scalar	vector	Scalar
601	//	SQDMLAL2	vector_Scalar	vector	Scalar
602	//	SQDMLSL	vector_Scalar	vector	Scalar
603	//	SQDMLSL2	vector_Scalar	vector	Scalar
604	//	SQDMULL	vector_Scalar	vector	Scalar
605	//	SQDMULL2	vector_Scalar	vector	Scalar
606	//	**AdvSIMD scalar two-reg r**			
607	//	SUQADD	Scalar		Scalar
608	//	SQABS	Scalar		Scalar
609	//	CMGT	zero_Scalar	zero	Scalar
610	//	CMEQ	zero_Scalar	zero	Scalar
611	//	CMLT	zero_Scalar	zero	Scalar
612	//	ABS	Scalar		Scalar
613	//	SQXTN	Scalar		Scalar
614	//	SQXTN2	Scalar		Scalar
615	//	FCVTNS	vector_Scalar	vector	Scalar
616	//	FCVTMS	vector_Scalar	vector	Scalar
617	//	FCVTAS	vector_Scalar	vector	Scalar
618	//	SCVTF	vector_integer_Scalar	vector_integer	Scalar
619	//	FCMGT	zero_Scalar	zero	Scalar
620	//	FCMEQ	zero_Scalar	zero	Scalar

1	in_use	Opcode	Extended Name	Specific	variant
621	//	FCMLT	zero_Scalar	zero	Scalar
622	//	FCVTPS	vector_Scalar	vector	Scalar
623	//	FCVTZS	vector_integer_Scalar	vector_integer	Scalar
624	//	FRECPE	Scalar		Scalar
625	//	FRECPX			
626	//	USQADD	Scalar		Scalar
627	//	SQNEG	Scalar		Scalar
628	//	CMGE	zero_Scalar	zero	Scalar
629	//	CMLE	zero_Scalar	zero	Scalar
630	//	NEG	vector_Scalar	vector	Scalar
631	//	SQXTUN	Scalar		Scalar
632	//	SQXTUN2	Scalar		Scalar
633	//	UQXTN	Scalar		Scalar
634	//	UQXTN2	Scalar		Scalar
635	//	FCVTXN	Scalar		Scalar
636	//	FCVTXN2	Scalar		Scalar
637	//	FCVTNU	vector_Scalar	vector	Scalar
638	//	FCVTMU	vector_Scalar	vector	Scalar
639	//	FCVTAU	vector_Scalar	vector	Scalar
640	//	UCVTF	vector_integer_Scalar	vector_integer	Scalar
641	//	FCMGE	zero_Scalar	zero	Scalar
642	//	FCMLE	zero_Scalar	zero	Scalar
643	//	FCVTPU	vector_Scalar	vector	Scalar
644	//	FCVTZU	vector_integer_Scalar	vector_integer	Scalar
645	//	FRSQRTE	Scalar		Scalar
646	//	**AdvSIMD scalar pairwise**			
647	//	ADDP	scalar	scalar	
648	//	FMAXNMP	scalar	scalar	
649	//	FADDP	scalar	scalar	
650	//	FMAXP	scalar	scalar	
651	//	FMINNMP	scalar	scalar	
652	//	FMINP	scalar	scalar	
653	//	**AdvSIMD scalar copy**			
654	//	DUP	element_Scalar	element	Scalar

1	in_use	Opcode	Extended Name	Specific	variant
655	//	**AdvSIMD scalar x indexe**			
656	//	SQDMLAL	by_element_Scalar	by_element	Scalar
657	//	SQDMLAL2	by_element_Scalar	by_element	Scalar
658	//	SQDMLSL	by_element_Scalar	by_element	Scalar
659	//	SQDMLSL2	by_element_Scalar	by_element	Scalar
660	//	SQDMULL	by_element_Scalar	by_element	Scalar
661	//	SQDMULL2	by_element_Scalar	by_element	Scalar
662	//	SQDMULH	by_element_Scalar	by_element	Scalar
663	//	SQRDMULH	by_element_Scalar	by_element	Scalar
664	//	FMLA	by_element_Scalar	by_element	Scalar
665	//	FMLS	by_element_Scalar	by_element	Scalar
666	//	FMUL	by_element_Scalar	by_element	Scalar
667	//	FMULX	by_element_Scalar	by_element	Scalar
668	//	**AdvSIMD scalar shift by i**			
669	//	SSHR	Scalar		Scalar
670	//	SSRA	Scalar		Scalar
671	//	SRSHR	Scalar		Scalar
672	//	SRSRA	Scalar		Scalar
673	//	SHL	Scalar		Scalar
674	//	SQSHL	immediate_Scalar	immediate	Scalar
675	//	SQSHRN	Scalar		Scalar
676	//	SQSHRN2	Scalar		Scalar
677	//	SQRSHRN	Scalar		Scalar
678	//	SQRSHRN2	Scalar		Scalar
679	//	SCVTF	vector_fixed_point_Scalar	vector_fixed_point	Scalar
680	//	FCVTZS	vector_fixed_point_Scalar	vector_fixed_point	Scalar
681	//	USHR	Scalar		Scalar
682	//	USRA	Scalar		Scalar
683	//	URSHR	Scalar		Scalar
684	//	URSRA	Scalar		Scalar
685	//	SRI	Scalar		Scalar
686	//	SLI	Scalar		Scalar
687	//	SQSHLU	Scalar		Scalar
688	//	UQSHL	immediate_Scalar	immediate	Scalar

1	in_use	Opcode	Extended Name	Specific	variant
689	//	SQSHRUN	Scalar		Scalar
690	//	SQSHRUN2	Scalar		Scalar
691	//	SQRSHRUN	Scalar		Scalar
692	//	SQRSHRUN2	Scalar		Scalar
693	//	UQSHRN	Scalar		Scalar
694	//	UQRSHRN	Scalar		Scalar
695	//	UQRSHRN2	Scalar		Scalar
696	//	UCVTF	vector_fixed_point_Scalar	vector_fixed_point	Scalar
697	//	FCVTZU	vector_fixed_point_Scalar	vector_fixed_point	Scalar
698	//	**Crypto three-reg SHA**			
699	//	SHA1C			
700	//	SHA1P			
701	//	SHA1M			
702	//	SHA1SU0			
703	//	SHA256H			
704	//	SHA256H2			
705	//	SHA256SU1			
706	//	**Crypto two-reg SHA**			
707	//	SHA1H			
708	//	SHA1SU1			
709	//	SHA256SU0			
710	//	**Crypto AES**			
711	//	AESE			
712	//	AESD			
713	//	AESMC			
714	//	AESIMC			
715	//	**AdvSIMD three same**			
716	//	SHADD			
717	//	SQADD	Vector		Vector
718	//	SRHADD			
719	//	SHSUB			
720	//	SQSUB	Vector		Vector
721	//	CMGT	register_Vector	register	Vector
722	//	CMGE	register_Vector	register	Vector

1	in_use	Opcode	Extended Name	Specific	variant
723	//	SSHL Vector			
724	//	SQSHL	register_Vector	register	Vector
725	//	SRSHL	Vector		Vector
726	//	SQRSHL	Vector		Vector
727	//	SMAX			
728	//	SMIN			
729	//	SABD			
730	//	SABA			
731	//	ADD	vector_Vector	vector	Vector
732	//	CMTST	Vector		Vector
733	//	MLA	vector	vector	
734	//	MUL	vector	vector	
735	//	SMAXP			
736	//	SMINP			
737	//	SQDMULH	vector_Vector	vector	Vector
738	//	ADDP	vector	vector	
739	//	FMAXNM	vector	vector	
740	//	FMLA	vector	vector	
741	//	FADD	vector	vector	
742	//	FMULX	Vector		Vector
743	//	FCMEQ	register_Vector	register	Vector
744	//	FMAX	vector	vector	
745	//	FRECPS	Vector		Vector
746	//	AND	vector	vector	
747	//	BIC	vector_register	vector_register	
748	//	FMINNM	vector	vector	
749	//	FMLS	vector	vector	
750	//	FSUB	vector	vector	
751	//	FMIN	vector	vector	
752	//	FRSQRTS	Vector		Vector
753	//	ORR	vector_register	vector_register	
754	//	ORN	vector	vector	
755	//	UHADD			
756	//	UQADD	Vector		Vector

1	in_use	Opcode	Extended Name	Specific	variant
757	//	URHADD			
758	//	UHSUB			
759	//		Vector		Vector
760	//	CMHI	register_Vector	register	Vector
761	//	CMHS	register_Vector	register	Vector
762	//	USHL	Vector		Vector
763	//	UQSHL	register_Vector	register	Vector
764	//	URSHL	Vector		Vector
765	//	UQRSHL	Vector		Vector
766	//	UMAX			
767	//	UMIN			
768	//	UABD			
769	//	UABA			
770	//	SUB	vector_Vector	vector	Vector
771	//	CMEQ	register_Vector	register	Vector
772	//	MLS	vector	vector	
773	//	PMUL			
774	//	UMAXP			
775	//	UMINP			
776	//	SQRDMULH	vector_Vector	vector	Vector
777	//	FMAXNMP	vector	vector	
778	//	FADDP	vector	vector	
779	//	FMUL	vector	vector	
780	//	FCMGE	register_Vector	register	Vector
781	//	FACGE	Vector		Vector
782	//	FMAXP	vector	vector	
783	//	FDIV	vector	vector	
784	//	EOR	vector	vector	
785	//	BSL			
786	//	FMINNMP	vector	vector	
787	//	FABD	Vector		Vector
788	//	FCMGT	register_Vector	register	Vector
789	//	FACGT	Vector		Vector
790	//	FMINP	vector	vector	

1	in_use	Opcode	Extended Name	Specific	variant
791	//	BIT			
792	//	BIF			
793	//	**AdvSIMD three different**			
794	//	SADDL			
795	//	SADDL2			
796	//	SADDW			
797	//	SADDW2			
798	//	SSUBL			
799	//	SSUBL2			
800	//	SSUBW			
801	//	SSUBW2			
802	//	ADDHN			
803	//	ADDHN2			
804	//	SABAL			
805	//	SABAL2			
806	//	SUBHN			
807	//	SUBHN2			
808	//	SABDL			
809	//	SABDL2			
810	//	SMLAL	vector	vector	
811	//	SMLAL2	vector	vector	
812	//	SQDMLAL	vector_Vector	vector	Vector
813	//	SQDMLAL2	vector_Vector	vector	Vector
814	//	SMLSL	vector	vector	
815	//	SMLSL2	vector	vector	
816	//	SQDMLSL	vector_Vector	vector	Vector
817	//	SQDMLSL2	vector_Vector	vector	Vector
818	//	SMULL	vector	vector	
819	//	SMULL2	vector	vector	
820	//	SQDMULL	vector_Vector	vector	Vector
821	//	SQDMULL2	vector_Vector	vector	Vector
822	//	PMULL			
823	//	PMULL2			
824	//	UADDL			

1	in_use	Opcode	Extended Name	Specific	variant
825	//	UADDL2			
826	//	UADDW			
827	//	UADDW2			
828	//	USUBL			
829	//	USUBL2			
830	//	USUBW			
831	//	USUBW2			
832	//	RADDHN			
833	//	RADDHN2			
834	//	UABAL			
835	//	UABAL2			
836	//	RSUBHN			
837	//	RSUBHN2			
838	//	UABDL			
839	//	UABDL2			
840	//	UMLAL	vector	vector	
841	//	UMLAL2	vector	vector	
842	//	UMLSL	vector	vector	
843	//	UMLSL2	vector	vector	
844	//	UMULL	vector	vector	
845	//	UMULL2	vector	vector	
846	//	**AdvSIMD two-reg misc**			
847	//	REV64			
848	//	REV16	vector	vector	
849	//	SADDLP			
850	//	SUQADD	Vector		Vector
851	//	CLS	vector	vector	
852	//	CNT			
853	//	SADALP			
854	//	SQABS	Vector		Vector
855	//	CMGT	zero_Vector	zero	Vector
856	//	CMEQ	zero_Vector	zero	Vector
857	//	CMLT	zero_Vector	zero	Vector
858	//	ABS	Vector		Vector

1	in_use	Opcode	Extended Name	Specific	variant
859	//	XTN			
860	//	XTN2			
861	//	SQXTN	Vector		Vector
862	//	SQXTN2	Vector		Vector
863	//	FCVTN			
864	//	FCVTN2			
865	//	FCVTL			
866	//	FCVTL2			
867	//	FRINTN	vector	vector	
868	//	FRINTM	vector	vector	
869	//	FCVTNS	vector_Vector	vector	Vector
870	//	FCVTMS	vector_Vector	vector	Vector
871	//	FCVTAS	vector_Vector	vector	Vector
872	//	SCVTF	vector_integer_Vector	vector_integer	Vector
873	//	FCMGT	zero_Vector	zero	Vector
874	//	FCMEQ	zero_Vector	zero	Vector
875	//	FCMLT	zero_Vector	zero	Vector
876	//	FABS	vector	vector	
877	//	FRINTP	vector	vector	
878	//	FRINTZ	vector	vector	
879	//	FCVTPS	vector_Vector	vector	Vector
880	//	FCVTZS	vector_integer_Vector	vector_integer	Vector
881	//	URECPE			
882	//	FRECPE	Vector		Vector
883	//	REV32	vector	vector	
884	//	UADDLP			
885	//	USQADD	Vector		Vector
886	//	CLZ	vector	vector	
887	//	UADALP			
888	//	SQNEG	Vector		Vector
889	//	CMGE	zero_Vector	zero	Vector
890	//	CMLE	zero_Vector	zero	Vector
891	//	NEG	vector_Vector	vector	Vector
892	//	SQXTUN	Vector		Vector

1	in_use	Opcode	Extended Name	Specific	variant
893	//	SQXTUN2	Vector		Vector
894	//	SHLL			
895	//	SHLL2			
896	//	UQXTN	Vector		Vector
897	//	UQXTN2	Vector		Vector
898	//	FCVTXN	Vector		Vector
899	//	FCVTXN2	Vector		Vector
900	//	FRINTA	vector	vector	
901	//	FRINTX	vector	vector	
902	//	FCVTNU	vector_Vector	vector	Vector
903	//	FCVTMU	vector_Vector	vector	Vector
904	//	FCVTAU	vector_Vector	vector	Vector
905	//	UCVTF	vector_integer_Vector	vector_integer	Vector
906	//	NOT			
907	//	RBIT	vector	vector	
908	//	FCMGE	zero_Vector	zero	Vector
909	//	FCMLE	zero_Vector	zero	Vector
910	//	FNEG	vector	vector	
911	//	FRINTI	vector	vector	
912	//	FCVTPU	vector_Vector	vector	Vector
913	//	FCVTZU	vector_integer_Vector	vector_integer	Vector
914	//	URSQRTE			
915	//	FRSQRTE	Vector		Vector
916	//	FSQRT	vector	vector	
917	//	**AdvSIMD across lanes**			
918	//	SADDLV			
919	//	SMAXV			
920	//	SMINV			
921	//	ADDV			
922	//	UADDLV			
923	//	UMAXV			
924	//	UMINV			
925	//	FMAXNMV			
926	//	FMAXV			

1	in_use	Opcode	Extended Name	Specific	variant
927	//	FMINNMV			
928	//	FMINV			
929	//	**AdvSIMD copy**			
930	//	DUP	element_Vector	element	Vector
931	//	DUP	general	general	
932	//	SMOV	32_bit		32_bit
933	//	UMOV	32_bit		32_bit
934	//	INS	general	general	
935	//	SMOV	64_bit		64_bit
936	//	UMOV	64_bit		64_bit
937	//	INS	element	element	
938	//	**AdvSIMD vector x indexe**			
939	//	SMLAL	by_element	by_element	
940	//	SMLAL2	by_element	by_element	
941	//	SQDMLAL	by_element_Vector	by_element	Vector
942	//	SQDMLAL2	by_element_Vector	by_element	Vector
943	//	SMLSL	by_element	by_element	
944	//	SMLSL2	by_element	by_element	
945	//	SQDMLSL	by_element_Vector	by_element	Vector
946	//	SQDMLSL2	by_element_Vector	by_element	Vector
947	//	MUL	by_element	by_element	
948	//	SMULL	by_element	by_element	
949	//	SMULL2	by_element	by_element	
950	//	SQDMULL	by_element_Vector	by_element	Vector
951	//	SQDMULL2	by_element_Vector	by_element	Vector
952	//	SQDMULH	by_element_Vector	by_element	Vector
953	//	SQRDMULH	by_element_Vector	by_element	Vector
954	//	FMLA	by_element_Vector	by_element	Vector
955	//	FMLS	by_element_Vector	by_element	Vector
956	//	FMUL	by_element_Vector	by_element	Vector
957	//	MLA	by_element	by_element	
958	//	UMLAL	by_element	by_element	
959	//	UMLAL2	by_element	by_element	
960	//	MLS	by_element	by_element	

1	in_use	Opcode	Extended Name	Specific	variant
961	//	UMLSL	by_element	by_element	
962	//	UMLSL2	by_element	by_element	
963	//	UMULL	by_element	by_element	
964	//	UMULL2	by_element	by_element	
965	//	FMULX	by_element_Vector	by_element	Vector
966	//	**AdvSIMD modified immed**			
967	//	MOVI	32_bit_shifted_immediate		32_bit_shifted
968	//	ORR	vector_immediate_32_bit	vector_immediate	32_bit
969	//	MOVI	16_bit_shifted_immediate		16_bit_shifted
970	//	ORR	vector_immediate_16_bit	vector_immediate	16_bit
971	//	MOVI	32_bit_shifting_ones		32_bit_shiftin
972	//	MOVI	8_bit		8_bit
973	//	FMOV	vector_immediate_Single_precision	vector_immediate	Single_precisi
974	//	MVNI	32_bit_shifted_immediate		32_bit_shifted
975	//	BIC	vector_immediate_32_bit	vector_immediate	32_bit
976	//	MVNI	16_bit_shifted_immediate		16_bit_shifted
977	//	BIC	vector_immediate_16_bit	vector_immediate	16_bit
978	//	MVNI	32_bit_shifting_ones		32_bit_shiftin
979	//	MOVI	64_bit_scalar		64_bit_scalar
980	//	MOVI	64_bit_vector		64_bit_vector
981	//	FMOV	vector_immediate_Double_precision	vector_immediate	Double_precis
982	//	**AdvSIMD shift by immedi**			
983	//	SSHR	Vector		Vector
984	//	SSRA	Vector		Vector
985	//	SRSHR	Vector		Vector
986	//	SRSRA	Vector		Vector
987	//	SHL	Vector		Vector
988	//	SQSHL	immediate_Vector	immediate	Vector
989	//	SHRN			
990	//	SHRN2			
991	//	RSHRN			
992	//	RSHRN2			
993	//	SQSHRN	Vector		Vector
994	//	SQSHRN2	Vector		Vector

1	in_use	Opcode	Extended Name	Specific	variant
995	//	SQRSHRN	Vector		Vector
996	//	SQRSHRN2	Vector		Vector
997	//	SSHLL			
998	//	SSHLL2			
999	//	SCVTF	vector_fixed_point_Vector	vector_fixed_point	Vector
1000	//	FCVTZS	vector_fixed_point_Vector	vector_fixed_point	Vector
1001	//	USHR	Vector		Vector
1002	//	USRA	Vector		Vector
1003	//	URSHR	Vector		Vector
1004	//	URSRA	Vector		Vector
1005	//	SRI	Vector		Vector
1006	//	SLI	Vector		Vector
1007	//	SQSHLU	Vector		Vector
1008	//	UQSHL	immediate_Vector	immediate	Vector
1009	//	SQSHRUN	Vector		Vector
1010	//	SQSHRUN2	Vector		Vector
1011	//	SQRSHRUN	Vector		Vector
1012	//	SQRSHRUN2	Vector		Vector
1013	//	UQSHRN	Vector		Vector
1014	//	UQRSHRN	Vector		Vector
1015	//	UQRSHRN2	Vector		Vector
1016	//	USHLL			
1017	//	USHLL2			
1018	//	UCVTF	vector_fixed_point_Vector	vector_fixed_point	Vector
1019	//	FCVTZU	vector_fixed_point_Vector	vector_fixed_point	Vector
1020	//	**AdvSIMD TBL/TBX**			
1021	//	TBL	Single_register_table		Single_registe
1022	//	TBX	Single_register_table		Single_registe
1023	//	TBL	Two_register_table		Two_register_
1024	//	TBX	Two_register_table		Two_register_
1025	//	TBL	Three_register_table		Three_registe
1026	//	TBX	Three_register_table		Three_registe
1027	//	TBL	Four_register_table		Four_register_
1028	//	TBX	Four_register_table		Four_register_

1	in_use	Opcode	Extended Name	Specific	variant
1029	//	**AdvSIMD ZIP/UZP/TRN**			
1030	//	UZP1			
1031	//	TRN1			
1032	//	ZIP1			
1033	//	UZP2			
1034	//	TRN2			
1035	//	ZIP2			
1036	//	**AdvSIMD EXT**			
1037	//	EXT			
1038	//	**Loads and stores**			
1039	//	**AdvSIMD load/store multi**			
1040	//	ST4	multiple_structures_No_offset	multiple_structures	No_offset
1041	//	ST1	multiple_structures_Four_registers	multiple_structures	Four_registers
1042	//	ST3	multiple_structures_No_offset	multiple_structures	No_offset
1043	//	ST1	multiple_structures_Three_registers	multiple_structures	Three_registe
1044	//	ST1	multiple_structures_One_register	multiple_structures	One_register
1045	//	ST2	multiple_structures_No_offset	multiple_structures	No_offset
1046	//	ST1	multiple_structures_Two_registers	multiple_structures	Two_registers
1047	//	LD4	multiple_structures_No_offset	multiple_structures	No_offset
1048	//	LD1	multiple_structures_Four_registers	multiple_structures	Four_registers
1049	//	LD3	multiple_structures_No_offset	multiple_structures	No_offset
1050	//	LD1	multiple_structures_Three_registers	multiple_structures	Three_registe
1051	//	LD1	multiple_structures_One_register	multiple_structures	One_register
1052	//	LD2	multiple_structures_No_offset	multiple_structures	No_offset
1053	//	LD1	multiple_structures_Two_registers	multiple_structures	Two_registers
1054	//	**AdvSIMD load/store multi**			
1055	//	ST4	multiple_structures_Register_offset	multiple_structures	Register_offse
1056	//	ST1	multiple_structures_Four_registers_regis	multiple_structures	Four_registers
1057	//	ST3	multiple_structures_Register_offset	multiple_structures	Register_offse
1058	//	ST1	multiple_structures_Three_registers_regi	multiple_structures	Three_registe
1059	//	ST1	multiple_structures_One_register_registe	multiple_structures	One_register_
1060	//	ST2	multiple_structures_Register_offset	multiple_structures	Register_offse
1061	//	ST1	multiple_structures_Two_registers_regist	multiple_structures	Two_registers
1062	//	ST4	multiple_structures_Immediate_offset	multiple_structures	Immediate_of

1	in_use	Opcode	Extended Name	Specific	variant
1063	//	ST1	multiple_structures_Four_registers_imme	multiple_structures	Four_registers
1064	//	ST3	multiple_structures_Immediate_offset	multiple_structures	Immediate_of
1065	//	ST1	multiple_structures_Three_registers_imn	multiple_structures	Three_registe
1066	//	ST1	multiple_structures_One_register_immed	multiple_structures	One_register_
1067	//	ST2	multiple_structures_Immediate_offset	multiple_structures	Immediate_of
1068	//	ST1	multiple_structures_Two_registers_imme	multiple_structures	Two_registers
1069	//	LD4	multiple_structures_Register_offset	multiple_structures	Register_offse
1070	//	LD1	multiple_structures_Four_registers_regis	multiple_structures	Four_registers
1071	//	LD3	multiple_structures_Register_offset	multiple_structures	Register_offse
1072	//	LD1	multiple_structures_Three_registers_regi	multiple_structures	Three_registe
1073	//	LD1	multiple_structures_One_register_registe	multiple_structures	One_register_
1074	//	LD2	multiple_structures_Register_offset	multiple_structures	Register_offse
1075	//	LD1	multiple_structures_Two_registers_regist	multiple_structures	Two_registers
1076	//	LD4	multiple_structures_Immediate_offset	multiple_structures	Immediate_of
1077	//	LD1	multiple_structures_Four_registers_imme	multiple_structures	Four_registers
1078	//	LD3	multiple_structures_Immediate_offset	multiple_structures	Immediate_of
1079	//	LD1	multiple_structures_Three_registers_imn	multiple_structures	Three_registe
1080	//	LD1	multiple_structures_One_register_immed	multiple_structures	One_register_
1081	//	LD2	multiple_structures_Immediate_offset	multiple_structures	Immediate_of
1082	//	LD1	multiple_structures_Two_registers_imme	multiple_structures	Two_registers
1083	//	**AdvSIMD load/store singl**			
1084	//	ST1	single_structure_8_bit	single_structure	8_bit
1085	//	ST3	single_structure_8_bit	single_structure	8_bit
1086	//	ST1	single_structure_16_bit	single_structure	16_bit
1087	//	ST3	single_structure_16_bit	single_structure	16_bit
1088	//	ST1	single_structure_32_bit	single_structure	32_bit
1089	//	ST1	single_structure_64_bit	single_structure	64_bit
1090	//	ST3	single_structure_32_bit	single_structure	32_bit
1091	//	ST3	single_structure_64_bit	single_structure	64_bit
1092	//	ST2	single_structure_8_bit	single_structure	8_bit
1093	//	ST4	single_structure_8_bit	single_structure	8_bit
1094	//	ST2	single_structure_16_bit	single_structure	16_bit
1095	//	ST4	single_structure_16_bit	single_structure	16_bit
1096	//	ST2	single_structure_32_bit	single_structure	32_bit

1	in_use	Opcode	Extended Name	Specific	variant
1097	//	ST2	single_structure_64_bit	single_structure	64_bit
1098	//	ST4	single_structure_32_bit	single_structure	32_bit
1099	//	ST4	single_structure_64_bit	single_structure	64_bit
1100	//	LD1	single_structure_8_bit	single_structure	8_bit
1101	//	LD3	single_structure_8_bit	single_structure	8_bit
1102	//	LD1	single_structure_16_bit	single_structure	16_bit
1103	//	LD3	single_structure_16_bit	single_structure	16_bit
1104	//	LD1	single_structure_32_bit	single_structure	32_bit
1105	//	LD1	single_structure_64_bit	single_structure	64_bit
1106	//	LD3	single_structure_32_bit	single_structure	32_bit
1107	//	LD3	single_structure_64_bit	single_structure	64_bit
1108	//	LD1R	No_offset		No_offset
1109	//	LD3R	No_offset		No_offset
1110	//	LD2	single_structure_8_bit	single_structure	8_bit
1111	//	LD4	single_structure_8_bit	single_structure	8_bit
1112	//	LD2	single_structure_16_bit	single_structure	16_bit
1113	//	LD4	single_structure_16_bit	single_structure	16_bit
1114	//	LD2	single_structure_32_bit	single_structure	32_bit
1115	//	LD2	single_structure_64_bit	single_structure	64_bit
1116	//	LD4	single_structure_32_bit	single_structure	32_bit
1117	//	LD4	single_structure_64_bit	single_structure	64_bit
1118	//	LD2R	No_offset		No_offset
1119	//	LD4R	No_offset		No_offset
1120	//	**AdvSIMD load/store singl**			
1121	//	ST1	single_structure_8_bit_register_offset	single_structure	8_bit_register
1122	//	ST3	single_structure_8_bit_register_offset	single_structure	8_bit_register
1123	//	ST1	single_structure_16_bit_register_offset	single_structure	16_bit_registe
1124	//	ST3	single_structure_16_bit_register_offset	single_structure	16_bit_registe
1125	//	ST1	single_structure_32_bit_register_offset	single_structure	32_bit_registe
1126	//	ST1	single_structure_64_bit_register_offset	single_structure	64_bit_registe
1127	//	ST3	single_structure_32_bit_register_offset	single_structure	32_bit_registe
1128	//	ST3	single_structure_64_bit_register_offset	single_structure	64_bit_registe
1129	//	ST1	single_structure_8_bit_immediate_offset	single_structure	8_bit_immedi
1130	//	ST3	single_structure_8_bit_immediate_offset	single_structure	8_bit_immedi

1	in_use	Opcode	Extended Name	Specific	variant
1131	//	ST1	single_structure_16_bit_immediate_offse	single_structure	16_bit_immec
1132	//	ST3	single_structure_16_bit_immediate_offse	single_structure	16_bit_immec
1133	//	ST1	single_structure_32_bit_immediate_offse	single_structure	32_bit_immec
1134	//	ST1	single_structure_64_bit_immediate_offse	single_structure	64_bit_immec
1135	//	ST3	single_structure_32_bit_immediate_offse	single_structure	32_bit_immec
1136	//	ST3	single_structure_64_bit_immediate_offse	single_structure	64_bit_immec
1137	//	ST2	single_structure_8_bit_register_offset	single_structure	8_bit_register
1138	//	ST4	single_structure_8_bit_register_offset	single_structure	8_bit_register
1139	//	ST2	single_structure_16_bit_register_offset	single_structure	16_bit_registe
1140	//	ST4	single_structure_16_bit_register_offset	single_structure	16_bit_registe
1141	//	ST2	single_structure_32_bit_register_offset	single_structure	32_bit_registe
1142	//	ST2	single_structure_64_bit_register_offset	single_structure	64_bit_registe
1143	//	ST4	single_structure_32_bit_register_offset	single_structure	32_bit_registe
1144	//	ST4	single_structure_64_bit_register_offset	single_structure	64_bit_registe
1145	//	ST2	single_structure_8_bit_immediate_offset	single_structure	8_bit_immedia
1146	//	ST4	single_structure_8_bit_immediate_offset	single_structure	8_bit_immedia
1147	//	ST2	single_structure_16_bit_immediate_offse	single_structure	16_bit_immec
1148	//	ST4	single_structure_16_bit_immediate_offse	single_structure	16_bit_immec
1149	//	ST2	single_structure_32_bit_immediate_offse	single_structure	32_bit_immec
1150	//	ST2	single_structure_64_bit_immediate_offse	single_structure	64_bit_immec
1151	//	ST4	single_structure_32_bit_immediate_offse	single_structure	32_bit_immec
1152	//	ST4	single_structure_64_bit_immediate_offse	single_structure	64_bit_immec
1153	//	LD1	single_structure_8_bit_register_offset	single_structure	8_bit_register
1154	//	LD3	single_structure_8_bit_register_offset	single_structure	8_bit_register
1155	//	LD1	single_structure_16_bit_register_offset	single_structure	16_bit_registe
1156	//	LD3	single_structure_16_bit_register_offset	single_structure	16_bit_registe
1157	//	LD1	single_structure_32_bit_register_offset	single_structure	32_bit_registe
1158	//	LD1	single_structure_64_bit_register_offset	single_structure	64_bit_registe
1159	//	LD3	single_structure_32_bit_register_offset	single_structure	32_bit_registe
1160	//	LD3	single_structure_64_bit_register_offset	single_structure	64_bit_registe
1161	//	LD1R	Register_offset		Register_offse
1162	//	LD3R	Register_offset		Register_offse
1163	//	LD1	single_structure_8_bit_immediate_offset	single_structure	8_bit_immedia
1164	//	LD3	single_structure_8_bit_immediate_offset	single_structure	8_bit_immedia

1	in_use	Opcode	Extended Name	Specific	variant
1165	//	LD1	single_structure_16_bit_immediate_offse	single_structure	16_bit_immed
1166	//	LD3	single_structure_16_bit_immediate_offse	single_structure	16_bit_immed
1167	//	LD1	single_structure_32_bit_immediate_offse	single_structure	32_bit_immed
1168	//	LD1	single_structure_64_bit_immediate_offse	single_structure	64_bit_immed
1169	//	LD3	single_structure_32_bit_immediate_offse	single_structure	32_bit_immed
1170	//	LD3	single_structure_64_bit_immediate_offse	single_structure	64_bit_immed
1171	//	LD1R	Immediate_offset		Immediate_of
1172	//	LD3R	Immediate_offset		Immediate_of
1173	//	LD2	single_structure_8_bit_register_offset	single_structure	8_bit_register
1174	//	LD4	single_structure_8_bit_register_offset	single_structure	8_bit_register
1175	//	LD2	single_structure_16_bit_register_offset	single_structure	16_bit_registe
1176	//	LD4	single_structure_16_bit_register_offset	single_structure	16_bit_registe
1177	//	LD2	single_structure_32_bit_register_offset	single_structure	32_bit_registe
1178	//	LD2	single_structure_64_bit_register_offset	single_structure	64_bit_registe
1179	//	LD4	single_structure_32_bit_register_offset	single_structure	32_bit_registe
1180	//	LD4	single_structure_64_bit_register_offset	single_structure	64_bit_registe
1181	//	LD2R	Register_offset		Register_offse
1182	//	LD4R	Register_offset		Register_offse
1183	//	LD2	single_structure_8_bit_immediate_offset	single_structure	8_bit_immedi
1184	//	LD4	single_structure_8_bit_immediate_offset	single_structure	8_bit_immedi
1185	//	LD2	single_structure_16_bit_immediate_offse	single_structure	16_bit_immed
1186	//	LD4	single_structure_16_bit_immediate_offse	single_structure	16_bit_immed
1187	//	LD2	single_structure_32_bit_immediate_offse	single_structure	32_bit_immed
1188	//	LD2	single_structure_64_bit_immediate_offse	single_structure	64_bit_immed
1189	//	LD4	single_structure_32_bit_immediate_offse	single_structure	32_bit_immed
1190	//	LD4	single_structure_64_bit_immediate_offse	single_structure	64_bit_immed
1191	//	LD2R	Immediate_offset		Immediate_of
1192	//	LD4R	Immediate_offset		Immediate_of

in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	UNALLOCATED					**0**	**0**											
	BAD	invalid operation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Branch,exception gener					**1**	**0**	**1**										
	Compare _ Branch (imme		**-**	**0**	**1**	**1**	**0**	**1**	**0**	**-**								
	CBZ		0	0	1	1	0	1	0	0								
	CBNZ		0	0	1	1	0	1	0	1								
	CBZ		1	0	1	1	0	1	0	0								
	CBNZ		1	0	1	1	0	1	0	1								
	Test & branch (immediate		**b5**	**0**	**1**	**1**	**0**	**1**	**1**	**-**			**b40**					
	TBZ		b5	0	1	1	0	1	1	0			b40					
	TBNZ		b5	0	1	1	0	1	1	1			b40					
	Conditional branch (imme		**0**	**1**	**0**	**1**	**0**	**1**	**0**	**-**								
	B_cond		0	1	0	1	0	1	0	0								
	Exception generation		**1**	**1**	**0**	**1**	**0**	**1**	**0**	**0**	**-**	**-**	**-**					
	SVC		1	1	0	1	0	1	0	0	0	0	0					
	HVC		1	1	0	1	0	1	0	0	0	0	0					
	SMC		1	1	0	1	0	1	0	0	0	0	0					
	BRK		1	1	0	1	0	1	0	0	0	0	1					
	HLT		1	1	0	1	0	1	0	0	0	1	0					
	DCPS1		1	1	0	1	0	1	0	0	1	0	1					
	DCPS2		1	1	0	1	0	1	0	0	1	0	1					
	DCPS3		1	1	0	1	0	1	0	0	1	0	1					
	System		**1**	**1**	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**0**	**-**	**-**	**-**		**op1**	
	MSR		1	1	0	1	0	1	0	1	0	0	0	0	0		op1	
	HINT		1	1	0	1	0	1	0	1	0	0	0	0	0	0	1	1
	CLREX		1	1	0	1	0	1	0	1	0	0	0	0	0	0	1	1
	DSB		1	1	0	1	0	1	0	1	0	0	0	0	0	0	1	1
	DMB		1	1	0	1	0	1	0	1	0	0	0	0	0	0	1	1
	ISB		1	1	0	1	0	1	0	1	0	0	0	0	0	0	1	1
	SYS		1	1	0	1	0	1	0	1	0	0	0	0	1		op1	
	MSR		1	1	0	1	0	1	0	1	0	0	0	1	-		op1	
	SYSL		1	1	0	1	0	1	0	1	0	0	1	0	1		op1	
	MRS		1	1	0	1	0	1	0	1	0	0	1	1	-		op1	
	Unconditional branch (reg		**1**	**1**	**0**	**1**	**0**	**1**	**1**		**opc**					**op2**		
	BR		1	1	0	1	0	1	1	0	0	0	0	1	1	1	1	1
	BLR		1	1	0	1	0	1	1	0	0	0	1	1	1	1	1	1
	RET		1	1	0	1	0	1	1	0	0	1	0	1	1	1	1	1
	ERET		1	1	0	1	0	1	1	0	1	0	0	1	1	1	1	1

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
40		DRPS		1	1	0	1	0	1	1	0	1	0	1	1	1	1	1	1
41		**Unconditional branch (im**		**-**	**0**	**0**	**1**	**0**	**1**										
42		B		0	0	0	1	0	1										
43		BL		1	0	0	1	0	1										
44		**Loads and stores**						**1**		**0**									
45		**Load/store exclusive**		**-**	**-**	**0**	**0**	**1**	**0**	**0**	**0**	**-**	**-**	**-**			**Rs**		
46		STXRB		0	0	0	0	1	0	0	0	0	0	0			Rs		
47		STLXRB		0	0	0	0	1	0	0	0	0	0	0			Rs		
48		LDXRB		0	0	0	0	1	0	0	0	0	1	0			Rs		
49		LDAXRB		0	0	0	0	1	0	0	0	0	1	0			Rs		
50		STLRB		0	0	0	0	1	0	0	0	1	0	0			Rs		
51		LDARB		0	0	0	0	1	0	0	0	1	1	0			Rs		
52		STXRH		0	1	0	0	1	0	0	0	0	0	0			Rs		
53		STLXRH		0	1	0	0	1	0	0	0	0	0	0			Rs		
54		LDXRH		0	1	0	0	1	0	0	0	0	1	0			Rs		
55		LDAXRH		0	1	0	0	1	0	0	0	0	1	0			Rs		
56		STLRH		0	1	0	0	1	0	0	0	1	0	0			Rs		
57		LDARH		0	1	0	0	1	0	0	0	1	1	0			Rs		
58		STXR		1	0	0	0	1	0	0	0	0	0	0			Rs		
59		STLXR		1	0	0	0	1	0	0	0	0	0	0			Rs		
60		STXP		1	0	0	0	1	0	0	0	0	0	1			Rs		
61		STLXP		1	0	0	0	1	0	0	0	0	0	1			Rs		
62		LDXR		1	0	0	0	1	0	0	0	0	1	0			Rs		
63		LDAXR		1	0	0	0	1	0	0	0	0	1	0			Rs		
64		LDXP		1	0	0	0	1	0	0	0	0	1	1			Rs		
65		LDAXP		1	0	0	0	1	0	0	0	0	1	1			Rs		
66		STLR		1	0	0	0	1	0	0	0	1	0	0			Rs		
67		LDAR		1	0	0	0	1	0	0	0	1	1	0			Rs		
68		STXR		1	1	0	0	1	0	0	0	0	0	0			Rs		
69		STLXR		1	1	0	0	1	0	0	0	0	0	0			Rs		
70		STXP		1	1	0	0	1	0	0	0	0	0	1			Rs		
71		STLXP		1	1	0	0	1	0	0	0	0	0	1			Rs		
72		LDXR		1	1	0	0	1	0	0	0	0	1	0			Rs		
73		LDAXR		1	1	0	0	1	0	0	0	0	1	0			Rs		
74		LDXP		1	1	0	0	1	0	0	0	0	1	1			Rs		
75		LDAXP		1	1	0	0	1	0	0	0	0	1	1			Rs		
76		STLR		1	1	0	0	1	0	0	0	1	0	0			Rs		
77		LDAR		1	1	0	0	1	0	0	0	1	1	0			Rs		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
78		**Load register (literal)**		**-**	**-**	**0**	**1**	**1**	**-**	**0**	**0**								
79		LDR		0	0	0	1	1	0	0	0								
80		LDR		0	0	0	1	1	1	0	0								
81		LDR		0	1	0	1	1	0	0	0								
82		LDR		0	1	0	1	1	1	0	0								
83		LDRSW		1	0	0	1	1	0	0	0								
84		LDR		1	0	0	1	1	1	0	0								
85		PRFM		1	1	0	1	1	0	0	0								
86		**Load/store no-allocate pa**		**-**	**-**	**1**	**0**	**1**	**-**	**0**	**0**	**0**	**-**				**imm7**		
87		STNP		0	0	1	0	1	0	0	0	0	0				imm7		
88		LDNP		0	0	1	0	1	0	0	0	0	1				imm7		
89		STNP		0	0	1	0	1	1	0	0	0	0				imm7		
90		LDNP		0	0	1	0	1	1	0	0	0	1				imm7		
91		STNP		0	1	1	0	1	1	0	0	0	0				imm7		
92		LDNP		0	1	1	0	1	1	0	0	0	1				imm7		
93		STNP		1	0	1	0	1	0	0	0	0	0				imm7		
94		LDNP		1	0	1	0	1	0	0	0	0	1				imm7		
95		STNP		1	0	1	0	1	1	0	0	0	0				imm7		
96		LDNP		1	0	1	0	1	1	0	0	0	1				imm7		
97		**Load/store register pair (p**		**-**	**-**	**1**	**0**	**1**	**-**	**0**	**0**	**1**	**-**				**imm7**		
98		STP		0	0	1	0	1	0	0	0	1	0				imm7		
99		LDP		0	0	1	0	1	0	0	0	1	1				imm7		
100		STP		0	0	1	0	1	1	0	0	1	0				imm7		
101		LDP		0	0	1	0	1	1	0	0	1	1				imm7		
102		LDPSW		0	1	1	0	1	0	0	0	1	1				imm7		
103		STP		0	1	1	0	1	1	0	0	1	0				imm7		
104		LDP		0	1	1	0	1	1	0	0	1	1				imm7		
105		STP		1	0	1	0	1	0	0	0	1	0				imm7		
106		LDP		1	0	1	0	1	0	0	0	1	1				imm7		
107		STP		1	0	1	0	1	1	0	0	1	0				imm7		
108		LDP		1	0	1	0	1	1	0	0	1	1				imm7		
109		**Load/store register pair (o**		**opc**		**1**	**0**	**1**	**V**	**0**	**1**	**0**	**L**				**imm7**		
110		STP		0	0	1	0	1	0	0	1	0	0				imm7		
111		LDP		0	0	1	0	1	0	0	1	0	1				imm7		
112		STP		0	0	1	0	1	1	0	1	0	0				imm7		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
113		LDP		0	0	1	0	1	1	0	1	0	1			imm7			
114		LDPSW		0	1	1	0	1	0	0	1	0	1			imm7			
115		STP		0	1	1	0	1	1	0	1	0	0			imm7			
116		LDP		0	1	1	0	1	1	0	1	0	1			imm7			
117		STP		1	0	1	0	1	0	0	1	0	0			imm7			
118		LDP		1	0	1	0	1	0	0	1	0	1			imm7			
119		STP		1	0	1	0	1	1	0	1	0	0			imm7			
120		LDP		1	0	1	0	1	1	0	1	0	1			imm7			
121		**Load/store register pair (p**		**opc**		**1**	**0**	**1**	**V**	**0**	**1**	**1**	**L**			**imm7**			
122		STP		0	0	1	0	1	0	0	1	1	0			imm7			
123		LDP		0	0	1	0	1	0	0	1	1	1			imm7			
124		STP		0	0	1	0	1	1	0	1	1	0			imm7			
125		LDP		0	0	1	0	1	1	0	1	1	1			imm7			
126		LDPSW		0	1	1	0	1	0	0	1	1	1			imm7			
127		STP		0	1	1	0	1	1	0	1	1	0			imm7			
128		LDP		0	1	1	0	1	1	0	1	1	1			imm7			
129		STP		1	0	1	0	1	0	0	1	1	0			imm7			
130		LDP		1	0	1	0	1	0	0	1	1	1			imm7			
131		STP		1	0	1	0	1	1	0	1	1	0			imm7			
132		LDP		1	0	1	0	1	1	0	1	1	1			imm7			
133		**Load/store register (unsc**		**size**		**1**	**1**	**1**	**V**	**0**	**0**	**opc**		**0**					**imm**
134		STURB		0	0	1	1	1	0	0	0	0	0	0					imm
135		LDURB		0	0	1	1	1	0	0	0	0	1	0					imm
136		LDURSB		0	0	1	1	1	0	0	0	1	0	0					imm
137		LDURSB		0	0	1	1	1	0	0	0	1	1	0					imm
138		STUR		0	0	1	1	1	1	0	0	0	0	0					imm
139		LDUR		0	0	1	1	1	1	0	0	0	1	0					imm
140		STUR		0	0	1	1	1	1	0	0	1	0	0					imm
141		LDUR		0	0	1	1	1	1	0	0	1	1	0					imm
142		STURH		0	1	1	1	1	0	0	0	0	0	0					imm
143		LDURH		0	1	1	1	1	0	0	0	0	1	0					imm
144		LDURSH		0	1	1	1	1	0	0	0	1	0	0					imm
145		LDURSH		0	1	1	1	1	0	0	0	1	1	0					imm
146		STUR		0	1	1	1	1	1	0	0	0	0	0					imm
147		LDUR		0	1	1	1	1	1	0	0	0	1	0					imm
148		STUR		1	0	1	1	1	0	0	0	0	0	0					imm
149		LDUR		1	0	1	1	1	0	0	0	0	1	0					imm
150		LDURSW		1	0	1	1	1	0	0	0	1	0	0					imm

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
151		STUR		1	0	1	1	1	1	0	0	0	0	0					imm9
152		LDUR		1	0	1	1	1	1	0	0	0	1	0					imm9
153		STUR		1	1	1	1	1	0	0	0	0	0	0					imm9
154		LDUR		1	1	1	1	1	0	0	0	0	1	0					imm9
155		PRFUM		1	1	1	1	1	0	0	0	1	0	0					imm9
156		STUR		1	1	1	1	1	1	0	0	0	0	0					imm9
157		LDUR		1	1	1	1	1	1	0	0	0	1	0					imm9
158		**Load/store register (imme**		**size**		**1**	**1**	**1**	**V**	**0**	**0**	**opc**		**0**					**imm9**
159		STRB		0	0	1	1	1	0	0	0	0	0	0					imm9
160		LDRB		0	0	1	1	1	0	0	0	0	1	0					imm9
161		LDRSB		0	0	1	1	1	0	0	0	1	0	0					imm9
162		LDRSB		0	0	1	1	1	0	0	0	1	1	0					imm9
163		STR		0	0	1	1	1	1	0	0	0	0	0					imm9
164		LDR		0	0	1	1	1	1	0	0	0	1	0					imm9
165		STR		0	0	1	1	1	1	0	0	1	0	0					imm9
166		LDR		0	0	1	1	1	1	0	0	1	1	0					imm9
167		STRH		0	1	1	1	1	0	0	0	0	0	0					imm9
168		LDRH		0	1	1	1	1	0	0	0	0	1	0					imm9
169		LDRSH		0	1	1	1	1	0	0	0	1	0	0					imm9
170		LDRSH		0	1	1	1	1	0	0	0	1	1	0					imm9
171		STR		0	1	1	1	1	1	0	0	0	0	0					imm9
172		LDR		0	1	1	1	1	1	0	0	0	1	0					imm9
173		STR		1	0	1	1	1	0	0	0	0	0	0					imm9
174		LDR		1	0	1	1	1	0	0	0	0	1	0					imm9
175		LDRSW		1	0	1	1	1	0	0	0	1	0	0					imm9
176		STR		1	0	1	1	1	1	0	0	0	0	0					imm9
177		LDR		1	0	1	1	1	1	0	0	0	1	0					imm9
178		STR		1	1	1	1	1	0	0	0	0	0	0					imm9
179		LDR		1	1	1	1	1	0	0	0	0	1	0					imm9
180		STR		1	1	1	1	1	1	0	0	0	0	0					imm9
181		LDR		1	1	1	1	1	1	0	0	0	1	0					imm9
182		**Load/store register (unpri**		**size**		**1**	**1**	**1**	**V**	**0**	**0**	**opc**		**0**					**imm9**
183		STTRB		0	0	1	1	1	0	0	0	0	0	0					imm9
184		LDTRB		0	0	1	1	1	0	0	0	0	1	0					imm9
185		LDTRSB		0	0	1	1	1	0	0	0	1	0	0					imm9
186		LDTRSB		0	0	1	1	1	0	0	0	1	1	0					imm9
187		STTRH		0	1	1	1	1	0	0	0	0	0	0					imm9
188		LDTRH		0	1	1	1	1	0	0	0	0	1	0					imm9

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
189		LDTRSH		0	1	1	1	1	0	0	0	1	0	0					imm9
190		LDTRSH		0	1	1	1	1	0	0	0	1	1	0					imm9
191		STTR		1	0	1	1	1	0	0	0	0	0	0					imm9
192		LDTR		1	0	1	1	1	0	0	0	0	1	0					imm9
193		LDTRSW		1	0	1	1	1	0	0	0	1	0	0					imm9
194		STTR		1	1	1	1	1	0	0	0	0	0	0					imm9
195		LDTR		1	1	1	1	1	0	0	0	0	1	0					imm9
196		**Load/store register (imme**		**size**		**1**	**1**	**1**	**V**	**0**	**0**	**opc**		**0**					**imm9**
197		STRB		0	0	1	1	1	0	0	0	0	0	0					imm9
198		LDRB		0	0	1	1	1	0	0	0	0	1	0					imm9
199		LDRSB		0	0	1	1	1	0	0	0	1	0	0					imm9
200		LDRSB		0	0	1	1	1	0	0	0	1	1	0					imm9
201		STR		0	0	1	1	1	1	0	0	0	0	0					imm9
202		LDR		0	0	1	1	1	1	0	0	0	1	0					imm9
203		STR		0	0	1	1	1	1	0	0	1	0	0					imm9
204		LDR		0	0	1	1	1	1	0	0	1	1	0					imm9
205		STRH		0	1	1	1	1	0	0	0	0	0	0					imm9
206		LDRH		0	1	1	1	1	0	0	0	0	1	0					imm9
207		LDRSH		0	1	1	1	1	0	0	0	1	0	0					imm9
208		LDRSH		0	1	1	1	1	0	0	0	1	1	0					imm9
209		STR		0	1	1	1	1	1	0	0	0	0	0					imm9
210		LDR		0	1	1	1	1	1	0	0	0	1	0					imm9
211		STR		1	0	1	1	1	0	0	0	0	0	0					imm9
212		LDR		1	0	1	1	1	0	0	0	0	1	0					imm9
213		LDRSW		1	0	1	1	1	0	0	0	1	0	0					imm9
214		STR		1	0	1	1	1	1	0	0	0	0	0					imm9
215		LDR		1	0	1	1	1	1	0	0	0	1	0					imm9
216		STR		1	1	1	1	1	0	0	0	0	0	0					imm9
217		LDR		1	1	1	1	1	0	0	0	0	1	0					imm9
218		STR		1	1	1	1	1	1	0	0	0	0	0					imm9
219		LDR		1	1	1	1	1	1	0	0	0	1	0					imm9
220		**Load/store register (regis**		**size**		**1**	**1**	**1**	**v**	**0**	**0**	**opc**		**1**			**Rm**		
221		STRB		0	0	1	1	1	0	0	0	0	0	1			Rm		
222		LDRB		0	0	1	1	1	0	0	0	0	1	1			Rm		
223		LDRSB		0	0	1	1	1	0	0	0	1	0	1			Rm		
224		LDRSB		0	0	1	1	1	0	0	0	1	1	1			Rm		
225		STR		0	0	1	1	1	1	0	0	0	0	1			Rm		
226		LDR		0	0	1	1	1	1	0	0	0	1	1			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
227		STR		0	0	1	1	1	1	0	0	1	0	1			Rm		
228		LDR		0	0	1	1	1	1	0	0	1	1	1			Rm		
229		STRH		0	1	1	1	1	0	0	0	0	0	1			Rm		
230		LDRH		0	1	1	1	1	0	0	0	0	1	1			Rm		
231		LDRSH		0	1	1	1	1	0	0	0	1	0	1			Rm		
232		LDRSH		0	1	1	1	1	0	0	0	1	1	1			Rm		
233		STR		0	1	1	1	1	1	0	0	0	0	1			Rm		
234		LDR		0	1	1	1	1	1	0	0	0	1	1			Rm		
235		STR		1	0	1	1	1	0	0	0	0	0	1			Rm		
236		LDR		1	0	1	1	1	0	0	0	0	1	1			Rm		
237		LDRSW		1	0	1	1	1	0	0	0	1	0	1			Rm		
238		STR		1	0	1	1	1	1	0	0	0	0	1			Rm		
239		LDR		1	0	1	1	1	1	0	0	0	1	1			Rm		
240		STR		1	1	1	1	1	0	0	0	0	0	1			Rm		
241		LDR		1	1	1	1	1	0	0	0	0	1	1			Rm		
242		PRFM		1	1	1	1	1	0	0	0	1	0	1			Rm		
243		STR		1	1	1	1	1	1	0	0	0	0	1			Rm		
244		LDR		1	1	1	1	1	1	0	0	0	1	1			Rm		
245		**Load/store register (unsig**		**size**		**1**	**1**	**1**	**v**	**0**	**1**	**opc**							**imn**
246		STRB		0	0	1	1	1	0	0	1	0	0						imn
247		LDRB		0	0	1	1	1	0	0	1	0	1						imn
248		LDRSB		0	0	1	1	1	0	0	1	1	0						imn
249		LDRSB		0	0	1	1	1	0	0	1	1	1						imn
250		STR		0	0	1	1	1	1	0	1	0	0						imn
251		LDR		0	0	1	1	1	1	0	1	0	1						imn
252		STR		0	0	1	1	1	1	0	1	1	0						imn
253		LDR		0	0	1	1	1	1	0	1	1	1						imn
254		STRH		0	1	1	1	1	0	0	1	0	0						imn
255		LDRH		0	1	1	1	1	0	0	1	0	1						imn
256		LDRSH		0	1	1	1	1	0	0	1	1	0						imn
257		LDRSH		0	1	1	1	1	0	0	1	1	1						imn
258		STR		0	1	1	1	1	1	0	1	0	0						imn
259		LDR		0	1	1	1	1	1	0	1	0	1						imn
260		STR		1	0	1	1	1	0	0	1	0	0						imn
261		LDR		1	0	1	1	1	0	0	1	0	1						imn
262		LDRSW		1	0	1	1	1	0	0	1	1	0						imn
263		STR		1	0	1	1	1	1	0	1	0	0						imn
264		LDR		1	0	1	1	1	1	0	1	0	1						imn

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
265		STR		1	1	1	1	1	0	0	1	0	0						imm
266		LDR		1	1	1	1	1	0	0	1	0	1						imm
267		PRFM		1	1	1	1	1	0	0	1	1	0						imm
268		STR		1	1	1	1	1	1	0	1	0	0						imm
269		LDR		1	1	1	1	1	1	0	1	0	1						imm
270		**Data processing – Imme**					**1**	**0**	**0**										
271		**PC-rel. addressing**		**op**	**immlo**		**1**	**0**	**0**	**0**	**0**								
272		ADR		0	immlo		1	0	0	0	0								
273		ADRP		1	immlo		1	0	0	0	0								
274		**Add/subtract (immediate)**		**sf**	**op**	**S**	**1**	**0**	**0**	**0**	**1**	**shift**							**imm**
275		ADD		0	0	0	1	0	0	0	1	-	-						imm
276		ADDS		0	0	1	1	0	0	0	1	-	-						imm
277		SUB		0	1	0	1	0	0	0	1	-	-						imm
278		SUBS		0	1	1	1	0	0	0	1	-	-						imm
279		ADD		1	0	0	1	0	0	0	1	-	-						imm
280		ADDS		1	0	1	1	0	0	0	1	-	-						imm
281		SUB		1	1	0	1	0	0	0	1	-	-						imm
282		SUBS		1	1	1	1	0	0	0	1	-	-						imm
283		**Logical (immediate)**		**sf**	**opc**		**1**	**0**	**0**	**1**	**0**	**0**	**N**			**immr**			
284		AND		0	0	0	1	0	0	1	0	0	0			immr			
285		ORR		0	0	1	1	0	0	1	0	0	0			immr			
286		EOR		0	1	0	1	0	0	1	0	0	0			immr			
287		ANDS		0	1	1	1	0	0	1	0	0	0			immr			
288		AND		1	0	0	1	0	0	1	0	0	-			immr			
289		ORR		1	0	1	1	0	0	1	0	0	-			immr			
290		EOR		1	1	0	1	0	0	1	0	0	-			immr			
291		ANDS		1	1	1	1	0	0	1	0	0	-			immr			
292		**Move wide (immediate)**		**sf**	**opc**		**1**	**0**	**0**	**1**	**0**	**1**	**hw**						
293		MOVN		0	0	0	1	0	0	1	0	1	-	-					
294		MOVZ		0	1	0	1	0	0	1	0	1	-	-					
295		MOVK		0	1	1	1	0	0	1	0	1	-	-					
296		MOVN		1	0	0	1	0	0	1	0	1	-	-					
297		MOVZ		1	1	0	1	0	0	1	0	1	-	-					
298		MOVK		1	1	1	1	0	0	1	0	1	-	-					
299		**Bitfield**		**sf**	**opc**		**1**	**0**	**0**	**1**	**1**	**0**	**N**			**immr**			
300		SBFM		0	0	0	1	0	0	1	1	0	0			immr			
301		BFM		0	0	1	1	0	0	1	1	0	0			immr			
302		UBFM		0	1	0	1	0	0	1	1	0	0			immr			

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
303		SBFM		1	0	0	1	0	0	1	1	0	1			immr			
304		BFM		1	0	1	1	0	0	1	1	0	1			immr			
305		UBFM		1	1	0	1	0	0	1	1	0	1			immr			
306		**Extract**		**sf**		**op21**	**1**	**0**	**0**	**1**	**1**	**1**	**N**	**o0**		**Rm**			
307		EXTR		0	0	0	1	0	0	1	1	1	0	0		Rm			
308		EXTR		1	0	0	1	0	0	1	1	1	1	0		Rm			
309		**Data Processing – regis**						**1**	**0**	**1**									
310		**Logical (shifted register)**		**sf**		**opc**	**0**	**1**	**0**	**1**	**0**	**shift**		**N**		**Rm**			
311		AND		0	0	0	0	1	0	1	0	shift		0		Rm			
312		BIC		0	0	0	0	1	0	1	0	shift		1		Rm			
313		ORR		0	0	1	0	1	0	1	0	shift		0		Rm			
314		ORN		0	0	1	0	1	0	1	0	shift		1		Rm			
315		EOR		0	1	0	0	1	0	1	0	shift		0		Rm			
316		EON		0	1	0	0	1	0	1	0	shift		1		Rm			
317		ANDS		0	1	1	0	1	0	1	0	shift		0		Rm			
318		BICS		0	1	1	0	1	0	1	0	shift		1		Rm			
319		AND		1	0	0	0	1	0	1	0	shift		0		Rm			
320		BIC		1	0	0	0	1	0	1	0	shift		1		Rm			
321		ORR		1	0	1	0	1	0	1	0	shift		0		Rm			
322		ORN		1	0	1	0	1	0	1	0	shift		1		Rm			
323		EOR		1	1	0	0	1	0	1	0	shift		0		Rm			
324		EON		1	1	0	0	1	0	1	0	shift		1		Rm			
325		ANDS		1	1	1	0	1	0	1	0	shift		0		Rm			
326		BICS		1	1	1	0	1	0	1	0	shift		1		Rm			
327		**Add/subtract (shifted regi**		**sf**	**op**	**S**	**0**	**1**	**0**	**1**	**1**	**shift**		**0**		**Rm**			
328		ADD		0	0	0	0	1	0	1	1	-	-	0		Rm			
329		ADDS		0	0	1	0	1	0	1	1	-	-	0		Rm			
330		SUB		0	1	0	0	1	0	1	1	-	-	0		Rm			
331		SUBS		0	1	1	0	1	0	1	1	-	-	0		Rm			
332		ADD		1	0	0	0	1	0	1	1	-	-	0		Rm			
333		ADDS		1	0	1	0	1	0	1	1	-	-	0		Rm			
334		SUB		1	1	0	0	1	0	1	1	-	-	0		Rm			
335		SUBS		1	1	1	0	1	0	1	1	-	-	0		Rm			
336		**Add/subtract (extended re**		**sf**	**op**	**S**	**0**	**1**	**0**	**1**	**1**	**opt**		**1**		**Rm**			
337		ADD		0	0	0	0	1	0	1	1	0	0	1		Rm			
338		ADDS		0	0	1	0	1	0	1	1	0	0	1		Rm			
339		SUB		0	1	0	0	1	0	1	1	0	0	1		Rm			
340		SUBS		0	1	1	0	1	0	1	1	0	0	1		Rm			

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
341		ADD		1	0	0	0	1	0	1	1	0	0	1			Rm		
342		ADDS		1	0	1	0	1	0	1	1	0	0	1			Rm		
343		SUB		1	1	0	0	1	0	1	1	0	0	1			Rm		
344		SUBS		1	1	1	0	1	0	1	1	0	0	1			Rm		
345		**Add/subtract (with carry)**		**sf**	**op**	**S**	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**0**			Rm		
346		ADC		0	0	0	1	1	0	1	0	0	0	0			Rm		
347		ADCS		0	0	1	1	1	0	1	0	0	0	0			Rm		
348		SBC		0	1	0	1	1	0	1	0	0	0	0			Rm		
349		SBCS		0	1	1	1	1	0	1	0	0	0	0			Rm		
350		ADC		1	0	0	1	1	0	1	0	0	0	0			Rm		
351		ADCS		1	0	1	1	1	0	1	0	0	0	0			Rm		
352		SBC		1	1	0	1	1	0	1	0	0	0	0			Rm		
353		SBCS		1	1	1	1	1	0	1	0	0	0	0			Rm		
354		**Conditional compare (reg**		**sf**	**op**	**S**	**1**	**1**	**0**	**1**	**0**	**0**	**1**	**0**			**imm5**		
355		CCMN		0	0	1	1	1	0	1	0	0	1	0			imm5		
356		CCMN		1	0	1	1	1	0	1	0	0	1	0			imm5		
357		CCMP		0	1	1	1	1	0	1	0	0	1	0			imm5		
358		CCMP		1	1	1	1	1	0	1	0	0	1	0			imm5		
359		**Conditional compare (imr**		**sf**	**op**	**S**	**1**	**1**	**0**	**1**	**0**	**0**	**1**	**0**			**imm5**		
360		CCMN		0	0	1	1	1	0	1	0	0	1	0			imm5		
361		CCMN		1	0	1	1	1	0	1	0	0	1	0			imm5		
362		CCMP		0	1	1	1	1	0	1	0	0	1	0			imm5		
363		CCMP		1	1	1	1	1	0	1	0	0	1	0			imm5		
364		**Conditional select**		**sf**	**op**	**S**	**1**	**1**	**0**	**1**	**0**	**1**	**0**	**0**			**Rm**		
365		CSEL		0	0	0	1	1	0	1	0	1	0	0			Rm		
366		CSINC		0	0	0	1	1	0	1	0	1	0	0			Rm		
367		CSINV		0	1	0	1	1	0	1	0	1	0	0			Rm		
368		CSNEG		0	1	0	1	1	0	1	0	1	0	0			Rm		
369		CSEL		1	0	0	1	1	0	1	0	1	0	0			Rm		
370		CSINC		1	0	0	1	1	0	1	0	1	0	0			Rm		
371		CSINV		1	1	0	1	1	0	1	0	1	0	0			Rm		
372		CSNEG		1	1	0	1	1	0	1	0	1	0	0			Rm		
373		**Data-processing (3 sourc**		**sf**		**op54**	**1**	**1**	**0**	**1**	**1**		**op31**				**Rm**		
374		MADD		0	0	0	1	1	0	1	1	0	0	0			Rm		
375		MADD		1	0	0	1	1	0	1	1	0	0	0			Rm		
376		SMADDL		1	0	0	1	1	0	1	1	0	0	1			Rm		
377		UMADDL		1	0	0	1	1	0	1	1	1	0	1			Rm		
378		MSUB		0	0	0	1	1	0	1	1	0	0	0			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
379		MSUB		1	0	0	1	1	0	1	1	0	0	0			Rm		
380		SMSUBL		1	0	0	1	1	0	1	1	0	0	1			Rm		
381		UMSUBL		1	0	0	1	1	0	1	1	1	0	1			Rm		
382		SMULH		1	0	0	1	1	0	1	1	0	1	0			Rm		
383		UMULH		1	0	0	1	1	0	1	1	1	1	0			Rm		
384		**Data-processing (2 sourc**		**sf**	**0**	**S**	**1**	**1**	**0**	**1**	**0**	**1**	**1**	**0**			**Rm**		
385		CRC32X		1	0	0	1	1	0	1	0	1	1	0			Rm		
386		CRC32CX		1	0	0	1	1	0	1	0	1	1	0			Rm		
387		CRC32B		0	0	0	1	1	0	1	0	1	1	0			Rm		
388		CRC32CB		0	0	0	1	1	0	1	0	1	1	0			Rm		
389		CRC32H		0	0	0	1	1	0	1	0	1	1	0			Rm		
390		CRC32CH		0	0	0	1	1	0	1	0	1	1	0			Rm		
391		CRC32W		0	0	0	1	1	0	1	0	1	1	0			Rm		
392		CRC32CW		0	0	0	1	1	0	1	0	1	1	0			Rm		
393		UDIV		0	0	0	1	1	0	1	0	1	1	0			Rm		
394		UDIV		1	0	0	1	1	0	1	0	1	1	0			Rm		
395		SDIV		0	0	0	1	1	0	1	0	1	1	0			Rm		
396		SDIV		1	0	0	1	1	0	1	0	1	1	0			Rm		
397		LSLV		0	0	0	1	1	0	1	0	1	1	0			Rm		
398		LSLV		1	0	0	1	1	0	1	0	1	1	0			Rm		
399		LSRV		0	0	0	1	1	0	1	0	1	1	0			Rm		
400		LSRV		1	0	0	1	1	0	1	0	1	1	0			Rm		
401		ASRV		0	0	0	1	1	0	1	0	1	1	0			Rm		
402		ASRV		1	0	0	1	1	0	1	0	1	1	0			Rm		
403		RORV		0	0	0	1	1	0	1	0	1	1	0			Rm		
404		RORV		1	0	0	1	1	0	1	0	1	1	0			Rm		
405		**Data-processing (1 sourc**		**sf**	**1**	**S**	**1**	**1**	**0**	**1**	**0**	**1**	**1**	**0**		**opcode2**			
406		RBIT		0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
407		RBIT		1	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
408		CLZ		0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
409		CLZ		1	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
410		CLS		0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
411		CLS		1	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
412		REV		0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
413		REV		1	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
414		REV16		1	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
415		REV16		0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0
416		REV32		1	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
417	//	**Data Processing – SIMD**						**1**	**1**	**1**									
418	//	**Floating-point<->fixed-po**		**sf**	**0**	**S**	**1**	**1**	**1**	**1**	**0**	**type**		**0**	**rmode**		**opcode**		
419	//	SCVTF		0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0
420	//	UCVTF		0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1
421	//	FCVTZS		0	0	0	1	1	1	1	0	1	1	0	1	1	0	0	0
422	//	FCVTZU		0	0	0	1	1	1	1	0	1	1	0	1	1	0	0	1
423	//	SCVTF		0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0
424	//	UCVTF		0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1
425	//	FCVTZS		0	0	0	1	1	1	1	0	1	1	0	1	1	0	0	0
426	//	FCVTZU		0	0	0	1	1	1	1	0	1	1	0	1	1	0	0	1
427	//	SCVTF		1	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0
428	//	UCVTF		1	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1
429	//	FCVTZS		1	0	0	1	1	1	1	0	1	1	0	1	1	0	0	0
430	//	FCVTZU		1	0	0	1	1	1	1	0	1	1	0	1	1	0	0	1
431	//	SCVTF		1	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0
432	//	UCVTF		1	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1
433	//	FCVTZS		1	0	0	1	1	1	1	0	1	1	0	1	1	0	0	0
434	//	FCVTZU		1	0	0	1	1	1	1	0	1	1	0	1	1	0	0	1
435	//	**Floating-point conditional**		**M**	**0**	**S**	**1**	**1**	**1**	**1**	**0**	**type**		**1**			**Rm**		
436	//	FCCMP		0	0	0	1	1	1	1	0	0	0	1			Rm		
437	//	FCCMPE		0	0	0	1	1	1	1	0	0	0	1			Rm		
438	//	FCCMP		0	0	0	1	1	1	1	0	0	1	1			Rm		
439	//	FCCMPE		0	0	0	1	1	1	1	0	0	1	1			Rm		
440	//	**Floating-point data-proce**		**M**	**0**	**S**	**1**	**1**	**1**	**1**	**0**	**type**		**1**			**Rm**		
441	//	FMUL		0	0	0	1	1	1	1	0	0	0	1			Rm		
442	//	FDIV		0	0	0	1	1	1	1	0	0	0	1			Rm		
443	//	FADD		0	0	0	1	1	1	1	0	0	0	1			Rm		
444	//	FSUB		0	0	0	1	1	1	1	0	0	0	1			Rm		
445	//	FMAX		0	0	0	1	1	1	1	0	0	0	1			Rm		
446	//	FMIN		0	0	0	1	1	1	1	0	0	0	1			Rm		
447	//	FMAXNM		0	0	0	1	1	1	1	0	0	0	1			Rm		
448	//	FMINNM		0	0	0	1	1	1	1	0	0	0	1			Rm		
449	//	FNMUL		0	0	0	1	1	1	1	0	0	0	1			Rm		
450	//	FMUL		0	0	0	1	1	1	1	0	0	1	1			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
451	//	FDIV		0	0	0	1	1	1	1	0	0	1	1			Rm		
452	//	FADD		0	0	0	1	1	1	1	0	0	1	1			Rm		
453	//	FSUB		0	0	0	1	1	1	1	0	0	1	1			Rm		
454	//	FMAX		0	0	0	1	1	1	1	0	0	1	1			Rm		
455	//	FMIN		0	0	0	1	1	1	1	0	0	1	1			Rm		
456	//	FMAXNM		0	0	0	1	1	1	1	0	0	1	1			Rm		
457	//	FMINNM		0	0	0	1	1	1	1	0	0	1	1			Rm		
458	//	FNMUL		0	0	0	1	1	1	1	0	0	1	1			Rm		
459	//	**Floating-point conditional**		**M**	**0**	**S**	**1**	**1**	**1**	**1**	**0**	**type**		**1**			**Rm**		
460	//	FCSEL		0	0	0	1	1	1	1	0	0	0	1			Rm		
461	//	FCSEL		0	0	0	1	1	1	1	0	0	1	1			Rm		
462	//	**Floating-point immediate**		**M**	**0**	**S**	**1**	**1**	**1**	**1**	**0**	**type**		**1**				**imm8**	
463	//	FMOV		0	0	0	1	1	1	1	0	0	0	1				imm8	
464	//	FMOV		0	0	0	1	1	1	1	0	0	1	1				imm8	
465	//	**Floating-point compare**		**M**	**0**	**S**	**1**	**1**	**1**	**1**	**0**	**type**		**1**			**Rm**		
466	//	FCMP		0	0	0	1	1	1	1	0	0	0	1			Rm		
467	//	FCMP		0	0	0	1	1	1	1	0	0	0	1			Rm		
468	//	FCMPE		0	0	0	1	1	1	1	0	0	0	1			Rm		
469	//	FCMPE		0	0	0	1	1	1	1	0	0	0	1			Rm		
470	//	FCMP		0	0	0	1	1	1	1	0	0	1	1			Rm		
471	//	FCMP		0	0	0	1	1	1	1	0	0	1	1			Rm		
472	//	FCMPE		0	0	0	1	1	1	1	0	0	1	1			Rm		
473	//	FCMPE		0	0	0	1	1	1	1	0	0	1	1			Rm		
474	//	**Floating-point data-proce**		**M**	**0**	**S**	**1**	**1**	**1**	**1**	**0**	**type**		**1**			**opcode**		
475	//	FMOV		0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0
476	//	FABS		0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0
477	//	FNEG		0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	1
478	//	FSQRT		0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	1
479	//	FCVT		0	0	0	1	1	1	1	0	0	0	1	0	0	0	1	0
480	//	FCVT		0	0	0	1	1	1	1	0	0	0	1	0	0	0	1	1
481	//	FRINTN		0	0	0	1	1	1	1	0	0	0	1	0	0	1	0	0
482	//	FRINTP		0	0	0	1	1	1	1	0	0	0	1	0	0	1	0	0
483	//	FRINTM		0	0	0	1	1	1	1	0	0	0	1	0	0	1	0	1
484	//	FRINTZ		0	0	0	1	1	1	1	0	0	0	1	0	0	1	0	1

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
485	//	FRINTA		0	0	0	1	1	1	1	0	0	0	1	0	0	1	1	0
486	//	FRINTX		0	0	0	1	1	1	1	0	0	0	1	0	0	1	1	1
487	//	FRINTI		0	0	0	1	1	1	1	0	0	0	1	0	0	1	1	1
488	//	FMOV		0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	0
489	//	FABS		0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	0
490	//	FNEG		0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1
491	//	FSQRT		0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1
492	//	FCVT		0	0	0	1	1	1	1	0	0	1	1	0	0	0	1	0
493	//	FCVT		0	0	0	1	1	1	1	0	0	1	1	0	0	0	1	1
494	//	FRINTN		0	0	0	1	1	1	1	0	0	1	1	0	0	1	0	0
495	//	FRINTP		0	0	0	1	1	1	1	0	0	1	1	0	0	1	0	0
496	//	FRINTM		0	0	0	1	1	1	1	0	0	1	1	0	0	1	0	1
497	//	FRINTZ		0	0	0	1	1	1	1	0	0	1	1	0	0	1	0	1
498	//	FRINTA		0	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0
499	//	FRINTX		0	0	0	1	1	1	1	0	0	1	1	0	0	1	1	1
500	//	FRINTI		0	0	0	1	1	1	1	0	0	1	1	0	0	1	1	1
501	//	FCVT		0	0	0	1	1	1	1	0	1	1	1	0	0	0	1	0
502	//	FCVT		0	0	0	1	1	1	1	0	1	1	1	0	0	0	1	0
503	//	**Floating-point<->integer c**		**sf**	**0**	**S**	**1**	**1**	**1**	**1**	**0**	**type**		**1**	**rmode**		**opcode**		
504	//	FCVTNS		0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0
505	//	FCVTNU		0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	1
506	//	SCVTF		0	0	0	1	1	1	1	0	0	0	1	0	0	0	1	0
507	//	UCVTF		0	0	0	1	1	1	1	0	0	0	1	0	0	0	1	1
508	//	FCVTAS		0	0	0	1	1	1	1	0	0	0	1	0	0	1	0	0
509	//	FCVTAU		0	0	0	1	1	1	1	0	0	0	1	0	0	1	0	1
510	//	FMOV		0	0	0	1	1	1	1	0	0	0	1	0	0	1	1	0
511	//	FMOV		0	0	0	1	1	1	1	0	0	0	1	0	0	1	1	1
512	//	FCVTPS		0	0	0	1	1	1	1	0	0	0	1	0	1	0	0	0
513	//	FCVTPU		0	0	0	1	1	1	1	0	0	0	1	0	1	0	0	1
514	//	FCVTMS		0	0	0	1	1	1	1	0	0	0	1	1	0	0	0	0
515	//	FCVTMU		0	0	0	1	1	1	1	0	0	0	1	1	0	0	0	1
516	//	FCVTZS		0	0	0	1	1	1	1	0	0	0	1	1	1	0	0	0
517	//	FCVTZU		0	0	0	1	1	1	1	0	0	0	1	1	1	0	0	1
518	//	FCVTNS		0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	0

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
519	//	FCVTNU		0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1
520	//	SCVTF		0	0	0	1	1	1	1	0	0	1	1	0	0	0	1	0
521	//	UCVTF		0	0	0	1	1	1	1	0	0	1	1	0	0	0	1	1
522	//	FCVTAS		0	0	0	1	1	1	1	0	0	1	1	0	0	1	0	0
523	//	FCVTAU		0	0	0	1	1	1	1	0	0	1	1	0	0	1	0	1
524	//	FCVTPS		0	0	0	1	1	1	1	0	0	1	1	0	1	0	0	0
525	//	FCVTPU		0	0	0	1	1	1	1	0	0	1	1	0	1	0	0	1
526	//	FCVTMS		0	0	0	1	1	1	1	0	0	1	1	1	0	0	0	0
527	//	FCVTMU		0	0	0	1	1	1	1	0	0	1	1	1	0	0	0	1
528	//	FCVTZS		0	0	0	1	1	1	1	0	0	1	1	1	1	0	0	0
529	//	FCVTZU		0	0	0	1	1	1	1	0	0	1	1	1	1	0	0	1
530	//	FCVTNS		1	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0
531	//	FCVTNU		1	0	0	1	1	1	1	0	0	0	1	0	0	0	0	1
532	//	SCVTF		1	0	0	1	1	1	1	0	0	0	1	0	0	0	1	0
533	//	UCVTF		1	0	0	1	1	1	1	0	0	0	1	0	0	0	1	1
534	//	FCVTAS		1	0	0	1	1	1	1	0	0	0	1	0	0	1	0	0
535	//	FCVTAU		1	0	0	1	1	1	1	0	0	0	1	0	0	1	0	1
536	//	FCVTPS		1	0	0	1	1	1	1	0	0	0	1	0	1	0	0	0
537	//	FCVTPU		1	0	0	1	1	1	1	0	0	0	1	0	1	0	0	1
538	//	FCVTMS		1	0	0	1	1	1	1	0	0	0	1	1	0	0	0	0
539	//	FCVTMU		1	0	0	1	1	1	1	0	0	0	1	1	0	0	0	1
540	//	FCVTZS		1	0	0	1	1	1	1	0	0	0	1	1	1	0	0	0
541	//	FCVTZU		1	0	0	1	1	1	1	0	0	0	1	1	1	0	0	1
542	//	FCVTNS		1	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0
543	//	FCVTNU		1	0	0	1	1	1	1	0	0	0	1	0	0	0	0	1
544	//	SCVTF		1	0	0	1	1	1	1	0	0	1	1	0	0	0	1	0
545	//	UCVTF		1	0	0	1	1	1	1	0	0	1	1	0	0	0	1	1
546	//	FCVTAS		1	0	0	1	1	1	1	0	0	1	1	0	0	1	0	0
547	//	FCVTAU		1	0	0	1	1	1	1	0	0	1	1	0	0	1	0	1
548	//	FMOV		1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0
549	//	FMOV		1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	1
550	//	FCVTPS		1	0	0	1	1	1	1	0	0	1	1	0	1	0	0	0
551	//	FCVTPU		1	0	0	1	1	1	1	0	0	1	1	0	1	0	0	1
552	//	FCVTMS		1	0	0	1	1	1	1	0	0	1	1	1	0	0	0	0

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
553	//	FCVTMU		1	0	0	1	1	1	1	0	0	1	1	1	0	0	0	1
554	//	FCVTZS		1	0	0	1	1	1	1	0	0	1	1	1	1	0	0	0
555	//	FCVTZU		1	0	0	1	1	1	1	0	0	1	1	1	1	0	0	1
556	//	FMOV		1	0	0	1	1	1	1	0	1	0	1	0	1	1	1	0
557	//	FMOV		1	0	0	1	1	1	1	0	1	0	1	0	1	1	1	1
558	//	**Floating-point data-proce**		**M**	**0**	**S**	**1**	**1**	**1**	**1**	**1**	**type**		**o1**			**Rm**		
559	//	FMADD		0	0	0	1	1	1	1	1	0	0	0			Rm		
560	//	FMSUB		0	0	0	1	1	1	1	1	0	0	0			Rm		
561	//	FNMADD		0	0	0	1	1	1	1	1	0	0	1			Rm		
562	//	FNMSUB		0	0	0	1	1	1	1	1	0	0	1			Rm		
563	//	FMADD		0	0	0	1	1	1	1	1	0	1	0			Rm		
564	//	FMSUB		0	0	0	1	1	1	1	1	0	1	0			Rm		
565	//	FNMADD		0	0	0	1	1	1	1	1	0	1	1			Rm		
566	//	FNMSUB		0	0	0	1	1	1	1	1	0	1	1			Rm		
567	//	**AdvSIMD scalar three san**		**0**	**1**	**U**	**1**	**1**	**1**	**1**	**0**	**size**		**1**			**Rm**		
568	//	SQADD		0	1	0	1	1	1	1	0	-	-	1			Rm		
569	//	SQSUB		0	1	0	1	1	1	1	0	-	-	1			Rm		
570	//	CMGT		0	1	0	1	1	1	1	0	-	-	1			Rm		
571	//	CMGE		0	1	0	1	1	1	1	0	-	-	1			Rm		
572	//	SSHL		0	1	0	1	1	1	1	0	-	-	1			Rm		
573	//	SQSHL		0	1	0	1	1	1	1	0	-	-	1			Rm		
574	//	SRSHL		0	1	0	1	1	1	1	0	-	-	1			Rm		
575	//	SQRSHL		0	1	0	1	1	1	1	0	-	-	1			Rm		
576	//	ADD		0	1	0	1	1	1	1	0	-	-	1			Rm		
577	//	CMTST		0	1	0	1	1	1	1	0	-	-	1			Rm		
578	//	SQDMULH		0	1	0	1	1	1	1	0	-	-	1			Rm		
579	//	FMULX		0	1	0	1	1	1	1	0	0	x	1			Rm		
580	//	FCMEQ		0	1	0	1	1	1	1	0	0	x	1			Rm		
581	//	FRECPS		0	1	0	1	1	1	1	0	0	x	1			Rm		
582	//	FRSQRTS		0	1	0	1	1	1	1	0	1	x	1			Rm		
583	//	UQADD		0	1	1	1	1	1	1	0	-	-	1			Rm		
584	//	UQSUB		0	1	1	1	1	1	1	0	-	-	1			Rm		
585	//	CMHI		0	1	1	1	1	1	1	0	-	-	1			Rm		
586	//	CMHS		0	1	1	1	1	1	1	0	-	-	1			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
587	//	USHL		0	1	1	1	1	1	1	0	-	-	1			Rm		
588	//	UQSHL		0	1	1	1	1	1	1	0	-	-	1			Rm		
589	//	URSHL		0	1	1	1	1	1	1	0	-	-	1			Rm		
590	//	UQRSHL		0	1	1	1	1	1	1	0	-	-	1			Rm		
591	//	SUB		0	1	1	1	1	1	1	0	-	-	1			Rm		
592	//	CMEQ		0	1	1	1	1	1	1	0	-	-	1			Rm		
593	//	SQRDMULH		0	1	1	1	1	1	1	0	-	-	1			Rm		
594	//	FCMGE		0	1	1	1	1	1	1	0	0	x	1			Rm		
595	//	FACGE		0	1	1	1	1	1	1	0	0	x	1			Rm		
596	//	FABD		0	1	1	1	1	1	1	0	1	x	1			Rm		
597	//	FCMGT		0	1	1	1	1	1	1	0	1	x	1			Rm		
598	//	FACGT		0	1	1	1	1	1	1	0	1	x	1			Rm		
599	//	**AdvSIMD scalar three diff**		**0**	**1**	**U**	**1**	**1**	**1**	**1**	**0**	**size**		**1**			**Rm**		
600	//	SQDMLAL	writes to low half of the dest. register	0	1	0	1	1	1	1	0	size		1			Rm		
601	//	SQDMLAL2	writes to high half of the dest. registe	0	1	0	1	1	1	1	0	size		1			Rm		
602	//	SQDMLSL	writes to low half of the dest. register	0	1	0	1	1	1	1	0	size		1			Rm		
603	//	SQDMLSL2	writes to high half of the dest. registe	0	1	0	1	1	1	1	0	size		1			Rm		
604	//	SQDMULL	writes to low half of the dest. register	0	1	0	1	1	1	1	0	size		1			Rm		
605	//	SQDMULL2	writes to high half of the dest. registe	0	1	0	1	1	1	1	0	size		1			Rm		
606	//	**AdvSIMD scalar two-reg r**		**0**	**1**	**U**	**1**	**1**	**1**	**1**	**0**	**size**		**1**	**0**	**0**	**0**	**0**	
607	//	SUQADD		0	1	0	1	1	1	1	0	-	-	1	0	0	0	0	0
608	//	SQABS		0	1	0	1	1	1	1	0	-	-	1	0	0	0	0	0
609	//	CMGT		0	1	0	1	1	1	1	0	-	-	1	0	0	0	0	0
610	//	CMEQ		0	1	0	1	1	1	1	0	-	-	1	0	0	0	0	0
611	//	CMLT		0	1	0	1	1	1	1	0	-	-	1	0	0	0	0	0
612	//	ABS		0	1	0	1	1	1	1	0	-	-	1	0	0	0	0	0
613	//	SQXTN	writes to low half of the dest. register	0	1	0	1	1	1	1	0	-	-	1	0	0	0	0	1
614	//	SQXTN2	writes to high half of the dest. registe	0	1	0	1	1	1	1	0	-	-	1	0	0	0	0	1
615	//	FCVTNS		0	1	0	1	1	1	1	0	0	x	1	0	0	0	0	1
616	//	FCVTMS		0	1	0	1	1	1	1	0	0	x	1	0	0	0	0	1
617	//	FCVTAS		0	1	0	1	1	1	1	0	0	x	1	0	0	0	0	1
618	//	SCVTF		0	1	0	1	1	1	1	0	0	x	1	0	0	0	0	1
619	//	FCMGT		0	1	0	1	1	1	1	0	1	x	1	0	0	0	0	0
620	//	FCMEQ		0	1	0	1	1	1	1	0	1	x	1	0	0	0	0	0

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
621	//	FCMLT		0	1	0	1	1	1	1	0	1	x	1	0	0	0	0	0
622	//	FCVTPS		0	1	0	1	1	1	1	0	1	x	1	0	0	0	0	1
623	//	FCVTZS		0	1	0	1	1	1	1	0	1	x	1	0	0	0	0	1
624	//	FRECPE		0	1	0	1	1	1	1	0	1	x	1	0	0	0	0	1
625	//	FRECPX		0	1	0	1	1	1	1	0	1	x	1	0	0	0	0	1
626	//	USQADD		0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	0
627	//	SQNEG		0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	0
628	//	CMGE		0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	0
629	//	CMLE		0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	0
630	//	NEG		0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	0
631	//	SQXTUN	writes to low half of the dest. register	0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	1
632	//	SQXTUN2	writes to high half of the dest. registe	0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	1
633	//	UQXTN	writes to low half of the dest. register	0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	1
634	//	UQXTN2	writes to high half of the dest. registe	0	1	1	1	1	1	1	0	-	-	1	0	0	0	0	1
635	//	FCVTXN	writes to low half of the dest. register	0	1	1	1	1	1	1	0	0	x	1	0	0	0	0	1
636	//	FCVTXN2	writes to high half of the dest. registe	0	1	1	1	1	1	1	0	0	x	1	0	0	0	0	1
637	//	FCVTNU		0	1	1	1	1	1	1	0	0	x	1	0	0	0	0	1
638	//	FCVTMU		0	1	1	1	1	1	1	0	0	x	1	0	0	0	0	1
639	//	FCVTAU		0	1	1	1	1	1	1	0	0	x	1	0	0	0	0	1
640	//	UCVTF		0	1	1	1	1	1	1	0	0	x	1	0	0	0	0	1
641	//	FCMGE		0	1	1	1	1	1	1	0	1	x	1	0	0	0	0	0
642	//	FCMLE		0	1	1	1	1	1	1	0	1	x	1	0	0	0	0	0
643	//	FCVTPU		0	1	1	1	1	1	1	0	1	x	1	0	0	0	0	1
644	//	FCVTZU		0	1	1	1	1	1	1	0	1	x	1	0	0	0	0	1
645	//	FRSQRTE		0	1	1	1	1	1	1	0	1	x	1	0	0	0	0	1
646	//	**AdvSIMD scalar pairwise**		**0**	**1**	**U**	**1**	**1**	**1**	**1**	**0**	**size**		**1**	**1**	**0**	**0**	**0**	
647	//	ADDP		0	1	0	1	1	1	1	0	-	-	1	1	0	0	0	1
648	//	FMAXNMP		0	1	1	1	1	1	1	0	0	x	1	1	0	0	0	0
649	//	FADDP		0	1	1	1	1	1	1	0	0	x	1	1	0	0	0	0
650	//	FMAXP		0	1	1	1	1	1	1	0	0	x	1	1	0	0	0	0
651	//	FMINNMP		0	1	1	1	1	1	1	0	1	x	1	1	0	0	0	0
652	//	FMINP		0	1	1	1	1	1	1	0	1	x	1	1	0	0	0	0
653	//	**AdvSIMD scalar copy**		**0**	**1**	**op**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**		**imm5**			
654	//	DUP		0	1	0	1	1	1	1	0	0	0	0	-	-	-	-	-

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
655	//	**AdvSIMD scalar x indexe**		**0**	**1**	**U**	**1**	**1**	**1**	**1**	**1**	**size**		**L**	**M**		**Rm**		
656	//	SQDMLAL		0	1	0	1	1	1	1	1	-	-	L	M		Rm		
657	//	SQDMLAL2		0	1	0	1	1	1	1	1	-	-	L	M		Rm		
658	//	SQDMLSL		0	1	0	1	1	1	1	1	-	-	L	M		Rm		
659	//	SQDMLSL2		0	1	0	1	1	1	1	1	-	-	L	M		Rm		
660	//	SQDMULL		0	1	0	1	1	1	1	1	-	-	L	M		Rm		
661	//	SQDMULL2		0	1	0	1	1	1	1	1	-	-	L	M		Rm		
662	//	SQDMULH		0	1	0	1	1	1	1	1	-	-	L	M		Rm		
663	//	SQRDMULH		0	1	0	1	1	1	1	1	-	-	L	M		Rm		
664	//	FMLA		0	1	0	1	1	1	1	1	1	x	L	M		Rm		
665	//	FMLS		0	1	0	1	1	1	1	1	1	x	L	M		Rm		
666	//	FMUL		0	1	0	1	1	1	1	1	1	x	L	M		Rm		
667	//	FMULX		0	1	1	1	1	1	1	1	1	x	L	M		Rm		
668	//	**AdvSIMD scalar shift by i**		**0**	**1**	**U**	**1**	**1**	**1**	**1**	**1**	**0**		**immh**			**immb**		
669	//	SSHR	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
670	//	SSRA	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
671	//	SRSHR	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
672	//	SRSRA	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
673	//	SHL	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
674	//	SQSHL	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
675	//	SQSHRN	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
676	//	SQSHRN2	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
677	//	SQRSHRN	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
678	//	SQRSHRN2	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
679	//	SCVTF	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
680	//	FCVTZS	immh != 0000	0	1	0	1	1	1	1	1	0		immh			immb		
681	//	USHR	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
682	//	USRA	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
683	//	URSHR	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
684	//	URSRA	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
685	//	SRI	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
686	//	SLI	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
687	//	SQSHLU	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
688	//	UQSHL	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
689	//	SQSHRUN	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
690	//	SQSHRUN2	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
691	//	SQRSHRUN	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
692	//	SQRSHRUN2	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
693	//	UQSHRN	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
694	//	UQRSHRN	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
695	//	UQRSHRN2	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
696	//	UCVTF	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
697	//	FCVTZU	immh != 0000	0	1	1	1	1	1	1	1	0		immh			immb		
698	//	**Crypto three-reg SHA**		**0**	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**size**		**0**			**Rm**		
699	//	SHA1C		0	1	0	1	1	1	1	0	0	0	0			Rm		
700	//	SHA1P		0	1	0	1	1	1	1	0	0	0	0			Rm		
701	//	SHA1M		0	1	0	1	1	1	1	0	0	0	0			Rm		
702	//	SHA1SU0		0	1	0	1	1	1	1	0	0	0	0			Rm		
703	//	SHA256H		0	1	0	1	1	1	1	0	0	0	0			Rm		
704	//	SHA256H2		0	1	0	1	1	1	1	0	0	0	0			Rm		
705	//	SHA256SU1		0	1	0	1	1	1	1	0	0	0	0			Rm		
706	//	**Crypto two-reg SHA**		**0**	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**size**		**1**	**0**	**1**	**0**	**0**	
707	//	SHA1H		0	1	0	1	1	1	1	0	0	0	1	0	1	0	0	0
708	//	SHA1SU1		0	1	0	1	1	1	1	0	0	0	1	0	1	0	0	0
709	//	SHA256SU0		0	1	0	1	1	1	1	0	0	0	1	0	1	0	0	0
710	//	**Crypto AES**		**0**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**size**		**1**	**0**	**1**	**0**	**0**	
711	//	AESE		0	1	0	0	1	1	1	0	0	0	1	0	1	0	0	0
712	//	AESD		0	1	0	0	1	1	1	0	0	0	1	0	1	0	0	0
713	//	AESMC		0	1	0	0	1	1	1	0	0	0	1	0	1	0	0	0
714	//	AESIMC		0	1	0	0	1	1	1	0	0	0	1	0	1	0	0	0
715	//	**AdvSIMD three same**		**0**	**Q**	**U**	**0**	**1**	**1**	**1**	**0**	**size**		**1**			**Rm**		
716	//	SHADD		0	Q	0	0	1	1	1	0	-	-	1			Rm		
717	//	SQADD		0	Q	0	0	1	1	1	0	-	-	1			Rm		
718	//	SRHADD		0	Q	0	0	1	1	1	0	-	-	1			Rm		
719	//	SHSUB		0	Q	0	0	1	1	1	0	-	-	1			Rm		
720	//	SQSUB		0	Q	0	0	1	1	1	0	-	-	1			Rm		
721	//	CMGT		0	Q	0	0	1	1	1	0	-	-	1			Rm		
722	//	CMGE		0	Q	0	0	1	1	1	0	-	-	1			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
723	//	SSHL Vector		0	Q	0	0	1	1	1	0	-	-	1			Rm		
724	//	SQSHL		0	Q	0	0	1	1	1	0	-	-	1			Rm		
725	//	SRSHL		0	Q	0	0	1	1	1	0	-	-	1			Rm		
726	//	SQRSHL		0	Q	0	0	1	1	1	0	-	-	1			Rm		
727	//	SMAX		0	Q	0	0	1	1	1	0	-	-	1			Rm		
728	//	SMIN		0	Q	0	0	1	1	1	0	-	-	1			Rm		
729	//	SABD		0	Q	0	0	1	1	1	0	-	-	1			Rm		
730	//	SABA		0	Q	0	0	1	1	1	0	-	-	1			Rm		
731	//	ADD		0	Q	0	0	1	1	1	0	-	-	1			Rm		
732	//	CMTST		0	Q	0	0	1	1	1	0	-	-	1			Rm		
733	//	MLA		0	Q	0	0	1	1	1	0	-	-	1			Rm		
734	//	MUL		0	Q	0	0	1	1	1	0	-	-	1			Rm		
735	//	SMAXP		0	Q	0	0	1	1	1	0	-	-	1			Rm		
736	//	SMINP		0	Q	0	0	1	1	1	0	-	-	1			Rm		
737	//	SQDMULH		0	Q	0	0	1	1	1	0	-	-	1			Rm		
738	//	ADDP		0	Q	0	0	1	1	1	0	-	-	1			Rm		
739	//	FMAXNM		0	Q	0	0	1	1	1	0	0	x	1			Rm		
740	//	FMLA		0	Q	0	0	1	1	1	0	0	x	1			Rm		
741	//	FADD		0	Q	0	0	1	1	1	0	0	x	1			Rm		
742	//	FMULX		0	Q	0	0	1	1	1	0	0	x	1			Rm		
743	//	FCMEQ		0	Q	0	0	1	1	1	0	0	x	1			Rm		
744	//	FMAX		0	Q	0	0	1	1	1	0	0	x	1			Rm		
745	//	FRECPS		0	Q	0	0	1	1	1	0	0	x	1			Rm		
746	//	AND		0	Q	0	0	1	1	1	0	0	0	1			Rm		
747	//	BIC		0	Q	0	0	1	1	1	0	0	1	1			Rm		
748	//	FMINNM		0	Q	0	0	1	1	1	0	1	x	1			Rm		
749	//	FMLS		0	Q	0	0	1	1	1	0	1	x	1			Rm		
750	//	FSUB		0	Q	0	0	1	1	1	0	1	x	1			Rm		
751	//	FMIN		0	Q	0	0	1	1	1	0	1	x	1			Rm		
752	//	FRSQRTS		0	Q	0	0	1	1	1	0	1	x	1			Rm		
753	//	ORR		0	Q	0	0	1	1	1	0	1	0	1			Rm		
754	//	ORN		0	Q	0	0	1	1	1	0	1	1	1			Rm		
755	//	UHADD		0	Q	1	0	1	1	1	0	-	-	1			Rm		
756	//	UQADD		0	Q	1	0	1	1	1	0	-	-	1			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
757	//	URHADD		0	Q	1	0	1	1	1	0	-	-	1			Rm		
758	//	UHSUB		0	Q	1	0	1	1	1	0	-	-	1			Rm		
759	//			0	Q	1	0	1	1	1	0	-	-	1			Rm		
760	//	CMHI		0	Q	1	0	1	1	1	0	-	-	1			Rm		
761	//	CMHS		0	Q	1	0	1	1	1	0	-	-	1			Rm		
762	//	USHL		0	Q	1	0	1	1	1	0	-	-	1			Rm		
763	//	UQSHL		0	Q	1	0	1	1	1	0	-	-	1			Rm		
764	//	URSHL		0	Q	1	0	1	1	1	0	-	-	1			Rm		
765	//	UQRSHL		0	Q	1	0	1	1	1	0	-	-	1			Rm		
766	//	UMAX		0	Q	1	0	1	1	1	0	-	-	1			Rm		
767	//	UMIN		0	Q	1	0	1	1	1	0	-	-	1			Rm		
768	//	UABD		0	Q	1	0	1	1	1	0	-	-	1			Rm		
769	//	UABA		0	Q	1	0	1	1	1	0	-	-	1			Rm		
770	//	SUB		0	Q	1	0	1	1	1	0	-	-	1			Rm		
771	//	CMEQ		0	Q	1	0	1	1	1	0	-	-	1			Rm		
772	//	MLS		0	Q	1	0	1	1	1	0	-	-	1			Rm		
773	//	PMUL		0	Q	1	0	1	1	1	0	-	-	1			Rm		
774	//	UMAXP		0	Q	1	0	1	1	1	0	-	-	1			Rm		
775	//	UMINP		0	Q	1	0	1	1	1	0	-	-	1			Rm		
776	//	SQRDMULH		0	Q	1	0	1	1	1	0	-	-	1			Rm		
777	//	FMAXNMP		0	Q	1	0	1	1	1	0	0	x	1			Rm		
778	//	FADDP		0	Q	1	0	1	1	1	0	0	x	1			Rm		
779	//	FMUL		0	Q	1	0	1	1	1	0	0	x	1			Rm		
780	//	FCMGE		0	Q	1	0	1	1	1	0	0	x	1			Rm		
781	//	FACGE		0	Q	1	0	1	1	1	0	0	x	1			Rm		
782	//	FMAXP		0	Q	1	0	1	1	1	0	0	x	1			Rm		
783	//	FDIV		0	Q	1	0	1	1	1	0	0	x	1			Rm		
784	//	EOR		0	Q	1	0	1	1	1	0	0	0	1			Rm		
785	//	BSL		0	Q	1	0	1	1	1	0	0	1	1			Rm		
786	//	FMINNMP		0	Q	1	0	1	1	1	0	1	x	1			Rm		
787	//	FABD		0	Q	1	0	1	1	1	0	1	x	1			Rm		
788	//	FCMGT		0	Q	1	0	1	1	1	0	1	x	1			Rm		
789	//	FACGT		0	Q	1	0	1	1	1	0	1	x	1			Rm		
790	//	FMINP		0	Q	1	0	1	1	1	0	1	x	1			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
791	//	BIT		0	Q	1	0	1	1	1	0	1	0	1			Rm		
792	//	BIF		0	Q	1	0	1	1	1	0	1	1	1			Rm		
793	//	**AdvSIMD three different**		**0**	**Q**	**U**	**0**	**1**	**1**	**1**	**0**	**size**		**1**			**Rm**		
794	//	SADDL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
795	//	SADDL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
796	//	SADDW	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
797	//	SADDW2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
798	//	SSUBL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
799	//	SSUBL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
800	//	SSUBW	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
801	//	SSUBW2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
802	//	ADDHN	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
803	//	ADDHN2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
804	//	SABAL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
805	//	SABAL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
806	//	SUBHN	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
807	//	SUBHN2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
808	//	SABDL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
809	//	SABDL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
810	//	SMLAL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
811	//	SMLAL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
812	//	SQDMLAL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
813	//	SQDMLAL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
814	//	SMLSL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
815	//	SMLSL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
816	//	SQDMLSL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
817	//	SQDMLSL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
818	//	SMULL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
819	//	SMULL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
820	//	SQDMULL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
821	//	SQDMULL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
822	//	PMULL	writes to low half of the dest. register	0	0	0	0	1	1	1	0	size		1			Rm		
823	//	PMULL2	writes to high half of the dest. registe	0	1	0	0	1	1	1	0	size		1			Rm		
824	//	UADDL	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
825	//	UADDL2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
826	//	UADDW	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
827	//	UADDW2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
828	//	USUBL	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
829	//	USUBL2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
830	//	USUBW	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
831	//	USUBW2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
832	//	RADDHN	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
833	//	RADDHN2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
834	//	UABAL	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
835	//	UABAL2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
836	//	RSUBHN	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
837	//	RSUBHN2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
838	//	UABDL	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
839	//	UABDL2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
840	//	UMLAL	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
841	//	UMLAL2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
842	//	UMLSL	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
843	//	UMLSL2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
844	//	UMULL	writes to low half of the dest. register	0	0	1	0	1	1	1	0	size		1			Rm		
845	//	UMULL2	writes to high half of the dest. registe	0	1	1	0	1	1	1	0	size		1			Rm		
846	//	**AdvSIMD two-reg misc**		**0**	**Q**	**U**	**0**	**1**	**1**	**1**	**0**	**size**		**1**	**0**	**0**	**0**	**0**	
847	//	REV64		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
848	//	REV16		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
849	//	SADDLP		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
850	//	SUQADD		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
851	//	CLS		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
852	//	CNT		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
853	//	SADALP		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
854	//	SQABS		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
855	//	CMGT		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
856	//	CMEQ		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
857	//	CMLT		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0
858	//	ABS		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	0

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
859	//	XTN		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	1
860	//	XTN2		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	1
861	//	SQXTN		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	1
862	//	SQXTN2		0	Q	0	0	1	1	1	0	-	-	1	0	0	0	0	1
863	//	FCVTN		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
864	//	FCVTN2		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
865	//	FCVTL		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
866	//	FCVTL2		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
867	//	FRINTN		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
868	//	FRINTM		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
869	//	FCVTNS		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
870	//	FCVTMS		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
871	//	FCVTAS		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
872	//	SCVTF		0	Q	0	0	1	1	1	0	0	x	1	0	0	0	0	1
873	//	FCMGT		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	0
874	//	FCMEQ		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	0
875	//	FCMLT		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	0
876	//	FABS		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	0
877	//	FRINTP		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	1
878	//	FRINTZ		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	1
879	//	FCVTPS		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	1
880	//	FCVTZS		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	1
881	//	URECPE		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	1
882	//	FRECPE		0	Q	0	0	1	1	1	0	1	x	1	0	0	0	0	1
883	//	REV32		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
884	//	UADDLP		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
885	//	USQADD		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
886	//	CLZ		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
887	//	UADALP		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
888	//	SQNEG		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
889	//	CMGE		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
890	//	CMLE		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
891	//	NEG		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	0
892	//	SQXTUN		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	1

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
893	//	SQXTUN2		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	1
894	//	SHLL		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	1
895	//	SHLL2		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	1
896	//	UQXTN		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	1
897	//	UQXTN2		0	Q	1	0	1	1	1	0	-	-	1	0	0	0	0	1
898	//	FCVTXN		0	Q	1	0	1	1	1	0	0	x	1	0	0	0	0	1
899	//	FCVTXN2		0	Q	1	0	1	1	1	0	0	x	1	0	0	0	0	1
900	//	FRINTA		0	Q	1	0	1	1	1	0	0	x	1	0	0	0	0	1
901	//	FRINTX		0	Q	1	0	1	1	1	0	0	x	1	0	0	0	0	1
902	//	FCVTNU		0	Q	1	0	1	1	1	0	0	x	1	0	0	0	0	1
903	//	FCVTMU		0	Q	1	0	1	1	1	0	0	x	1	0	0	0	0	1
904	//	FCVTAU		0	Q	1	0	1	1	1	0	0	x	1	0	0	0	0	1
905	//	UCVTF		0	Q	1	0	1	1	1	0	0	x	1	0	0	0	0	1
906	//	NOT		0	Q	1	0	1	1	1	0	0	0	1	0	0	0	0	0
907	//	RBIT		0	Q	1	0	1	1	1	0	0	1	1	0	0	0	0	0
908	//	FCMGE		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	0
909	//	FCMLE		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	0
910	//	FNEG		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	0
911	//	FRINTI		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	1
912	//	FCVTPU		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	1
913	//	FCVTZU		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	1
914	//	URSQRTE		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	1
915	//	FRSQRTE		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	1
916	//	FSQRT		0	Q	1	0	1	1	1	0	1	x	1	0	0	0	0	1
917	//	**AdvSIMD across lanes**		**0**	**Q**	**U**	**0**	**1**	**1**	**1**	**0**	**size**		**1**	**1**	**0**	**0**	**0**	
918	//	SADDLV		0	Q	0	0	1	1	1	0	-	-	1	1	0	0	0	0
919	//	SMAXV		0	Q	0	0	1	1	1	0	-	-	1	1	0	0	0	0
920	//	SMINV		0	Q	0	0	1	1	1	0	-	-	1	1	0	0	0	1
921	//	ADDV		0	Q	0	0	1	1	1	0	-	-	1	1	0	0	0	1
922	//	UADDLV		0	Q	1	0	1	1	1	0	-	-	1	1	0	0	0	0
923	//	UMAXV		0	Q	1	0	1	1	1	0	-	-	1	1	0	0	0	0
924	//	UMINV		0	Q	1	0	1	1	1	0	-	-	1	1	0	0	0	1
925	//	FMAXNMV		0	Q	1	0	1	1	1	0	0	x	1	1	0	0	0	0
926	//	FMAXV		0	Q	1	0	1	1	1	0	0	x	1	1	0	0	0	0

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
927	//	FMINNMV		0	Q	1	0	1	1	1	0	1	x	1	1	0	0	0	0
928	//	FMINV		0	Q	1	0	1	1	1	0	1	x	1	1	0	0	0	0
929	//	**AdvSIMD copy**		**0**	**Q**	**op**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**		**imm5**			
930	//	DUP		0	-	0	0	1	1	1	0	0	0	0	-	-	-	-	-
931	//	DUP		0	-	0	0	1	1	1	0	0	0	0	-	-	-	-	-
932	//	SMOV		0	0	0	0	1	1	1	0	0	0	0	-	-	-	-	-
933	//	UMOV		0	0	0	0	1	1	1	0	0	0	0	-	-	-	-	-
934	//	INS		0	1	0	0	1	1	1	0	0	0	0	-	-	-	-	-
935	//	SMOV		0	1	0	0	1	1	1	0	0	0	0	-	-	-	-	-
936	//	UMOV		0	1	0	0	1	1	1	0	0	0	0	-	-	-	-	-
937	//	INS		0	1	1	0	1	1	1	0	0	0	0	-	-	-	-	-
938	//	**AdvSIMD vector x indexe**		**0**	**Q**	**U**	**0**	**1**	**1**	**1**	**1**	**size**		**L**	**M**		**Rm**		
939	//	SMLAL		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
940	//	SMLAL2		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
941	//	SQDMLAL		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
942	//	SQDMLAL2		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
943	//	SMLSL		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
944	//	SMLSL2		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
945	//	SQDMLSL		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
946	//	SQDMLSL2		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
947	//	MUL		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
948	//	SMULL		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
949	//	SMULL2		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
950	//	SQDMULL		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
951	//	SQDMULL2		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
952	//	SQDMULH		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
953	//	SQRDMULH		0	Q	0	0	1	1	1	1	-	-	L	M		Rm		
954	//	FMLA		0	Q	0	0	1	1	1	1	1	x	L	M		Rm		
955	//	FMLS		0	Q	0	0	1	1	1	1	1	x	L	M		Rm		
956	//	FMUL		0	Q	0	0	1	1	1	1	1	x	L	M		Rm		
957	//	MLA		0	Q	1	0	1	1	1	1	-	-	L	M		Rm		
958	//	UMLAL		0	Q	1	0	1	1	1	1	-	-	L	M		Rm		
959	//	UMLAL2		0	Q	1	0	1	1	1	1	-	-	L	M		Rm		
960	//	MLS		0	Q	1	0	1	1	1	1	-	-	L	M		Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
961	//	UMLSL		0	Q	1	0	1	1	1	1	-	-	L	M		Rm		
962	//	UMLSL2		0	Q	1	0	1	1	1	1	-	-	L	M		Rm		
963	//	UMULL		0	Q	1	0	1	1	1	1	-	-	L	M		Rm		
964	//	UMULL2		0	Q	1	0	1	1	1	1	-	-	L	M		Rm		
965	//	FMULX		0	Q	1	0	1	1	1	1	1	x	L	M		Rm		
966	//	**AdvSIMD modified immec**		**0**	**Q**	**op**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**a**	**b**	**c**
967	//	MOVI		0	-	0	0	1	1	1	1	0	0	0	0	0	a	b	c
968	//	ORR		0	-	0	0	1	1	1	1	0	0	0	0	0	a	b	c
969	//	MOVI		0	-	0	0	1	1	1	1	0	0	0	0	0	a	b	c
970	//	ORR		0	-	0	0	1	1	1	1	0	0	0	0	0	a	b	c
971	//	MOVI		0	-	0	0	1	1	1	1	0	0	0	0	0	a	b	c
972	//	MOVI		0	-	0	0	1	1	1	1	0	0	0	0	0	a	b	c
973	//	FMOV		0	-	0	0	1	1	1	1	0	0	0	0	0	a	b	c
974	//	MVNI		0	-	1	0	1	1	1	1	0	0	0	0	0	a	b	c
975	//	BIC		0	-	1	0	1	1	1	1	0	0	0	0	0	a	b	c
976	//	MVNI		0	-	1	0	1	1	1	1	0	0	0	0	0	a	b	c
977	//	BIC		0	-	1	0	1	1	1	1	0	0	0	0	0	a	b	c
978	//	MVNI		0	-	1	0	1	1	1	1	0	0	0	0	0	a	b	c
979	//	MOVI		0	0	1	0	1	1	1	1	0	0	0	0	0	a	b	c
980	//	MOVI		0	1	1	0	1	1	1	1	0	0	0	0	0	a	b	c
981	//	FMOV		0	1	1	0	1	1	1	1	0	0	0	0	0	a	b	c
982	//	**AdvSIMD shift by immedi**		**0**	**Q**	**U**	**0**	**1**	**1**	**1**	**1**	**0**		**immh**			**immb**		
983	//	SSHR		0	Q	0	0	1	1	1	1	0		immh			immb		
984	//	SSRA		0	Q	0	0	1	1	1	1	0		immh			immb		
985	//	SRSHR		0	Q	0	0	1	1	1	1	0		immh			immb		
986	//	SRSRA		0	Q	0	0	1	1	1	1	0		immh			immb		
987	//	SHL		0	Q	0	0	1	1	1	1	0		immh			immb		
988	//	SQSHL		0	Q	0	0	1	1	1	1	0		immh			immb		
989	//	SHRN		0	Q	0	0	1	1	1	1	0		immh			immb		
990	//	SHRN2		0	Q	0	0	1	1	1	1	0		immh			immb		
991	//	RSHRN		0	Q	0	0	1	1	1	1	0		immh			immb		
992	//	RSHRN2		0	Q	0	0	1	1	1	1	0		immh			immb		
993	//	SQSHRN		0	Q	0	0	1	1	1	1	0		immh			immb		
994	//	SQSHRN2		0	Q	0	0	1	1	1	1	0		immh			immb		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
995	//	SQRSHRN		0	Q	0	0	1	1	1	1	0		immh				immb	
996	//	SQRSHRN2		0	Q	0	0	1	1	1	1	0		immh				immb	
997	//	SSHLL		0	Q	0	0	1	1	1	1	0		immh				immb	
998	//	SSHLL2		0	Q	0	0	1	1	1	1	0		immh				immb	
999	//	SCVTF		0	Q	0	0	1	1	1	1	0		immh				immb	
1000	//	FCVTZS		0	Q	0	0	1	1	1	1	0		immh				immb	
1001	//	USHR		0	Q	1	0	1	1	1	1	0		immh				immb	
1002	//	USRA		0	Q	1	0	1	1	1	1	0		immh				immb	
1003	//	URSHR		0	Q	1	0	1	1	1	1	0		immh				immb	
1004	//	URSRA		0	Q	1	0	1	1	1	1	0		immh				immb	
1005	//	SRI		0	Q	1	0	1	1	1	1	0		immh				immb	
1006	//	SLI		0	Q	1	0	1	1	1	1	0		immh				immb	
1007	//	SQSHLU		0	Q	1	0	1	1	1	1	0		immh				immb	
1008	//	UQSHL		0	Q	1	0	1	1	1	1	0		immh				immb	
1009	//	SQSHRUN		0	Q	1	0	1	1	1	1	0		immh				immb	
1010	//	SQSHRUN2		0	Q	1	0	1	1	1	1	0		immh				immb	
1011	//	SQRSHRUN		0	Q	1	0	1	1	1	1	0		immh				immb	
1012	//	SQRSHRUN2		0	Q	1	0	1	1	1	1	0		immh				immb	
1013	//	UQSHRN		0	Q	1	0	1	1	1	1	0		immh				immb	
1014	//	UQRSHRN		0	Q	1	0	1	1	1	1	0		immh				immb	
1015	//	UQRSHRN2		0	Q	1	0	1	1	1	1	0		immh				immb	
1016	//	USHLL		0	Q	1	0	1	1	1	1	0		immh				immb	
1017	//	USHLL2		0	Q	1	0	1	1	1	1	0		immh				immb	
1018	//	UCVTF		0	Q	1	0	1	1	1	1	0		immh				immb	
1019	//	FCVTZU		0	Q	1	0	1	1	1	1	0		immh				immb	
1020	//	**AdvSIMD TBL/TBX**		**0**	**Q**	**0**	**0**	**1**	**1**	**1**	**0**	**op2**		**0**			**Rm**		
1021	//	TBL		0	Q	0	0	1	1	1	0	0	0	0			Rm		
1022	//	TBX		0	Q	0	0	1	1	1	0	0	0	0			Rm		
1023	//	TBL		0	Q	0	0	1	1	1	0	0	0	0			Rm		
1024	//	TBX		0	Q	0	0	1	1	1	0	0	0	0			Rm		
1025	//	TBL		0	Q	0	0	1	1	1	0	0	0	0			Rm		
1026	//	TBX		0	Q	0	0	1	1	1	0	0	0	0			Rm		
1027	//	TBL		0	Q	0	0	1	1	1	0	0	0	0			Rm		
1028	//	TBX		0	Q	0	0	1	1	1	0	0	0	0			Rm		

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1029	//	**AdvSIMD ZIP/UZP/TRN**		**0**	**Q**	**0**	**0**	**1**	**1**	**1**	**0**	**size**		**0**			**Rm**		
1030	//	UZP1		0	Q	0	0	1	1	1	0	size		0			Rm		
1031	//	TRN1		0	Q	0	0	1	1	1	0	size		0			Rm		
1032	//	ZIP1		0	Q	0	0	1	1	1	0	size		0			Rm		
1033	//	UZP2		0	Q	0	0	1	1	1	0	size		0			Rm		
1034	//	TRN2		0	Q	0	0	1	1	1	0	size		0			Rm		
1035	//	ZIP2		0	Q	0	0	1	1	1	0	size		0			Rm		
1036	//	**AdvSIMD EXT**		**0**	**Q**	**1**	**0**	**1**	**1**	**1**	**0**	**op2**		**0**			**Rm**		
1037	//	EXT		0	Q	1	0	1	1	1	0	0	0	0			Rm		
1038	//	**Loads and stores**						**1**		**0**									
1039	//	**AdvSIMD load/store multi**		**0**	**Q**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**L**	**0**	**0**	**0**	**0**	**0**	**0**
1040	//	ST4		0	Q	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1041	//	ST1		0	Q	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1042	//	ST3		0	Q	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1043	//	ST1		0	Q	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1044	//	ST1		0	Q	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1045	//	ST2		0	Q	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1046	//	ST1		0	Q	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1047	//	LD4		0	Q	0	0	1	1	0	0	0	1	0	0	0	0	0	0
1048	//	LD1		0	Q	0	0	1	1	0	0	0	1	0	0	0	0	0	0
1049	//	LD3		0	Q	0	0	1	1	0	0	0	1	0	0	0	0	0	0
1050	//	LD1		0	Q	0	0	1	1	0	0	0	1	0	0	0	0	0	0
1051	//	LD1		0	Q	0	0	1	1	0	0	0	1	0	0	0	0	0	0
1052	//	LD2		0	Q	0	0	1	1	0	0	0	1	0	0	0	0	0	0
1053	//	LD1		0	Q	0	0	1	1	0	0	0	1	0	0	0	0	0	0
1054	//	**AdvSIMD load/store multi**		**0**	**Q**	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**L**	**0**			**Rm**		
1055	//	ST4	Rm != 11111	0	Q	0	0	1	1	0	0	1	0	0			Rm		
1056	//	ST1	Rm != 11111	0	Q	0	0	1	1	0	0	1	0	0			Rm		
1057	//	ST3	Rm != 11111	0	Q	0	0	1	1	0	0	1	0	0			Rm		
1058	//	ST1	Rm != 11111	0	Q	0	0	1	1	0	0	1	0	0			Rm		
1059	//	ST1	Rm != 11111	0	Q	0	0	1	1	0	0	1	0	0			Rm		
1060	//	ST2	Rm != 11111	0	Q	0	0	1	1	0	0	1	0	0			Rm		
1061	//	ST1	Rm != 11111	0	Q	0	0	1	1	0	0	1	0	0			Rm		
1062	//	ST4		0	Q	0	0	1	1	0	0	1	0	0	1	1	1	1	1

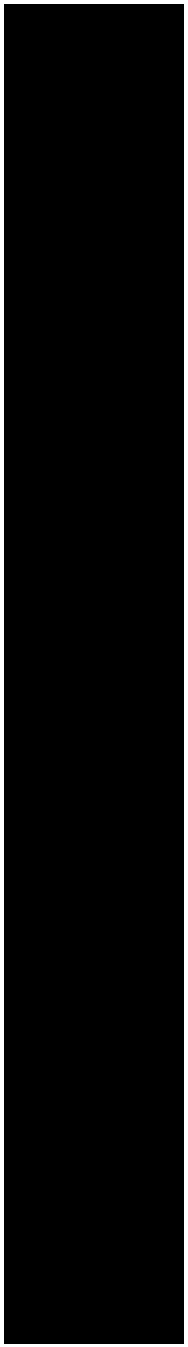
1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1063	//	ST1		0	Q	0	0	1	1	0	0	1	0	0	1	1	1	1	1
1064	//	ST3		0	Q	0	0	1	1	0	0	1	0	0	1	1	1	1	1
1065	//	ST1		0	Q	0	0	1	1	0	0	1	0	0	1	1	1	1	1
1066	//	ST1		0	Q	0	0	1	1	0	0	1	0	0	1	1	1	1	1
1067	//	ST2		0	Q	0	0	1	1	0	0	1	0	0	1	1	1	1	1
1068	//	ST1		0	Q	0	0	1	1	0	0	1	0	0	1	1	1	1	1
1069	//	LD4	Rm != 11111	0	Q	0	0	1	1	0	0	1	1	0			Rm		
1070	//	LD1	Rm != 11111	0	Q	0	0	1	1	0	0	1	1	0			Rm		
1071	//	LD3	Rm != 11111	0	Q	0	0	1	1	0	0	1	1	0			Rm		
1072	//	LD1	Rm != 11111	0	Q	0	0	1	1	0	0	1	1	0			Rm		
1073	//	LD1	Rm != 11111	0	Q	0	0	1	1	0	0	1	1	0			Rm		
1074	//	LD2	Rm != 11111	0	Q	0	0	1	1	0	0	1	1	0			Rm		
1075	//	LD1	Rm != 11111	0	Q	0	0	1	1	0	0	1	1	0			Rm		
1076	//	LD4		0	Q	0	0	1	1	0	0	1	1	0	1	1	1	1	1
1077	//	LD1		0	Q	0	0	1	1	0	0	1	1	0	1	1	1	1	1
1078	//	LD3		0	Q	0	0	1	1	0	0	1	1	0	1	1	1	1	1
1079	//	LD1		0	Q	0	0	1	1	0	0	1	1	0	1	1	1	1	1
1080	//	LD1		0	Q	0	0	1	1	0	0	1	1	0	1	1	1	1	1
1081	//	LD2		0	Q	0	0	1	1	0	0	1	1	0	1	1	1	1	1
1082	//	LD1		0	Q	0	0	1	1	0	0	1	1	0	1	1	1	1	1
1083	//	**AdvSIMD load/store singl**		**0**	**Q**	**0**	**0**	**1**	**1**	**0**	**1**	**0**	**L**	**R**	**0**	**0**	**0**	**0**	**0**
1084	//	ST1		0	Q	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1085	//	ST3		0	Q	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1086	//	ST1		0	Q	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1087	//	ST3		0	Q	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1088	//	ST1		0	Q	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1089	//	ST1		0	Q	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1090	//	ST3		0	Q	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1091	//	ST3		0	Q	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1092	//	ST2		0	Q	0	0	1	1	0	1	0	0	1	0	0	0	0	0
1093	//	ST4		0	Q	0	0	1	1	0	1	0	0	1	0	0	0	0	0
1094	//	ST2		0	Q	0	0	1	1	0	1	0	0	1	0	0	0	0	0
1095	//	ST4		0	Q	0	0	1	1	0	1	0	0	1	0	0	0	0	0
1096	//	ST2		0	Q	0	0	1	1	0	1	0	0	1	0	0	0	0	0

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1097	//	ST2		0	Q	0	0	1	1	0	1	0	0	1	0	0	0	0	0
1098	//	ST4		0	Q	0	0	1	1	0	1	0	0	1	0	0	0	0	0
1099	//	ST4		0	Q	0	0	1	1	0	1	0	0	1	0	0	0	0	0
1100	//	LD1		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1101	//	LD3		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1102	//	LD1		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1103	//	LD3		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1104	//	LD1		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1105	//	LD1		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1106	//	LD3		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1107	//	LD3		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1108	//	LD1R		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1109	//	LD3R		0	Q	0	0	1	1	0	1	0	1	0	0	0	0	0	0
1110	//	LD2		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1111	//	LD4		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1112	//	LD2		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1113	//	LD4		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1114	//	LD2		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1115	//	LD2		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1116	//	LD4		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1117	//	LD4		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1118	//	LD2R		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1119	//	LD4R		0	Q	0	0	1	1	0	1	0	1	1	0	0	0	0	0
1120	//	**AdvSIMD load/store singl**		**0**	**Q**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**L**	**R**			Rm		
1121	//	ST1	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	0			Rm		
1122	//	ST3	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	0			Rm		
1123	//	ST1	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	0			Rm		
1124	//	ST3	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	0			Rm		
1125	//	ST1	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	0			Rm		
1126	//	ST1	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	0			Rm		
1127	//	ST3	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	0			Rm		
1128	//	ST3	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	0			Rm		
1129	//	ST1		0	Q	0	0	1	1	0	1	1	0	0	1	1	1	1	1
1130	//	ST3		0	Q	0	0	1	1	0	1	1	0	0	1	1	1	1	1

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1131	//	ST1		0	Q	0	0	1	1	0	1	1	0	0	1	1	1	1	1
1132	//	ST3		0	Q	0	0	1	1	0	1	1	0	0	1	1	1	1	1
1133	//	ST1		0	Q	0	0	1	1	0	1	1	0	0	1	1	1	1	1
1134	//	ST1		0	Q	0	0	1	1	0	1	1	0	0	1	1	1	1	1
1135	//	ST3		0	Q	0	0	1	1	0	1	1	0	0	1	1	1	1	1
1136	//	ST3		0	Q	0	0	1	1	0	1	1	0	0	1	1	1	1	1
1137	//	ST2	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	1			Rm		
1138	//	ST4	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	1			Rm		
1139	//	ST2	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	1			Rm		
1140	//	ST4	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	1			Rm		
1141	//	ST2	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	1			Rm		
1142	//	ST2	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	1			Rm		
1143	//	ST4	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	1			Rm		
1144	//	ST4	Rm != 11111	0	Q	0	0	1	1	0	1	1	0	1			Rm		
1145	//	ST2		0	Q	0	0	1	1	0	1	1	0	1	1	1	1	1	1
1146	//	ST4		0	Q	0	0	1	1	0	1	1	0	1	1	1	1	1	1
1147	//	ST2		0	Q	0	0	1	1	0	1	1	0	1	1	1	1	1	1
1148	//	ST4		0	Q	0	0	1	1	0	1	1	0	1	1	1	1	1	1
1149	//	ST2		0	Q	0	0	1	1	0	1	1	0	1	1	1	1	1	1
1150	//	ST2		0	Q	0	0	1	1	0	1	1	0	1	1	1	1	1	1
1151	//	ST4		0	Q	0	0	1	1	0	1	1	0	1	1	1	1	1	1
1152	//	ST4		0	Q	0	0	1	1	0	1	1	0	1	1	1	1	1	1
1153	//	LD1	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1154	//	LD3	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1155	//	LD1	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1156	//	LD3	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1157	//	LD1	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1158	//	LD1	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1159	//	LD3	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1160	//	LD3	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1161	//	LD1R	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1162	//	LD3R	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	0			Rm		
1163	//	LD1		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1164	//	LD3		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1

1	in_use	Opcode	comments	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1165	//	LD1		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1166	//	LD3		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1167	//	LD1		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1168	//	LD1		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1169	//	LD3		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1170	//	LD3		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1171	//	LD1R		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1172	//	LD3R		0	Q	0	0	1	1	0	1	1	1	0	1	1	1	1	1
1173	//	LD2	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1174	//	LD4	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1175	//	LD2	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1176	//	LD4	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1177	//	LD2	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1178	//	LD2	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1179	//	LD4	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1180	//	LD4	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1181	//	LD2R	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1182	//	LD4R	Rm != 11111	0	Q	0	0	1	1	0	1	1	1	1			Rm		
1183	//	LD2		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1184	//	LD4		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1185	//	LD2		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1186	//	LD4		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1187	//	LD2		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1188	//	LD2		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1189	//	LD4		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1190	//	LD4		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1191	//	LD2R		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1
1192	//	LD4R		0	Q	0	0	1	1	0	1	1	1	1	1	1	1	1	1

in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
	UNALLOCATED																	
	BAD	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x00000000
	Branch,exception gener																	
	Compare _ Branch (imme	**imm19**													**Rt**			
	CBZ	imm19													Rt			0x34000000
	CBNZ	imm19													Rt			0x35000000
	CBZ	imm19													Rt			0xB4000000
	CBNZ	imm19													Rt			0xB5000000
	Test & branch (immediate				**imm14**										**Rt**			
	TBZ				imm14										Rt			0x36000000
	TBNZ				imm14										Rt			0x37000000
	Conditional branch (imme	**imm19**											**-**		**cond**			
	B_cond	imm19											0		cond			0x54000000
	Exception generation			**imm16**									**-**	**-**	**-**	**-**	**-**	
	SVC			imm16									0	0	0	0	1	0xD4000001
	HVC			imm16									0	0	0	1	0	0xD4000002
	SMC			imm16									0	0	0	1	1	0xD4000003
	BRK			imm16									0	0	0	0	0	0xD4200000
	HLT			imm16									0	0	0	0	0	0xD4400000
	DCPS1			imm16									0	0	0	0	1	0xD4A00001
	DCPS2			imm16									0	0	0	1	0	0xD4A00002
	DCPS3			imm16									0	0	0	1	1	0xD4A00003
	System		**CRn**				**CRm**				**op2**				**Rt**			
	MSR	0	1	0	0		CR_m				op2		1	1	1	1	1	0xD500401F
	HINT	0	0	1	0		CR_m				op2		1	1	1	1	1	0xD503201F
	CLREX	0	0	1	1		CR_m			0	1	0	1	1	1	1	1	0xD503305F
	DSB	0	0	1	1		CR_m			1	0	0	1	1	1	1	1	0xD503309F
	DMB	0	0	1	1		CR_m			1	0	1	1	1	1	1	1	0xD50330BF
	ISB	0	0	1	1		CR_m			1	1	0	1	1	1	1	1	0xD50330DF
	SYS		CR_n				CR_m				op2				Rt			0xD5080000
	MSR		CR_n				CR_m				op2				Rt			0xD5100000
	SYSL		CR_n				CR_m				op2				Rt			0xD5280000
	MRS		CR_n				CR_m				op2				Rt			0xD5300000
	Unconditional branch (reg			**op3**						**Rn**					**op4**			
	BR	0	0	0	0	0	0			Rn			0	0	0	0	0	0xD61F0000
	BLR	0	0	0	0	0	0			Rn			0	0	0	0	0	0xD63F0000
	RET	0	0	0	0	0	0			Rn			0	0	0	0	0	0xD65F0000
	ERET	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0xD69F03E0

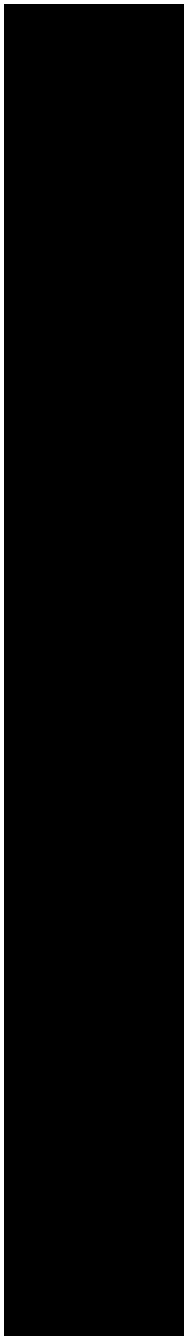


1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
40		DRPS	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0xD6BF03E0
41		**Unconditional branch (im**			**imm26**														
42		B			imm26														0x14000000
43		BL			imm26														0x94000000
44		**Loads and stores**																	
45		**Load/store exclusive**	**-**			**Rt2**					**Rn**					**Rt**			
46		STXRB	0			Rt2					Rn					Rt			0x08000000
47		STLXRB	1			Rt2					Rn					Rt			0x08008000
48		LDXRB	0			Rt2					Rn					Rt			0x08400000
49		LDAXRB	1			Rt2					Rn					Rt			0x08408000
50		STLRB	1			Rt2					Rn					Rt			0x08808000
51		LDARB	1			Rt2					Rn					Rt			0x08C08000
52		STXRH	0			Rt2					Rn					Rt			0x48000000
53		STLXRH	1			Rt2					Rn					Rt			0x48008000
54		LDXRH	0			Rt2					Rn					Rt			0x48400000
55		LDAXRH	1			Rt2					Rn					Rt			0x48408000
56		STLRH	1			Rt2					Rn					Rt			0x48808000
57		LDARH	1			Rt2					Rn					Rt			0x48C08000
58		STXR	0			Rt2					Rn					Rt			0x88000000
59		STLXR	1			Rt2					Rn					Rt			0x88008000
60		STXP	0			Rt2					Rn					Rt			0x88200000
61		STLXP	1			Rt2					Rn					Rt			0x88208000
62		LDXR	0			Rt2					Rn					Rt			0x88400000
63		LDAXR	1			Rt2					Rn					Rt			0x88408000
64		LDXP	0			Rt2					Rn					Rt			0x88600000
65		LDAXP	1			Rt2					Rn					Rt			0x88608000
66		STLR	1			Rt2					Rn					Rt			0x88808000
67		LDAR	1			Rt2					Rn					Rt			0x88C08000
68		STXR	0			Rt2					Rn					Rt			0xC8000000
69		STLXR	1			Rt2					Rn					Rt			0xC8008000
70		STXP	0			Rt2					Rn					Rt			0xC8200000
71		STLXP	1			Rt2					Rn					Rt			0xC8208000
72		LDXR	0			Rt2					Rn					Rt			0xC8400000
73		LDAXR	1			Rt2					Rn					Rt			0xC8408000
74		LDXP	0			Rt2					Rn					Rt			0xC8600000
75		LDAXP	1			Rt2					Rn					Rt			0xC8608000
76		STLR	1			Rt2					Rn					Rt			0xC8808000
77		LDAR	1			Rt2					Rn					Rt			0xC8C08000

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
78		**Load register (literal)**	**imm19**													**Rt**			
79		LDR	imm19													Rt			0x18000000
80		LDR	imm19													Rt			0x1C000000
81		LDR	imm19													Rt			0x58000000
82		LDR	imm19													Rt			0x5C000000
83		LDRSW	imm19													Rt			0x98000000
84		LDR	imm19													Rt			0x9C000000
85		PRFM	imm19													Rt			0xD8000000
86		**Load/store no-allocate pa**				**Rt2**					**Rn**					**Rt**			
87		STNP				Rt2					Rn					Rt			0x28000000
88		LDNP				Rt2					Rn					Rt			0x28400000
89		STNP				Rt2					Rn					Rt			0x2C000000
90		LDNP				Rt2					Rn					Rt			0x2C400000
91		STNP				Rt2					Rn					Rt			0x6C000000
92		LDNP				Rt2					Rn					Rt			0x6C400000
93		STNP				Rt2					Rn					Rt			0xA8000000
94		LDNP				Rt2					Rn					Rt			0xA8400000
95		STNP				Rt2					Rn					Rt			0xAC000000
96		LDNP				Rt2					Rn					Rt			0xAC400000
97		**Load/store register pair (p**				**Rt2**					**Rn**					**Rt**			
98		STP				Rt2					Rn					Rt			0x28800000
99		LDP				Rt2					Rn					Rt			0x28C00000
100		STP				Rt2					Rn					Rt			0x2C800000
101		LDP				Rt2					Rn					Rt			0x2CC00000
102		LDPSW				Rt2					Rn					Rt			0x68C00000
103		STP				Rt2					Rn					Rt			0x6C800000
104		LDP				Rt2					Rn					Rt			0x6CC00000
105		STP				Rt2					Rn					Rt			0xA8800000
106		LDP				Rt2					Rn					Rt			0xA8C00000
107		STP				Rt2					Rn					Rt			0xAC800000
108		LDP				Rt2					Rn					Rt			0xACC00000
109		**Load/store register pair (o**				**Rt2**					**Rn**					**Rt**			
110		STP				Rt2					Rn					Rt			0x29000000
111		LDP				Rt2					Rn					Rt			0x29400000
112		STP				Rt2					Rn					Rt			0x2D000000

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
113		LDP				Rt2					Rn					Rt			0x2D400000
114		LDPSW				Rt2					Rn					Rt			0x69400000
115		STP				Rt2					Rn					Rt			0x6D000000
116		LDP				Rt2					Rn					Rt			0x6D400000
117		STP				Rt2					Rn					Rt			0xA9000000
118		LDP				Rt2					Rn					Rt			0xA9400000
119		STP				Rt2					Rn					Rt			0xAD000000
120		LDP				Rt2					Rn					Rt			0xAD400000
121		**Load/store register pair (p**				**Rt2**					**Rn**					**Rt**			
122		STP				Rt2					Rn					Rt			0x29800000
123		LDP				Rt2					Rn					Rt			0x29C00000
124		STP				Rt2					Rn					Rt			0x2D800000
125		LDP				Rt2					Rn					Rt			0x2DC00000
126		LDPSW				Rt2					Rn					Rt			0x69C00000
127		STP				Rt2					Rn					Rt			0x6D800000
128		LDP				Rt2					Rn					Rt			0x6DC00000
129		STP				Rt2					Rn					Rt			0xA9800000
130		LDP				Rt2					Rn					Rt			0xA9C00000
131		STP				Rt2					Rn					Rt			0xAD800000
132		LDP				Rt2					Rn					Rt			0xADC00000
133		**Load/store register (unsc**					**0**	**0**			**Rn**					**Rt**			
134		STURB					0	0			Rn					Rt			0x38000000
135		LDURB					0	0			Rn					Rt			0x38400000
136		LDURSB					0	0			Rn					Rt			0x38800000
137		LDURSB					0	0			Rn					Rt			0x38C00000
138		STUR					0	0			Rn					Rt			0x3C000000
139		LDUR					0	0			Rn					Rt			0x3C400000
140		STUR					0	0			Rn					Rt			0x3C800000
141		LDUR					0	0			Rn					Rt			0x3CC00000
142		STURH					0	0			Rn					Rt			0x78000000
143		LDURH					0	0			Rn					Rt			0x78400000
144		LDURSH					0	0			Rn					Rt			0x78800000
145		LDURSH					0	0			Rn					Rt			0x78C00000
146		STUR					0	0			Rn					Rt			0x7C000000
147		LDUR					0	0			Rn					Rt			0x7C400000
148		STUR					0	0			Rn					Rt			0xB8000000
149		LDUR					0	0			Rn					Rt			0xB8400000
150		LDURSW					0	0			Rn					Rt			0xB8800000

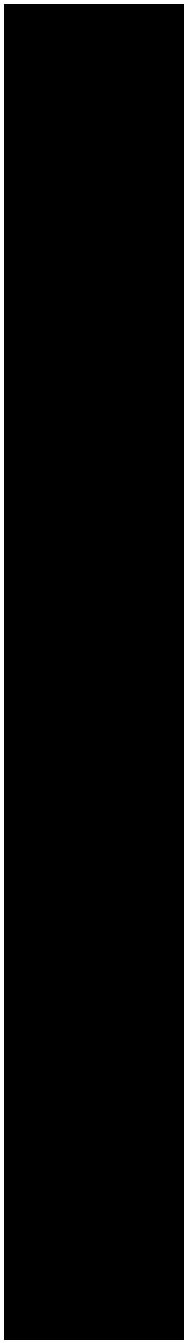
1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
151		STUR	1	0	1	1	0	0			Rn					Rt			0xBC000000
152		LDUR	1	0	1	1	0	0			Rn					Rt			0xBC400000
153		STUR	1	1	1	1	0	0			Rn					Rt			0xF8000000
154		LDUR	1	1	1	1	0	0			Rn					Rt			0xF8400000
155		PRFUM	1	1	1	1	0	0			Rn					Rt			0xF8800000
156		STUR	1	1	1	1	0	0			Rn					Rt			0xFC000000
157		LDUR	1	1	1	1	0	0			Rn					Rt			0xFC400000
158		**Load/store register (imme**	**1**	**1**	**1**	**1**	**0**	**1**			**Rn**					**Rt**			
159		STRB	1	1	1	1	0	1			Rn					Rt			0x38000400
160		LDRB	1	1	1	1	0	1			Rn					Rt			0x38400400
161		LDRSB	1	1	1	1	0	1			Rn					Rt			0x38800400
162		LDRSB	1	1	1	1	0	1			Rn					Rt			0x38C00400
163		STR	1	1	1	1	0	1			Rn					Rt			0x3C000400
164		LDR	1	1	1	1	0	1			Rn					Rt			0x3C400400
165		STR	1	1	1	1	0	1			Rn					Rt			0x3C800400
166		LDR	1	1	1	1	0	1			Rn					Rt			0x3CC00400
167		STRH	1	1	1	1	0	1			Rn					Rt			0x78000400
168		LDRH	1	1	1	1	0	1			Rn					Rt			0x78400400
169		LDRSH	1	1	1	1	0	1			Rn					Rt			0x78800400
170		LDRSH	1	1	1	1	0	1			Rn					Rt			0x78C00400
171		STR	1	1	1	1	0	1			Rn					Rt			0x7C000400
172		LDR	1	1	1	1	0	1			Rn					Rt			0x7C400400
173		STR	1	1	1	1	0	1			Rn					Rt			0xB8000400
174		LDR	1	1	1	1	0	1			Rn					Rt			0xB8400400
175		LDRSW	1	1	1	1	0	1			Rn					Rt			0xB8800400
176		STR	1	1	1	1	0	1			Rn					Rt			0xBC000400
177		LDR	1	1	1	1	0	1			Rn					Rt			0xBC400400
178		STR	1	1	1	1	0	1			Rn					Rt			0xF8000400
179		LDR	1	1	1	1	0	1			Rn					Rt			0xF8400400
180		STR	1	1	1	1	0	1			Rn					Rt			0xFC000400
181		LDR	1	1	1	1	0	1			Rn					Rt			0xFC400400
182		**Load/store register (unpri**	**1**	**1**	**1**	**1**	**1**	**0**			**Rn**					**Rt**			
183		STTRB	1	1	1	1	1	0			Rn					Rt			0x38000800
184		LDTRB	1	1	1	1	1	0			Rn					Rt			0x38400800
185		LDTRSB	1	1	1	1	1	0			Rn					Rt			0x38800800
186		LDTRSB	1	1	1	1	1	0			Rn					Rt			0x38C00800
187		STTRH	1	1	1	1	1	0			Rn					Rt			0x78000800
188		LDTRH	1	1	1	1	1	0			Rn					Rt			0x78400800



1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
189		LDTRSH	3				1	0			Rn					Rt			0x78800800
190		LDTRSH	3				1	0			Rn					Rt			0x78C00800
191		STTR	3				1	0			Rn					Rt			0xB8000800
192		LDTR	3				1	0			Rn					Rt			0xB8400800
193		LDTRSW	3				1	0			Rn					Rt			0xB8800800
194		STTR	3				1	0			Rn					Rt			0xF8000800
195		LDTR	3				1	0			Rn					Rt			0xF8400800
196		**Load/store register (imme**	**3**				**1**	**1**			**Rn**					**Rt**			
197		STRB	3				1	1			Rn					Rt			0x38000C00
198		LDRB	3				1	1			Rn					Rt			0x38400C00
199		LDRSB	3				1	1			Rn					Rt			0x38800C00
200		LDRSB	3				1	1			Rn					Rt			0x38C00C00
201		STR	3				1	1			Rn					Rt			0x3C000C00
202		LDR	3				1	1			Rn					Rt			0x3C400C00
203		STR	3				1	1			Rn					Rt			0x3C800C00
204		LDR	3				1	1			Rn					Rt			0x3CC00C00
205		STRH	3				1	1			Rn					Rt			0x78000C00
206		LDRH	3				1	1			Rn					Rt			0x78400C00
207		LDRSH	3				1	1			Rn					Rt			0x78800C00
208		LDRSH	3				1	1			Rn					Rt			0x78C00C00
209		STR	3				1	1			Rn					Rt			0x7C000C00
210		LDR	3				1	1			Rn					Rt			0x7C400C00
211		STR	3				1	1			Rn					Rt			0xB8000C00
212		LDR	3				1	1			Rn					Rt			0xB8400C00
213		LDRSW	3				1	1			Rn					Rt			0xB8800C00
214		STR	3				1	1			Rn					Rt			0xBC000C00
215		LDR	3				1	1			Rn					Rt			0xBC400C00
216		STR	3				1	1			Rn					Rt			0xF8000C00
217		LDR	3				1	1			Rn					Rt			0xF8400C00
218		STR	3				1	1			Rn					Rt			0xFC000C00
219		LDR	3				1	1			Rn					Rt			0xFC400C00
220		**Load/store register (regis**		**option**			**S**	**1**	**0**		**Rn**					**Rt**			
221		STRB		-	-	-	S	1	0		Rn					Rt			0x38200800
222		LDRB		-	-	-	S	1	0		Rn					Rt			0x38600800
223		LDRSB		-	-	-	S	1	0		Rn					Rt			0x38A00800
224		LDRSB		-	-	-	S	1	0		Rn					Rt			0x38E00800
225		STR		-	-	-	S	1	0		Rn					Rt			0x3C200800
226		LDR		-	-	-	S	1	0		Rn					Rt			0x3C600800

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
227		STR	-	-	-	S	1	0			Rn					Rt			0x3CA00800
228		LDR	-	-	-	S	1	0			Rn					Rt			0x3CE00800
229		STRH	-	-	-	S	1	0			Rn					Rt			0x78200800
230		LDRH	-	-	-	S	1	0			Rn					Rt			0x78600800
231		LDRSH	-	-	-	S	1	0			Rn					Rt			0x78A00800
232		LDRSH	-	-	-	S	1	0			Rn					Rt			0x78E00800
233		STR	-	-	-	S	1	0			Rn					Rt			0x7C200800
234		LDR	-	-	-	S	1	0			Rn					Rt			0x7C600800
235		STR	-	-	-	S	1	0			Rn					Rt			0xB8200800
236		LDR	-	-	-	S	1	0			Rn					Rt			0xB8600800
237		LDRSW	-	-	-	S	1	0			Rn					Rt			0xB8A00800
238		STR	-	-	-	S	1	0			Rn					Rt			0xBC200800
239		LDR	-	-	-	S	1	0			Rn					Rt			0xBC600800
240		STR	-	-	-	S	1	0			Rn					Rt			0xF8200800
241		LDR	-	-	-	S	1	0			Rn					Rt			0xF8600800
242		PRFM	-	-	-	S	1	0			Rn					Rt			0xF8A00800
243		STR	-	-	-	S	1	0			Rn					Rt			0xFC200800
244		LDR	-	-	-	S	1	0			Rn					Rt			0xFC600800
245		**Load/store register (unsig**	**12**								**Rn**					**Rt**			
246		STRB	12								Rn					Rt			0x39000000
247		LDRB	12								Rn					Rt			0x39400000
248		LDRSB	12								Rn					Rt			0x39800000
249		LDRSB	12								Rn					Rt			0x39C00000
250		STR	12								Rn					Rt			0x3D000000
251		LDR	12								Rn					Rt			0x3D400000
252		STR	12								Rn					Rt			0x3D800000
253		LDR	12								Rn					Rt			0x3DC00000
254		STRH	12								Rn					Rt			0x79000000
255		LDRH	12								Rn					Rt			0x79400000
256		LDRSH	12								Rn					Rt			0x79800000
257		LDRSH	12								Rn					Rt			0x79C00000
258		STR	12								Rn					Rt			0x7D000000
259		LDR	12								Rn					Rt			0x7D400000
260		STR	12								Rn					Rt			0xB9000000
261		LDR	12								Rn					Rt			0xB9400000
262		LDRSW	12								Rn					Rt			0xB9800000
263		STR	12								Rn					Rt			0xBD000000
264		LDR	12								Rn					Rt			0xBD400000

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
265		STR	ı12								Rn					Rt			0xF9000000
266		LDR	ı12								Rn					Rt			0xF9400000
267		PRFM	ı12								Rn					Rt			0xF9800000
268		STR	ı12								Rn					Rt			0xFD000000
269		LDR	ı12								Rn					Rt			0xFD400000
270		**Data processing – Imme**																	
271		**PC-rel. addressing**	**immhi**													**Rd**			
272		ADR	immhi													Rd			0x10000000
273		ADRP	immhi													Rd			0x90000000
274		**Add/subtract (immediate)**	**ı12**								**Rn**					**Rd**			
275		ADD	ı12								Rn					Rd			0x11000000
276		ADDS	ı12								Rn					Rd			0x31000000
277		SUB	ı12								Rn					Rd			0x51000000
278		SUBS	ı12								Rn					Rd			0x71000000
279		ADD	ı12								Rn					Rd			0x91000000
280		ADDS	ı12								Rn					Rd			0xB1000000
281		SUB	ı12								Rn					Rd			0xD1000000
282		SUBS	ı12								Rn					Rd			0xF1000000
283		**Logical (immediate)**			**imms**						**Rn**					**Rd**			
284		AND			imms						Rn					Rd			0x12000000
285		ORR			imms						Rn					Rd			0x32000000
286		EOR			imms						Rn					Rd			0x52000000
287		ANDS			imms						Rn					Rd			0x72000000
288		AND			imms						Rn					Rd			0x92000000
289		ORR			imms						Rn					Rd			0xB2000000
290		EOR			imms						Rn					Rd			0xD2000000
291		ANDS			imms						Rn					Rd			0xF2000000
292		**Move wide (immediate)**			**imm16**											**Rd**			
293		MOVN			imm16											Rd			0x12800000
294		MOVZ			imm16											Rd			0x52800000
295		MOVK			imm16											Rd			0x72800000
296		MOVN			imm16											Rd			0x92800000
297		MOVZ			imm16											Rd			0xD2800000
298		MOVK			imm16											Rd			0xF2800000
299		**Bitfield**			**imms**						**Rn**					**Rd**			
300		SBFM			imms						Rn					Rd			0x13000000
301		BFM			imms						Rn					Rd			0x33000000
302		UBFM			imms						Rn					Rd			0x53000000



1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
303		SBFM			imms						Rn					Rd			0x93400000
304		BFM			imms						Rn					Rd			0xB3400000
305		UBFM			imms						Rn					Rd			0xD3400000
306		**Extract**			**imms**						**Rn**					**Rd**			
307		EXTR	0	x	x	x	x	x			Rn					Rd			0x13800000
308		EXTR	-	-	-	-	-	-			Rn					Rd			0x93C00000
309		**Data Processing – regis**																	
310		**Logical (shifted register)**			**imm6**						**Rn**					**Rd**			
311		AND	-	-	-	-	-	-			Rn					Rd			0x0A000000
312		BIC	-	-	-	-	-	-			Rn					Rd			0x0A200000
313		ORR	-	-	-	-	-	-			Rn					Rd			0x2A000000
314		ORN	-	-	-	-	-	-			Rn					Rd			0x2A200000
315		EOR	-	-	-	-	-	-			Rn					Rd			0x4A000000
316		EON	-	-	-	-	-	-			Rn					Rd			0x4A200000
317		ANDS	-	-	-	-	-	-			Rn					Rd			0x6A000000
318		BICS	-	-	-	-	-	-			Rn					Rd			0x6A200000
319		AND	-	-	-	-	-	-			Rn					Rd			0x8A000000
320		BIC	-	-	-	-	-	-			Rn					Rd			0x8A200000
321		ORR	-	-	-	-	-	-			Rn					Rd			0xAA000000
322		ORN	-	-	-	-	-	-			Rn					Rd			0xAA200000
323		EOR	-	-	-	-	-	-			Rn					Rd			0xCA000000
324		EON	-	-	-	-	-	-			Rn					Rd			0xCA200000
325		ANDS	-	-	-	-	-	-			Rn					Rd			0xEA000000
326		BICS	-	-	-	-	-	-			Rn					Rd			0xEA200000
327		**Add/subtract (shifted regi**			**imm6**						**Rn**					**Rd**			
328		ADD	-	-	-	-	-	-			Rn					Rd			0x0B000000
329		ADDS	-	-	-	-	-	-			Rn					Rd			0x2B000000
330		SUB	-	-	-	-	-	-			Rn					Rd			0x4B000000
331		SUBS	-	-	-	-	-	-			Rn					Rd			0x6B000000
332		ADD	-	-	-	-	-	-			Rn					Rd			0x8B000000
333		ADDS	-	-	-	-	-	-			Rn					Rd			0xAB000000
334		SUB	-	-	-	-	-	-			Rn					Rd			0xCB000000
335		SUBS	-	-	-	-	-	-			Rn					Rd			0xEB000000
336		**Add/subtract (extended re**	**option**				**imm3**				**Rn**					**Rd**			
337		ADD	option				-	-	-		Rn					Rd			0x0B200000
338		ADDS	option				-	-	-		Rn					Rd			0x2B200000
339		SUB	option				-	-	-		Rn					Rd			0x4B200000
340		SUBS	option				-	-	-		Rn					Rd			0x6B200000

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
341		ADD		option		-	-	-			Rn					Rd			0x8B200000
342		ADDS		option		-	-	-			Rn					Rd			0xAB200000
343		SUB		option		-	-	-			Rn					Rd			0xCB200000
344		SUBS		option		-	-	-			Rn					Rd			0xEB200000
345		**Add/subtract (with carry)**			**opcode2**						**Rn**					**Rd**			
346		ADC	0	0	0	0	0	0			Rn					Rd			0x1A000000
347		ADCS	0	0	0	0	0	0			Rn					Rd			0x3A000000
348		SBC	0	0	0	0	0	0			Rn					Rd			0x5A000000
349		SBCS	0	0	0	0	0	0			Rn					Rd			0x7A000000
350		ADC	0	0	0	0	0	0			Rn					Rd			0x9A000000
351		ADCS	0	0	0	0	0	0			Rn					Rd			0xBA000000
352		SBC	0	0	0	0	0	0			Rn					Rd			0xDA000000
353		SBCS	0	0	0	0	0	0			Rn					Rd			0xFA000000
354		**Conditional compare (reg**		**cond**			**0**	**o2**			**Rn**			**o3**		**nzcv**			
355		CCMN		cond			0	0			Rn			0		nzcv			0x3A400000
356		CCMN		cond			0	0			Rn			0		nzcv			0xBA400000
357		CCMP		cond			0	0			Rn			0		nzcv			0x7A400000
358		CCMP		cond			0	0			Rn			0		nzcv			0xFA400000
359		**Conditional compare (imr**		**cond**			**1**	**o2**			**Rn**			**o3**		**nzcv**			
360		CCMN		cond			1	0			Rn			0		nzcv			0x3A400800
361		CCMN		cond			1	0			Rn			0		nzcv			0xBA400800
362		CCMP		cond			1	0			Rn			0		nzcv			0x7A400800
363		CCMP		cond			1	0			Rn			0		nzcv			0xFA400800
364		**Conditional select**		**cond**			**op2**				**Rn**					**Rd**			
365		CSEL		cond			0	0			Rn					Rd			0x1A800000
366		CSINC		cond			0	1			Rn					Rd			0x1A800400
367		CSINV		cond			0	0			Rn					Rd			0x5A800000
368		CSNEG		cond			0	1			Rn					Rd			0x5A800400
369		CSEL		cond			0	0			Rn					Rd			0x9A800000
370		CSINC		cond			0	1			Rn					Rd			0x9A800400
371		CSINV		cond			0	0			Rn					Rd			0xDA800000
372		CSNEG		cond			0	1			Rn					Rd			0xDA800400
373		**Data-processing (3 sourc**	**o0**			**Ra**					**Rn**					**Rd**			
374		MADD	0			Ra					Rn					Rd			0x1B000000
375		MADD	0			Ra					Rn					Rd			0x9B000000
376		SMADDL	0			Ra					Rn					Rd			0x9B200000
377		UMADDL	0			Ra					Rn					Rd			0x9BA00000
378		MSUB	1			Ra					Rn					Rd			0x1B008000

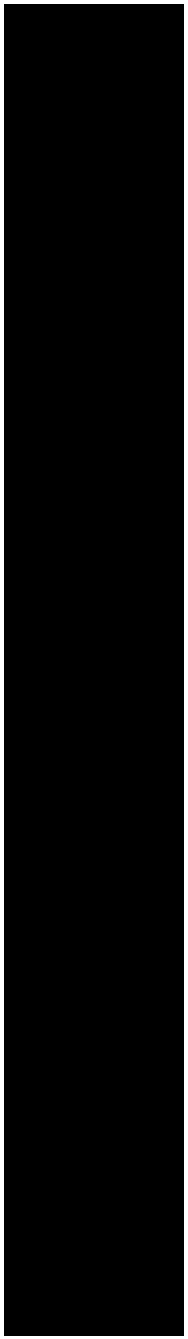
1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
379		MSUB	1			Ra					Rn					Rd			0x9B008000
380		SMSUBL	1			Ra					Rn					Rd			0x9B208000
381		UMSUBL	1			Ra					Rn					Rd			0x9BA08000
382		SMULH	0			Ra					Rn					Rd			0x9B400000
383		UMULH	0			Ra					Rn					Rd			0x9BC00000
384		**Data-processing (2 sourc**			**opcode**						**Rn**					**Rd**			
385		CRC32X	0	1	0	0	1	1			Rn					Rd			0x9AC04C00
386		CRC32CX	0	1	0	1	1	1			Rn					Rd			0x9AC05C00
387		CRC32B	0	1	0	0	0	0			Rn					Rd			0x1AC04000
388		CRC32CB	0	1	0	1	0	0			Rn					Rd			0x1AC05000
389		CRC32H	0	1	0	0	0	1			Rn					Rd			0x1AC04400
390		CRC32CH	0	1	0	1	0	1			Rn					Rd			0x1AC05400
391		CRC32W	0	1	0	0	1	0			Rn					Rd			0x1AC04800
392		CRC32CW	0	1	0	1	1	0			Rn					Rd			0x1AC05800
393		UDIV	0	0	0	0	1	0			Rn					Rd			0x1AC00800
394		UDIV	0	0	0	0	1	0			Rn					Rd			0x9AC00800
395		SDIV	0	0	0	0	1	1			Rn					Rd			0x1AC00C00
396		SDIV	0	0	0	0	1	1			Rn					Rd			0x9AC00C00
397		LSLV	0	0	1	0	0	0			Rn					Rd			0x1AC02000
398		LSLV	0	0	1	0	0	0			Rn					Rd			0x9AC02000
399		LSRV	0	0	1	0	0	1			Rn					Rd			0x1AC02400
400		LSRV	0	0	1	0	0	1			Rn					Rd			0x9AC02400
401		ASRV	0	0	1	0	1	0			Rn					Rd			0x1AC02800
402		ASRV	0	0	1	0	1	0			Rn					Rd			0x9AC02800
403		RORV	0	0	1	0	1	1			Rn					Rd			0x1AC02C00
404		RORV	0	0	1	0	1	1			Rn					Rd			0x9AC02C00
405		**Data-processing (1 sourc**			**opcode**						**Rn**					**Rd**			
406		RBIT	0	0	0	0	0	0			Rn					Rd			0x5AC00000
407		RBIT	0	0	0	0	0	0			Rn					Rd			0xDAC00000
408		CLZ	0	0	0	1	0	0			Rn					Rd			0x5AC01000
409		CLZ	0	0	0	1	0	0			Rn					Rd			0xDAC01000
410		CLS	0	0	0	1	0	1			Rn					Rd			0x5AC01400
411		CLS	0	0	0	1	0	1			Rn					Rd			0xDAC01400
412		REV	0	0	0	0	1	0			Rn					Rd			0x5AC00800
413		REV	0	0	0	0	1	1			Rn					Rd			0xDAC00C00
414		REV16	0	0	0	0	0	1			Rn					Rd			0xDAC00400
415		REV16	0	0	0	0	0	1			Rn					Rd			0x5AC00400
416		REV32	0	0	0	0	1	0			Rn					Rd			0xDAC00800

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
417	//	**Data Processing – SIMD**																	
418	//	**Floating-point<->fixed-po**			**scale**						**Rn**					**Rd**			
419	//	SCVTF	-	-	-	-	-	-			Rn					Rd			0x1E020000
420	//	UCVTF	-	-	-	-	-	-			Rn					Rd			0x1E030000
421	//	FCVTZS	-	-	-	-	-	-			Rn					Rd			0x1ED80000
422	//	FCVTZU	-	-	-	-	-	-			Rn					Rd			0x1ED90000
423	//	SCVTF	-	-	-	-	-	-			Rn					Rd			0x1E020000
424	//	UCVTF	-	-	-	-	-	-			Rn					Rd			0x1E030000
425	//	FCVTZS	-	-	-	-	-	-			Rn					Rd			0x1ED80000
426	//	FCVTZU	-	-	-	-	-	-			Rn					Rd			0x1ED90000
427	//	SCVTF	-	-	-	-	-	-			Rn					Rd			0x9E020000
428	//	UCVTF	-	-	-	-	-	-			Rn					Rd			0x9E030000
429	//	FCVTZS	-	-	-	-	-	-			Rn					Rd			0x9ED80000
430	//	FCVTZU	-	-	-	-	-	-			Rn					Rd			0x9ED90000
431	//	SCVTF	-	-	-	-	-	-			Rn					Rd			0x9E020000
432	//	UCVTF	-	-	-	-	-	-			Rn					Rd			0x9E030000
433	//	FCVTZS	-	-	-	-	-	-			Rn					Rd			0x9ED80000
434	//	FCVTZU	-	-	-	-	-	-			Rn					Rd			0x9ED90000
435	//	**Floating-point conditional**			**cond**		**0**	**1**			**Rn**			**op**		**nzcv**			
436	//	FCCMP			cond		0	1			Rn			0		nzcv			0x1E200400
437	//	FCCMPE			cond		0	1			Rn			1		nzcv			0x1E200410
438	//	FCCMP			cond		0	1			Rn			0		nzcv			0x1E600400
439	//	FCCMPE			cond		0	1			Rn			1		nzcv			0x1E600410
440	//	**Floating-point data-proce**			**opcode**		**1**	**0**			**Rn**					**Rd**			
441	//	FMUL	0	0	0	0	1	0			Rn					Rd			0x1E200800
442	//	FDIV	0	0	0	1	1	0			Rn					Rd			0x1E201800
443	//	FADD	0	0	1	0	1	0			Rn					Rd			0x1E202800
444	//	FSUB	0	0	1	1	1	0			Rn					Rd			0x1E203800
445	//	FMAX	0	1	0	0	1	0			Rn					Rd			0x1E204800
446	//	FMIN	0	1	0	1	1	0			Rn					Rd			0x1E205800
447	//	FMAXNM	0	1	1	0	1	0			Rn					Rd			0x1E206800
448	//	FMINNM	0	1	1	1	1	0			Rn					Rd			0x1E207800
449	//	FNMUL	1	0	0	0	1	0			Rn					Rd			0x1E208800
450	//	FMUL	0	0	0	0	1	0			Rn					Rd			0x1E600800

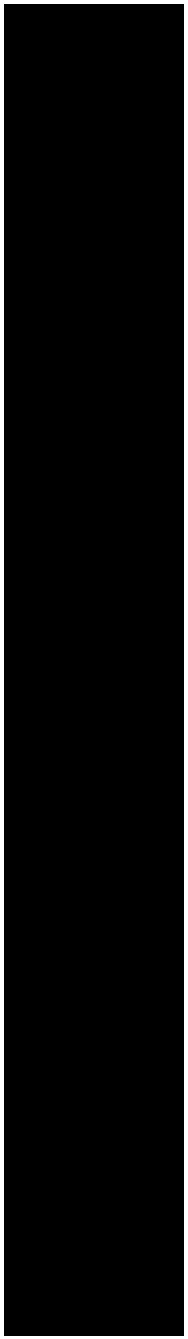
1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
451	//	FDIV	0	0	0	1	1	0			Rn					Rd			0x1E601800
452	//	FADD	0	0	1	0	1	0			Rn					Rd			0x1E602800
453	//	FSUB	0	0	1	1	1	0			Rn					Rd			0x1E603800
454	//	FMAX	0	1	0	0	1	0			Rn					Rd			0x1E604800
455	//	FMIN	0	1	0	1	1	0			Rn					Rd			0x1E605800
456	//	FMAXNM	0	1	1	0	1	0			Rn					Rd			0x1E606800
457	//	FMINNM	0	1	1	1	1	0			Rn					Rd			0x1E607800
458	//	FNMUL	1	0	0	0	1	0			Rn					Rd			0x1E608800
459	//	**Floating-point conditional**		**cond**			**1**	**1**			**Rn**					**Rd**			
460	//	FCSEL		cond			1	1			Rn					Rd			0x1E200C00
461	//	FCSEL		cond			1	1			Rn					Rd			0x1E600C00
462	//	**Floating-point immediate**				**1**	**0**	**0**			**imm5**					**Rd**			
463	//	FMOV				1	0	0	0	0	0	0	0			Rd			0x1E201000
464	//	FMOV				1	0	0	0	0	0	0	0			Rd			0x1E601000
465	//	**Floating-point compare**		**op**		**1**	**0**	**0**	**0**		**Rn**				**opcode2**				
466	//	FCMP	0	0	1	0	0	0			Rn			0	0	0	0	0	0x1E202000
467	//	FCMP	0	0	1	0	0	0			Rn			0	1	0	0	0	0x1E202008
468	//	FCMPE	0	0	1	0	0	0			Rn			1	0	0	0	0	0x1E202010
469	//	FCMPE	0	0	1	0	0	0			Rn			1	1	0	0	0	0x1E202018
470	//	FCMP	0	0	1	0	0	0			Rn			0	0	0	0	0	0x1E602000
471	//	FCMP	0	0	1	0	0	0			Rn			0	1	0	0	0	0x1E602008
472	//	FCMPE	0	0	1	0	0	0			Rn			1	0	0	0	0	0x1E602010
473	//	FCMPE	0	0	1	0	0	0			Rn			1	1	0	0	0	0x1E602018
474	//	**Floating-point data-proce**		**1**	**0**	**0**	**0**	**0**			**Rn**					**Rd**			
475	//	FMOV	0	1	0	0	0	0			Rn					Rd			0x1E204000
476	//	FABS	1	1	0	0	0	0			Rn					Rd			0x1E20C000
477	//	FNEG	0	1	0	0	0	0			Rn					Rd			0x1E214000
478	//	FSQRT	1	1	0	0	0	0			Rn					Rd			0x1E21C000
479	//	FCVT	1	1	0	0	0	0			Rn					Rd			0x1E22C000
480	//	FCVT	1	1	0	0	0	0			Rn					Rd			0x1E23C000
481	//	FRINTN	0	1	0	0	0	0			Rn					Rd			0x1E244000
482	//	FRINTP	1	1	0	0	0	0			Rn					Rd			0x1E24C000
483	//	FRINTM	0	1	0	0	0	0			Rn					Rd			0x1E254000
484	//	FRINTZ	1	1	0	0	0	0			Rn					Rd			0x1E25C000

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
485	//	FRINTA	0	1	0	0	0	0			Rn					Rd			0x1E264000
486	//	FRINTX	0	1	0	0	0	0			Rn					Rd			0x1E274000
487	//	FRINTI	1	1	0	0	0	0			Rn					Rd			0x1E27C000
488	//	FMOV	0	1	0	0	0	0			Rn					Rd			0x1E604000
489	//	FABS	1	1	0	0	0	0			Rn					Rd			0x1E60C000
490	//	FNEG	0	1	0	0	0	0			Rn					Rd			0x1E614000
491	//	FSQRT	1	1	0	0	0	0			Rn					Rd			0x1E61C000
492	//	FCVT	1	1	0	0	0	0			Rn					Rd			0x1E62C000
493	//	FCVT	1	1	0	0	0	0			Rn					Rd			0x1E63C000
494	//	FRINTN	0	1	0	0	0	0			Rn					Rd			0x1E644000
495	//	FRINTP	1	1	0	0	0	0			Rn					Rd			0x1E64C000
496	//	FRINTM	0	1	0	0	0	0			Rn					Rd			0x1E654000
497	//	FRINTZ	1	1	0	0	0	0			Rn					Rd			0x1E65C000
498	//	FRINTA	0	1	0	0	0	0			Rn					Rd			0x1E664000
499	//	FRINTX	0	1	0	0	0	0			Rn					Rd			0x1E674000
500	//	FRINTI	1	1	0	0	0	0			Rn					Rd			0x1E67C000
501	//	FCVT	0	1	0	0	0	0			Rn					Rd			0x1EE24000
502	//	FCVT	1	1	0	0	0	0			Rn					Rd			0x1EE2C000
503	//	**Floating-point<->integer c**	**0**	**0**	**0**	**0**	**0**	**0**			**Rn**					**Rd**			
504	//	FCVTNS	0	0	0	0	0	0			Rn					Rd			0x1E200000
505	//	FCVTNU	0	0	0	0	0	0			Rn					Rd			0x1E210000
506	//	SCVTF	0	0	0	0	0	0			Rn					Rd			0x1E220000
507	//	UCVTF	0	0	0	0	0	0			Rn					Rd			0x1E230000
508	//	FCVTAS	0	0	0	0	0	0			Rn					Rd			0x1E240000
509	//	FCVTAU	0	0	0	0	0	0			Rn					Rd			0x1E250000
510	//	FMOV	0	0	0	0	0	0			Rn					Rd			0x1E260000
511	//	FMOV	0	0	0	0	0	0			Rn					Rd			0x1E270000
512	//	FCVTPS	0	0	0	0	0	0			Rn					Rd			0x1E280000
513	//	FCVTPU	0	0	0	0	0	0			Rn					Rd			0x1E290000
514	//	FCVTMS	0	0	0	0	0	0			Rn					Rd			0x1E300000
515	//	FCVTMU	0	0	0	0	0	0			Rn					Rd			0x1E310000
516	//	FCVTZS	0	0	0	0	0	0			Rn					Rd			0x1E380000
517	//	FCVTZU	0	0	0	0	0	0			Rn					Rd			0x1E390000
518	//	FCVTNS	0	0	0	0	0	0			Rn					Rd			0x1E600000

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
519	//	FCVTNU	0	0	0	0	0	0			Rn					Rd			0x1E610000
520	//	SCVTF	0	0	0	0	0	0			Rn					Rd			0x1E620000
521	//	UCVTF	0	0	0	0	0	0			Rn					Rd			0x1E630000
522	//	FCVTAS	0	0	0	0	0	0			Rn					Rd			0x1E640000
523	//	FCVTAU	0	0	0	0	0	0			Rn					Rd			0x1E650000
524	//	FCVTPS	0	0	0	0	0	0			Rn					Rd			0x1E680000
525	//	FCVTPU	0	0	0	0	0	0			Rn					Rd			0x1E690000
526	//	FCVTMS	0	0	0	0	0	0			Rn					Rd			0x1E700000
527	//	FCVTMU	0	0	0	0	0	0			Rn					Rd			0x1E710000
528	//	FCVTZS	0	0	0	0	0	0			Rn					Rd			0x1E780000
529	//	FCVTZU	0	0	0	0	0	0			Rn					Rd			0x1E790000
530	//	FCVTNS	0	0	0	0	0	0			Rn					Rd			0x9E200000
531	//	FCVTNU	0	0	0	0	0	0			Rn					Rd			0x9E210000
532	//	SCVTF	0	0	0	0	0	0			Rn					Rd			0x9E220000
533	//	UCVTF	0	0	0	0	0	0			Rn					Rd			0x9E230000
534	//	FCVTAS	0	1	0	0	0	0			Rn					Rd			0x9E244000
535	//	FCVTAU	0	0	0	0	0	0			Rn					Rd			0x9E250000
536	//	FCVTPS	0	0	0	0	0	0			Rn					Rd			0x9E280000
537	//	FCVTPU	0	0	0	0	0	0			Rn					Rd			0x9E290000
538	//	FCVTMS	0	0	0	0	0	0			Rn					Rd			0x9E300000
539	//	FCVTMU	1	0	0	0	0	0			Rn					Rd			0x9E318000
540	//	FCVTZS	0	0	0	0	0	0			Rn					Rd			0x9E380000
541	//	FCVTZU	0	0	0	0	0	0			Rn					Rd			0x9E390000
542	//	FCVTNS	0	0	0	0	0	0			Rn					Rd			0x9E200000
543	//	FCVTNU	0	0	0	0	0	0			Rn					Rd			0x9E210000
544	//	SCVTF	0	0	0	0	0	0			Rn					Rd			0x9E620000
545	//	UCVTF	0	0	0	0	0	0			Rn					Rd			0x9E630000
546	//	FCVTAS	0	0	0	0	0	0			Rn					Rd			0x9E640000
547	//	FCVTAU	0	0	0	0	0	0			Rn					Rd			0x9E650000
548	//	FMOV	0	0	0	0	0	0			Rn					Rd			0x9E660000
549	//	FMOV	0	0	0	0	0	0			Rn					Rd			0x9E670000
550	//	FCVTPS	0	0	0	0	0	0			Rn					Rd			0x9E680000
551	//	FCVTPU	0	0	0	0	0	0			Rn					Rd			0x9E690000
552	//	FCVTMS	0	0	0	0	0	0			Rn					Rd			0x9E700000



1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
553	//	FCVTMU	0	0	0	0	0	0			Rn					Rd			0x9E710000
554	//	FCVTZS	0	0	0	0	0	0			Rn					Rd			0x9E780000
555	//	FCVTZU	0	0	0	0	0	0			Rn					Rd			0x9E790000
556	//	FMOV	0	0	0	0	0	0			Rn					Rd			0x9EAE0000
557	//	FMOV	0	0	0	0	0	0			Rn					Rd			0x9EAF0000
558	//	**Floating-point data-proce**	**o0**			**Ra**					**Rn**					**Rd**			
559	//	FMADD	0			Ra					Rn					Rd			0x1F000000
560	//	FMSUB	1			Ra					Rn					Rd			0x1F008000
561	//	FNMADD	0			Ra					Rn					Rd			0x1F200000
562	//	FNMSUB	1			Ra					Rn					Rd			0x1F208000
563	//	FMADD	0			Ra					Rn					Rd			0x1F400000
564	//	FMSUB	1			Ra					Rn					Rd			0x1F408000
565	//	FNMADD	0			Ra					Rn					Rd			0x1F600000
566	//	FNMSUB	1			Ra					Rn					Rd			0x1F608000
567	//	**AdvSIMD scalar three san**		**opcode**				**1**			**Rn**					**Rd**			
568	//	SQADD	0	0	0	0	1	1			Rn					Rd			0x5E200C00
569	//	SQSUB	0	0	1	0	1	1			Rn					Rd			0x5E202C00
570	//	CMGT	0	0	1	1	0	1			Rn					Rd			0x5E203400
571	//	CMGE	0	0	1	1	1	1			Rn					Rd			0x5E203C00
572	//	SSHL	0	1	0	0	0	1			Rn					Rd			0x5E204400
573	//	SQSHL	0	1	0	0	1	1			Rn					Rd			0x5E204C00
574	//	SRSHL	0	1	0	1	0	1			Rn					Rd			0x5E205400
575	//	SQRSHL	0	1	0	1	1	1			Rn					Rd			0x5E205C00
576	//	ADD	1	0	0	0	0	1			Rn					Rd			0x5E208400
577	//	CMTST	1	0	0	0	1	1			Rn					Rd			0x5E208C00
578	//	SQDMULH	1	0	1	1	0	1			Rn					Rd			0x5E20B400
579	//	FMULX	1	1	0	1	1	1			Rn					Rd			0x5E20DC00
580	//	FCMEQ	1	1	1	0	0	1			Rn					Rd			0x5E20E400
581	//	FRECPS	1	1	1	1	1	1			Rn					Rd			0x5E20FC00
582	//	FRSQRTS	1	1	1	1	1	1			Rn					Rd			0x5EA0FC00
583	//	UQADD	0	0	0	0	1	1			Rn					Rd			0x7E200C00
584	//	UQSUB	0	0	1	0	1	1			Rn					Rd			0x7E202C00
585	//	CMHI	0	0	1	1	0	1			Rn					Rd			0x7E203400
586	//	CMHS	0	0	1	1	1	1			Rn					Rd			0x7E203C00



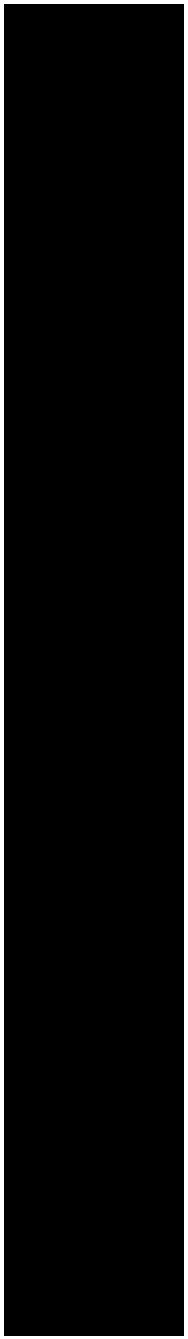
1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
587	//	USHL	0	1	0	0	0	1			Rn					Rd			0x7E204400
588	//	UQSHL	0	1	0	0	1	1			Rn					Rd			0x7E204C00
589	//	URSHL	0	1	0	1	0	1			Rn					Rd			0x7E205400
590	//	UQRSHL	0	1	0	1	1	1			Rn					Rd			0x7E205C00
591	//	SUB	1	0	0	0	0	1			Rn					Rd			0x7E208400
592	//	CMEQ	1	0	0	0	1	1			Rn					Rd			0x7E208C00
593	//	SQRDMULH	1	0	1	1	0	1			Rn					Rd			0x7E20B400
594	//	FCMGE	1	1	1	0	0	1			Rn					Rd			0x7E20E400
595	//	FACGE	1	1	1	0	1	1			Rn					Rd			0x7E20EC00
596	//	FABD	1	1	0	1	0	1			Rn					Rd			0x7EA0D400
597	//	FCMGT	1	1	1	0	0	1			Rn					Rd			0x7EA0E400
598	//	FACGT	1	1	1	0	1	1			Rn					Rd			0x7EA0EC00
599	//	**AdvSIMD scalar three diff**	**opcode**				**0**	**0**			**Rn**					**Rd**			
600	//	SQDMLAL	1	0	0	1	0	0			Rn					Rd			0x5E209000
601	//	SQDMLAL2	1	0	0	1	0	0			Rn					Rd			0x5E209000
602	//	SQDMLSL	1	0	1	1	0	0			Rn					Rd			0x5E20B000
603	//	SQDMLSL2	1	0	1	1	0	0			Rn					Rd			0x5E20B000
604	//	SQDMULL	1	1	0	1	0	0			Rn					Rd			0x5E20D000
605	//	SQDMULL2	1	1	0	1	0	0			Rn					Rd			0x5E20D000
606	//	**AdvSIMD scalar two-reg r**	**opcode**				**1**	**0**			**Rn**					**Rd**			
607	//	SUQADD	0	0	1	1	1	0			Rn					Rd			0x5E203800
608	//	SQABS	0	1	1	1	1	0			Rn					Rd			0x5E207800
609	//	CMGT	1	0	0	0	1	0			Rn					Rd			0x5E208800
610	//	CMEQ	1	0	0	1	1	0			Rn					Rd			0x5E209800
611	//	CMLT	1	0	1	0	1	0			Rn					Rd			0x5E20A800
612	//	ABS	1	0	1	1	1	0			Rn					Rd			0x5E20B800
613	//	SQXTN	0	1	0	0	1	0			Rn					Rd			0x5E214800
614	//	SQXTN2	0	1	0	0	1	0			Rn					Rd			0x5E214800
615	//	FCVTNS	1	0	1	0	1	0			Rn					Rd			0x5E21A800
616	//	FCVTMS	1	0	1	1	1	0			Rn					Rd			0x5E21B800
617	//	FCVTAS	1	1	0	0	1	0			Rn					Rd			0x5E21C800
618	//	SCVTF	1	1	0	1	1	0			Rn					Rd			0x5E21D800
619	//	FCMGT	1	1	0	0	1	0			Rn					Rd			0x5EA0C800
620	//	FCMEQ	1	1	0	1	1	0			Rn					Rd			0x5EA0D800

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
621	//	FCMLT	1	1	1	0	1	0			Rn					Rd			0x5EA0E800
622	//	FCVTPS	1	0	1	0	1	0			Rn					Rd			0x5EA1A800
623	//	FCVTZS	1	0	1	1	1	0			Rn					Rd			0x5EA1B800
624	//	FRECPE	1	1	0	1	1	0			Rn					Rd			0x5EA1D800
625	//	FRECPX	1	1	1	1	1	0			Rn					Rd			0x5EA1F800
626	//	USQADD	0	0	1	1	1	0			Rn					Rd			0x7E203800
627	//	SQNEG	0	1	1	1	1	0			Rn					Rd			0x7E207800
628	//	CMGE	1	0	0	0	1	0			Rn					Rd			0x7E208800
629	//	CMLE	1	0	0	1	1	0			Rn					Rd			0x7E209800
630	//	NEG	1	0	1	1	1	0			Rn					Rd			0x7E20B800
631	//	SQXTUN	0	0	1	0	1	0			Rn					Rd			0x7E212800
632	//	SQXTUN2	0	0	1	0	1	0			Rn					Rd			0x7E212800
633	//	UQXTN	0	1	0	0	1	0			Rn					Rd			0x7E214800
634	//	UQXTN2	0	1	0	0	1	0			Rn					Rd			0x7E214800
635	//	FCVTXN	0	1	1	0	1	0			Rn					Rd			0x7E216800
636	//	FCVTXN2	0	1	1	0	1	0			Rn					Rd			0x7E216800
637	//	FCVTNU	1	0	1	0	1	0			Rn					Rd			0x7E21A800
638	//	FCVTMU	1	0	1	1	1	0			Rn					Rd			0x7E21B800
639	//	FCVTAU	1	1	0	0	1	0			Rn					Rd			0x7E21C800
640	//	UCVTF	1	1	0	1	1	0			Rn					Rd			0x7E21D800
641	//	FCMGE	1	1	0	0	1	0			Rn					Rd			0x7EA0C800
642	//	FCMLE	1	1	0	1	1	0			Rn					Rd			0x7EA0D800
643	//	FCVTPU	1	0	1	0	1	0			Rn					Rd			0x7EA1A800
644	//	FCVTZU	1	0	1	1	1	0			Rn					Rd			0x7EA1B800
645	//	FRSQRTE	1	1	0	1	1	0			Rn					Rd			0x7EA1D800
646	//	**AdvSIMD scalar pairwise**	**opcode**				**1**	**0**			**Rn**					**Rd**			
647	//	ADDP	1	0	1	1	1	0			Rn					Rd			0x5E31B800
648	//	FMAXNMP	1	1	0	0	1	0			Rn					Rd			0x7E30C800
649	//	FADDP	1	1	0	1	1	0			Rn					Rd			0x7E30D800
650	//	FMAXP	1	1	1	1	1	0			Rn					Rd			0x7E30F800
651	//	FMINNMP	1	1	0	0	1	0			Rn					Rd			0x7EB0C800
652	//	FMINP	1	1	1	1	1	0			Rn					Rd			0x7EB0F800
653	//	**AdvSIMD scalar copy**	**0**		**imm4**			**1**			**Rn**					**Rd**			
654	//	DUP	0	0	0	0	0	1			Rn					Rd			0x5E000400

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
655	//	**AdvSIMD scalar x indexed**	**opcode**				**H**	**0**			**Rn**					**Rd**			
656	//	SQDMLAL	0	0	1	1	H	0			Rn					Rd			0x5F003000
657	//	SQDMLAL2	0	0	1	1	H	0			Rn					Rd			0x5F003000
658	//	SQDMLSL	0	1	1	1	H	0			Rn					Rd			0x5F007000
659	//	SQDMLSL2	0	1	1	1	H	0			Rn					Rd			0x5F007000
660	//	SQDMULL	1	0	1	1	H	0			Rn					Rd			0x5F00B000
661	//	SQDMULL2	1	0	1	1	H	0			Rn					Rd			0x5F00B000
662	//	SQDMULH	1	1	0	0	H	0			Rn					Rd			0x5F00C000
663	//	SQRDMULH	1	1	0	1	H	0			Rn					Rd			0x5F00D000
664	//	FMLA	0	0	0	1	H	0			Rn					Rd			0x5F801000
665	//	FMLS	0	1	0	1	H	0			Rn					Rd			0x5F805000
666	//	FMUL	1	0	0	1	H	0			Rn					Rd			0x5F809000
667	//	FMULX	1	0	0	1	H	0			Rn					Rd			0x7F809000
668	//	**AdvSIMD scalar shift by i**	**opcode**					**1**			**Rn**					**Rd**			
669	//	SSHR	0	0	0	0	0	1			Rn					Rd			0x5F000400
670	//	SSRA	0	0	0	1	0	1			Rn					Rd			0x5F001400
671	//	SRSHR	0	0	1	0	0	1			Rn					Rd			0x5F002400
672	//	SRSRA	0	0	1	1	0	1			Rn					Rd			0x5F003400
673	//	SHL	0	1	0	1	0	1			Rn					Rd			0x5F005400
674	//	SQSHL	0	1	1	1	0	1			Rn					Rd			0x5F007400
675	//	SQSHRN	1	0	0	1	0	1			Rn					Rd			0x5F009400
676	//	SQSHRN2	1	0	0	1	0	1			Rn					Rd			0x5F009400
677	//	SQRSHRN	1	0	0	1	1	1			Rn					Rd			0x5F009C00
678	//	SQRSHRN2	1	0	0	1	1	1			Rn					Rd			0x5F009C00
679	//	SCVTF	1	1	1	0	0	1			Rn					Rd			0x5F00E400
680	//	FCVTZS	1	1	1	1	1	1			Rn					Rd			0x5F00FC00
681	//	USHR	0	0	0	0	0	1			Rn					Rd			0x7F000400
682	//	USRA	0	0	0	1	0	1			Rn					Rd			0x7F001400
683	//	URSHR	0	0	1	0	0	1			Rn					Rd			0x7F002400
684	//	URSRA	0	0	1	1	0	1			Rn					Rd			0x7F003400
685	//	SRI	0	1	0	0	0	1			Rn					Rd			0x7F004400
686	//	SLI	0	1	0	1	0	1			Rn					Rd			0x7F005400
687	//	SQSHLU	0	1	1	0	0	1			Rn					Rd			0x7F006400
688	//	UQSHL	0	1	1	1	0	1			Rn					Rd			0x7F007400

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
689	//	SQSHRUN	1	0	0	0	0	1			Rn					Rd			0x7F008400
690	//	SQSHRUN2	1	0	0	0	0	1			Rn					Rd			0x7F008400
691	//	SQRSHRUN	1	0	0	0	1	1			Rn					Rd			0x7F008C00
692	//	SQRSHRUN2	1	0	0	0	1	1			Rn					Rd			0x7F008C00
693	//	UQSHRN	1	0	0	1	0	1			Rn					Rd			0x7F009400
694	//	UQRSHRN	1	0	0	1	1	1			Rn					Rd			0x7F009C00
695	//	UQRSHRN2	1	0	0	1	1	1			Rn					Rd			0x7F009C00
696	//	UCVTF	1	1	1	0	0	1			Rn					Rd			0x7F00E400
697	//	FCVTZU	1	1	1	1	1	1			Rn					Rd			0x7F00FC00
698	//	**Crypto three-reg SHA**	**0**	**opcode**			**0**	**0**			**Rn**					**Rd**			
699	//	SHA1C	0	0	0	0	0	0			Rn					Rd			0x5E000000
700	//	SHA1P	0	0	0	1	0	0			Rn					Rd			0x5E001000
701	//	SHA1M	0	0	1	0	0	0			Rn					Rd			0x5E002000
702	//	SHA1SU0	0	0	1	1	0	0			Rn					Rd			0x5E003000
703	//	SHA256H	0	1	0	0	0	0			Rn					Rd			0x5E004000
704	//	SHA256H2	0	1	0	1	0	0			Rn					Rd			0x5E005000
705	//	SHA256SU1	0	1	1	0	0	0			Rn					Rd			0x5E006000
706	//	**Crypto two-reg SHA**	**opcode**				**1**	**0**			**Rn**					**Rd**			
707	//	SHA1H	0	0	0	0	1	0			Rn					Rd			0x5E280800
708	//	SHA1SU1	0	0	0	1	1	0			Rn					Rd			0x5E281800
709	//	SHA256SU0	0	0	1	0	1	0			Rn					Rd			0x5E282800
710	//	**Crypto AES**	**opcode**				**1**	**0**			**Rn**					**Rd**			
711	//	AESE	0	1	0	0	1	0			Rn					Rd			0x4E284800
712	//	AESD	0	1	0	1	1	0			Rn					Rd			0x4E285800
713	//	AESMC	0	1	1	0	1	0			Rn					Rd			0x4E286800
714	//	AESIMC	0	1	1	1	1	0			Rn					Rd			0x4E287800
715	//	**AdvSIMD three same**		**opcode**				**1**			**Rn**					**Rd**			
716	//	SHADD	0	0	0	0	0	1			Rn					Rd			0x0E200400
717	//	SQADD	0	0	0	0	1	1			Rn					Rd			0x0E200C00
718	//	SRHADD	0	0	0	1	0	1			Rn					Rd			0x0E201400
719	//	SHSUB	0	0	1	0	0	1			Rn					Rd			0x0E202400
720	//	SQSUB	0	0	1	0	1	1			Rn					Rd			0x0E202C00
721	//	CMGT	0	0	1	1	0	1			Rn					Rd			0x0E203400
722	//	CMGE	0	0	1	1	1	1			Rn					Rd			0x0E203C00

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
723	//	SSHL Vector	0	1	0	0	0	1			Rn					Rd			0x0E204400
724	//	SQSHL	0	1	0	0	1	1			Rn					Rd			0x0E204C00
725	//	SRSHL	0	1	0	1	0	1			Rn					Rd			0x0E205400
726	//	SQRSHL	0	1	0	1	1	1			Rn					Rd			0x0E205C00
727	//	SMAX	0	1	1	0	0	1			Rn					Rd			0x0E206400
728	//	SMIN	0	1	1	0	1	1			Rn					Rd			0x0E206C00
729	//	SABD	0	1	1	1	0	1			Rn					Rd			0x0E207400
730	//	SABA	0	1	1	1	1	1			Rn					Rd			0x0E207C00
731	//	ADD	1	0	0	0	0	1			Rn					Rd			0x0E208400
732	//	CMTST	1	0	0	0	1	1			Rn					Rd			0x0E208C00
733	//	MLA	1	0	0	1	0	1			Rn					Rd			0x0E209400
734	//	MUL	1	0	0	1	1	1			Rn					Rd			0x0E209C00
735	//	SMAXP	1	0	1	0	0	1			Rn					Rd			0x0E20A400
736	//	SMINP	1	0	1	0	1	1			Rn					Rd			0x0E20AC00
737	//	SQDMULH	1	0	1	1	0	1			Rn					Rd			0x0E20B400
738	//	ADDP	1	0	1	1	1	1			Rn					Rd			0x0E20BC00
739	//	FMAXNM	1	1	0	0	0	1			Rn					Rd			0x0E20C400
740	//	FMLA	1	1	0	0	1	1			Rn					Rd			0x0E20CC00
741	//	FADD	1	1	0	1	0	1			Rn					Rd			0x0E20D400
742	//	FMULX	1	1	0	1	1	1			Rn					Rd			0x0E20DC00
743	//	FCMEQ	1	1	1	0	0	1			Rn					Rd			0x0E20E400
744	//	FMAX	1	1	1	1	0	1			Rn					Rd			0x0E20F400
745	//	FRECPS	1	1	1	1	1	1			Rn					Rd			0x0E20FC00
746	//	AND	0	0	0	1	1	1			Rn					Rd			0x0E201C00
747	//	BIC	0	0	0	1	1	1			Rn					Rd			0x0E601C00
748	//	FMINNM	1	1	0	0	0	1			Rn					Rd			0x0EA0C400
749	//	FMLS	1	1	0	0	1	1			Rn					Rd			0x0EA0CC00
750	//	FSUB	1	1	0	1	0	1			Rn					Rd			0x0EA0D400
751	//	FMIN	1	1	1	1	0	1			Rn					Rd			0x0EA0F400
752	//	FRSQRTS	1	1	1	1	1	1			Rn					Rd			0x0EA0FC00
753	//	ORR	0	0	0	1	1	1			Rn					Rd			0x0EA01C00
754	//	ORN	0	0	0	1	1	1			Rn					Rd			0x0EE01C00
755	//	UHADD	0	0	0	0	0	1			Rn					Rd			0x2E200400
756	//	UQADD	0	0	0	0	1	1			Rn					Rd			0x2E200C00



1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
757	//	URHADD	0	0	0	1	0	1			Rn					Rd			0x2E201400
758	//	UHSUB	0	0	1	0	0	1			Rn					Rd			0x2E202400
759	//		0	0	1	0	1	1			Rn					Rd			0x2E202C00
760	//	CMHI	0	0	1	1	0	1			Rn					Rd			0x2E203400
761	//	CMHS	0	0	1	1	1	1			Rn					Rd			0x2E203C00
762	//	USHL	0	1	0	0	0	1			Rn					Rd			0x2E204400
763	//	UQSHL	0	1	0	0	1	1			Rn					Rd			0x2E204C00
764	//	URSHL	0	1	0	1	0	1			Rn					Rd			0x2E205400
765	//	UQRSHL	0	1	0	1	1	1			Rn					Rd			0x2E205C00
766	//	UMAX	0	1	1	0	0	1			Rn					Rd			0x2E206400
767	//	UMIN	0	1	1	0	1	1			Rn					Rd			0x2E206C00
768	//	UABD	0	1	1	1	0	1			Rn					Rd			0x2E207400
769	//	UABA	0	1	1	1	1	1			Rn					Rd			0x2E207C00
770	//	SUB	1	0	0	0	0	1			Rn					Rd			0x2E208400
771	//	CMEQ	1	0	0	0	1	1			Rn					Rd			0x2E208C00
772	//	MLS	1	0	0	1	0	1			Rn					Rd			0x2E209400
773	//	PMUL	1	0	0	1	1	1			Rn					Rd			0x2E209C00
774	//	UMAXP	1	0	1	0	0	1			Rn					Rd			0x2E20A400
775	//	UMINP	1	0	1	0	1	1			Rn					Rd			0x2E20AC00
776	//	SQRDMULH	1	0	1	1	0	1			Rn					Rd			0x2E20B400
777	//	FMAXNMP	1	0	1	1	0	1			Rn					Rd			0x2E20B400
778	//	FADDP	1	1	0	1	0	1			Rn					Rd			0x2E20D400
779	//	FMUL	1	1	0	1	1	1			Rn					Rd			0x2E20DC00
780	//	FCMGE	1	1	1	0	0	1			Rn					Rd			0x2E20E400
781	//	FACGE	1	1	1	0	1	1			Rn					Rd			0x2E20EC00
782	//	FMAXP	1	1	1	1	0	1			Rn					Rd			0x2E20F400
783	//	FDIV	1	1	1	1	1	1			Rn					Rd			0x2E20FC00
784	//	EOR	0	0	0	1	1	1			Rn					Rd			0x2E201C00
785	//	BSL	0	0	0	1	1	1			Rn					Rd			0x2E601C00
786	//	FMINNMP	1	1	0	0	0	1			Rn					Rd			0x2EA0C400
787	//	FABD	1	1	0	1	0	1			Rn					Rd			0x2EA0D400
788	//	FCMGT	1	1	1	0	0	1			Rn					Rd			0x2EA0E400
789	//	FACGT	1	1	1	0	1	1			Rn					Rd			0x2EA0EC00
790	//	FMINP	1	1	1	1	0	1			Rn					Rd			0x2EA0F400

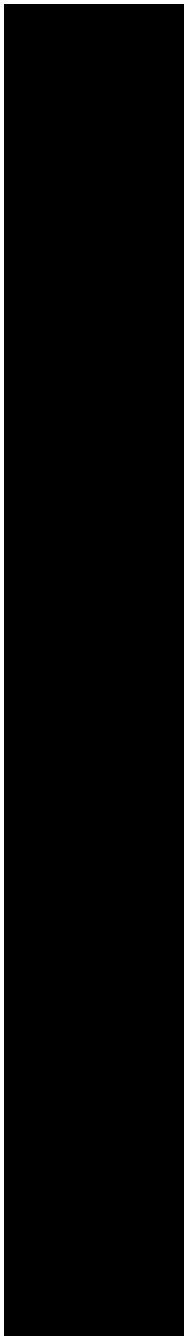
1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
791	//	BIT	0	0	0	1	1	1			Rn					Rd			0x2EA01C00
792	//	BIF	0	0	0	1	1	1			Rn					Rd			0x2EE01C00
793	//	**AdvSIMD three different**		**opcode**			**0**	**0**			**Rn**					**Rd**			
794	//	SADDL	0	0	0	0	0	0			Rn					Rd			0x0E200000
795	//	SADDL2	0	0	0	0	0	0			Rn					Rd			0x4E200000
796	//	SADDW	0	0	0	1	0	0			Rn					Rd			0x0E201000
797	//	SADDW2	0	0	0	1	0	0			Rn					Rd			0x4E201000
798	//	SSUBL	0	0	1	0	0	0			Rn					Rd			0x0E202000
799	//	SSUBL2	0	0	1	0	0	0			Rn					Rd			0x4E202000
800	//	SSUBW	0	0	1	1	0	0			Rn					Rd			0x0E203000
801	//	SSUBW2	0	0	1	1	0	0			Rn					Rd			0x4E203000
802	//	ADDHN	0	1	0	0	0	0			Rn					Rd			0x0E204000
803	//	ADDHN2	0	1	0	0	0	0			Rn					Rd			0x4E204000
804	//	SABAL	0	1	0	1	0	0			Rn					Rd			0x0E205000
805	//	SABAL2	0	1	0	1	0	0			Rn					Rd			0x4E205000
806	//	SUBHN	0	1	1	0	0	0			Rn					Rd			0x0E206000
807	//	SUBHN2	0	1	1	0	0	0			Rn					Rd			0x4E206000
808	//	SABDL	0	1	1	1	0	0			Rn					Rd			0x0E207000
809	//	SABDL2	0	1	1	1	0	0			Rn					Rd			0x4E207000
810	//	SMLAL	1	0	0	0	0	0			Rn					Rd			0x0E208000
811	//	SMLAL2	1	0	0	0	0	0			Rn					Rd			0x4E208000
812	//	SQDMLAL	1	0	0	1	0	0			Rn					Rd			0x0E209000
813	//	SQDMLAL2	1	0	0	1	0	0			Rn					Rd			0x4E209000
814	//	SMLSL	1	0	1	0	0	0			Rn					Rd			0x0E20A000
815	//	SMLSL2	1	0	1	0	0	0			Rn					Rd			0x4E20A000
816	//	SQDMLSL	1	0	1	1	0	0			Rn					Rd			0x0E20B000
817	//	SQDMLSL2	1	0	1	1	0	0			Rn					Rd			0x4E20B000
818	//	SMULL	1	1	0	0	0	0			Rn					Rd			0x0E20C000
819	//	SMULL2	1	1	0	0	0	0			Rn					Rd			0x4E20C000
820	//	SQDMULL	1	1	0	1	0	0			Rn					Rd			0x0E20D000
821	//	SQDMULL2	1	1	0	1	0	0			Rn					Rd			0x4E20D000
822	//	PMULL	1	1	1	0	0	0			Rn					Rd			0x0E20E000
823	//	PMULL2	1	1	1	0	0	0			Rn					Rd			0x4E20E000
824	//	UADDL	0	0	0	0	0	0			Rn					Rd			0x2E200000

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
825	//	UADDL2	0	0	0	0	0	0			Rn					Rd			0x6E200000
826	//	UADDW	0	0	0	1	0	0			Rn					Rd			0x2E201000
827	//	UADDW2	0	0	0	1	0	0			Rn					Rd			0x6E201000
828	//	USUBL	0	0	1	0	0	0			Rn					Rd			0x2E202000
829	//	USUBL2	0	0	1	0	0	0			Rn					Rd			0x6E202000
830	//	USUBW	0	0	1	1	0	0			Rn					Rd			0x2E203000
831	//	USUBW2	0	0	1	1	0	0			Rn					Rd			0x6E203000
832	//	RADDHN	0	1	0	0	0	0			Rn					Rd			0x2E204000
833	//	RADDHN2	0	1	0	0	0	0			Rn					Rd			0x6E204000
834	//	UABAL	0	1	0	1	0	0			Rn					Rd			0x2E205000
835	//	UABAL2	0	1	0	1	0	0			Rn					Rd			0x6E205000
836	//	RSUBHN	0	1	1	0	0	0			Rn					Rd			0x2E206000
837	//	RSUBHN2	0	1	1	0	0	0			Rn					Rd			0x6E206000
838	//	UABDL	0	1	1	1	0	0			Rn					Rd			0x2E207000
839	//	UABDL2	0	1	1	1	0	0			Rn					Rd			0x6E207000
840	//	UMLAL	1	0	0	0	0	0			Rn					Rd			0x2E208000
841	//	UMLAL2	1	0	0	0	0	0			Rn					Rd			0x6E208000
842	//	UMLSL	1	0	1	0	0	0			Rn					Rd			0x2E20A000
843	//	UMLSL2	1	0	1	0	0	0			Rn					Rd			0x6E20A000
844	//	UMULL	1	1	0	0	0	0			Rn					Rd			0x2E20C000
845	//	UMULL2	1	1	0	0	0	0			Rn					Rd			0x6E20C000
846	//	**AdvSIMD two-reg misc**	**opcode**				**1**	**0**			**Rn**					**Rd**			
847	//	REV64	0	0	0	0	1	0			Rn					Rd			0x0E200800
848	//	REV16	0	0	0	1	1	0			Rn					Rd			0x0E201800
849	//	SADDLP	0	0	1	0	1	0			Rn					Rd			0x0E202800
850	//	SUQADD	0	0	1	1	1	0			Rn					Rd			0x0E203800
851	//	CLS	0	1	0	0	1	0			Rn					Rd			0x0E204800
852	//	CNT	0	1	0	1	1	0			Rn					Rd			0x0E205800
853	//	SADALP	0	1	1	0	1	0			Rn					Rd			0x0E206800
854	//	SQABS	0	1	1	1	1	0			Rn					Rd			0x0E207800
855	//	CMGT	1	0	0	0	1	0			Rn					Rd			0x0E208800
856	//	CMEQ	1	0	0	1	1	0			Rn					Rd			0x0E209800
857	//	CMLT	1	0	1	0	1	0			Rn					Rd			0x0E20A800
858	//	ABS	1	0	1	1	1	0			Rn					Rd			0x0E20B800

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
859	//	XTN	0	0	1	0	1	0			Rn					Rd			0x0E212800
860	//	XTN2	0	0	1	0	1	0			Rn					Rd			0x0E212800
861	//	SQXTN	0	1	0	0	1	0			Rn					Rd			0x0E214800
862	//	SQXTN2	0	1	0	0	1	0			Rn					Rd			0x0E214800
863	//	FCVTN	0	1	1	0	1	0			Rn					Rd			0x0E216800
864	//	FCVTN2	0	1	1	0	1	0			Rn					Rd			0x0E216800
865	//	FCVTL	0	1	1	1	1	0			Rn					Rd			0x0E217800
866	//	FCVTL2	0	1	1	1	1	0			Rn					Rd			0x0E217800
867	//	FRINTN	1	0	0	0	1	0			Rn					Rd			0x0E218800
868	//	FRINTM	1	0	0	1	1	0			Rn					Rd			0x0E219800
869	//	FCVTNS	1	0	1	0	1	0			Rn					Rd			0x0E21A800
870	//	FCVTMS	1	0	1	1	1	0			Rn					Rd			0x0E21B800
871	//	FCVTAS	1	1	0	0	1	0			Rn					Rd			0x0E21C800
872	//	SCVTF	1	1	0	1	1	0			Rn					Rd			0x0E21D800
873	//	FCMGT	1	1	0	0	1	0			Rn					Rd			0x0EA0C800
874	//	FCMEQ	1	1	0	1	1	0			Rn					Rd			0x0EA0D800
875	//	FCMLT	1	1	1	0	1	0			Rn					Rd			0x0EA0E800
876	//	FABS	1	1	1	1	1	0			Rn					Rd			0x0EA0F800
877	//	FRINTP	1	0	0	0	1	0			Rn					Rd			0x0EA18800
878	//	FRINTZ	1	0	0	1	1	0			Rn					Rd			0x0EA19800
879	//	FCVTPS	1	0	1	0	1	0			Rn					Rd			0x0EA1A800
880	//	FCVTZS	1	0	1	1	1	0			Rn					Rd			0x0EA1B800
881	//	URECPE	1	1	0	0	1	0			Rn					Rd			0x0EA1C800
882	//	FRECPE	1	1	0	1	1	0			Rn					Rd			0x0EA1D800
883	//	REV32	0	0	0	0	1	0			Rn					Rd			0x2E200800
884	//	UADDLP	0	0	1	0	1	0			Rn					Rd			0x2E202800
885	//	USQADD	0	0	1	1	1	0			Rn					Rd			0x2E203800
886	//	CLZ	0	1	0	0	1	0			Rn					Rd			0x2E204800
887	//	UADALP	0	1	1	0	1	0			Rn					Rd			0x2E206800
888	//	SQNEG	0	1	1	1	1	0			Rn					Rd			0x2E207800
889	//	CMGE	1	0	0	0	1	0			Rn					Rd			0x2E208800
890	//	CMLE	1	0	0	1	1	0			Rn					Rd			0x2E209800
891	//	NEG	1	0	1	1	1	0			Rn					Rd			0x2E20B800
892	//	SQXTUN	0	0	1	0	1	0			Rn					Rd			0x2E212800

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
893	//	SQXTUN2	0	0	1	0	1	0			Rn					Rd			0x2E212800
894	//	SHLL	0	0	1	1	1	0			Rn					Rd			0x2E213800
895	//	SHLL2	0	0	1	1	1	0			Rn					Rd			0x2E213800
896	//	UQXTN	0	1	0	0	1	0			Rn					Rd			0x2E214800
897	//	UQXTN2	0	1	0	0	1	0			Rn					Rd			0x2E214800
898	//	FCVTXN	0	1	1	0	1	0			Rn					Rd			0x2E216800
899	//	FCVTXN2	0	1	1	0	1	0			Rn					Rd			0x2E216800
900	//	FRINTA	1	0	0	0	1	0			Rn					Rd			0x2E218800
901	//	FRINTX	1	0	0	1	1	0			Rn					Rd			0x2E219800
902	//	FCVTNU	1	0	1	0	1	0			Rn					Rd			0x2E21A800
903	//	FCVTMU	1	0	1	1	1	0			Rn					Rd			0x2E21B800
904	//	FCVTAU	1	1	0	0	1	0			Rn					Rd			0x2E21C800
905	//	UCVTF	1	1	0	1	1	0			Rn					Rd			0x2E21D800
906	//	NOT	0	1	0	1	1	0			Rn					Rd			0x2E205800
907	//	RBIT	0	1	0	1	1	0			Rn					Rd			0x2E605800
908	//	FCMGE	1	1	0	0	1	0			Rn					Rd			0x2EA0C800
909	//	FCMLE	1	1	0	1	1	0			Rn					Rd			0x2EA0D800
910	//	FNEG	1	1	1	1	1	0			Rn					Rd			0x2EA0F800
911	//	FRINTI	1	0	0	1	1	0			Rn					Rd			0x2EA19800
912	//	FCVTPU	1	0	1	0	1	0			Rn					Rd			0x2EA1A800
913	//	FCVTZU	1	0	1	1	1	0			Rn					Rd			0x2EA1B800
914	//	URSQRTE	1	1	0	0	1	0			Rn					Rd			0x2EA1C800
915	//	FRSQRTE	1	1	0	1	1	0			Rn					Rd			0x2EA1D800
916	//	FSQRT	1	1	1	1	1	0			Rn					Rd			0x2EA1F800
917	//	**AdvSIMD across lanes**	**opcode**				**1**	**0**			**Rn**					**Rd**			
918	//	SADDLV	0	0	1	1	1	0			Rn					Rd			0x0E303800
919	//	SMAXV	1	0	1	0	1	0			Rn					Rd			0x0E30A800
920	//	SMINV	1	0	1	0	1	0			Rn					Rd			0x0E31A800
921	//	ADDV	1	0	1	1	1	0			Rn					Rd			0x0E31B800
922	//	UADDLV	0	0	1	1	1	0			Rn					Rd			0x2E303800
923	//	UMAXV	1	0	1	0	1	0			Rn					Rd			0x2E30A800
924	//	UMINV	1	0	1	0	1	0			Rn					Rd			0x2E31A800
925	//	FMAXNMV	1	1	0	0	1	0			Rn					Rd			0x2E30C800
926	//	FMAXV	1	1	1	1	1	0			Rn					Rd			0x2E30F800

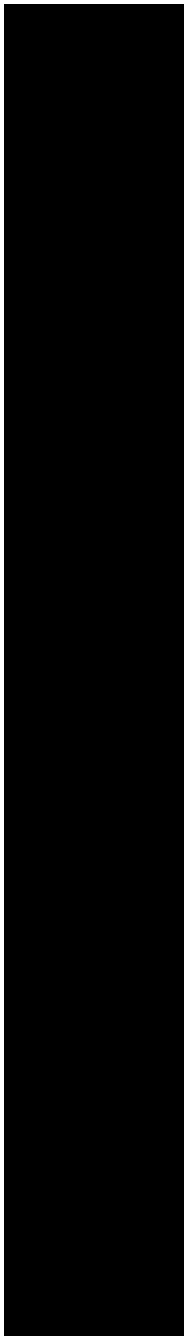
1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
927	//	FMINNMV	1	1	0	0	1	0			Rn					Rd			0x2EB0C800
928	//	FMINV	1	1	1	1	1	0			Rn					Rd			0x2EB0F800
929	//	**AdvSIMD copy**	**0**			**imm4**		**1**			**Rn**					**Rd**			
930	//	DUP	0	0	0	0	0	1			Rn					Rd			0x0E000400
931	//	DUP	0	0	0	0	1	1			Rn					Rd			0x0E000C00
932	//	SMOV	0	0	1	0	1	1			Rn					Rd			0x0E002C00
933	//	UMOV	0	0	1	1	1	1			Rn					Rd			0x0E003C00
934	//	INS	0	0	0	1	1	1			Rn					Rd			0x4E001C00
935	//	SMOV	0	0	1	0	1	1			Rn					Rd			0x4E002C00
936	//	UMOV	0	0	1	1	1	1			Rn					Rd			0x4E003C00
937	//	INS	0	-	-	-	-	1			Rn					Rd			0x6E000400
938	//	**AdvSIMD vector x indexe**		**opcode**			**H**	**0**			**Rn**					**Rd**			
939	//	SMLAL	0	0	1	0	H	0			Rn					Rd			0x0F002000
940	//	SMLAL2	0	0	1	0	H	0			Rn					Rd			0x0F002000
941	//	SQDMLAL	0	0	1	1	H	0			Rn					Rd			0x0F003000
942	//	SQDMLAL2	0	0	1	1	H	0			Rn					Rd			0x0F003000
943	//	SMLSL	0	1	1	0	H	0			Rn					Rd			0x0F006000
944	//	SMLSL2	0	1	1	0	H	0			Rn					Rd			0x0F006000
945	//	SQDMLSL	0	1	1	1	H	0			Rn					Rd			0x0F007000
946	//	SQDMLSL2	0	1	1	1	H	0			Rn					Rd			0x0F007000
947	//	MUL	1	0	0	0	H	0			Rn					Rd			0x0F008000
948	//	SMULL	1	0	1	0	H	0			Rn					Rd			0x0F00A000
949	//	SMULL2	1	0	1	0	H	0			Rn					Rd			0x0F00A000
950	//	SQDMULL	1	0	1	1	H	0			Rn					Rd			0x0F00B000
951	//	SQDMULL2	1	0	1	1	H	0			Rn					Rd			0x0F00B000
952	//	SQDMULH	1	1	0	0	H	0			Rn					Rd			0x0F00C000
953	//	SQRDMULH	1	1	0	1	H	0			Rn					Rd			0x0F00D000
954	//	FMLA	0	0	0	1	H	0			Rn					Rd			0x0F801000
955	//	FMLS	0	1	0	1	H	0			Rn					Rd			0x0F805000
956	//	FMUL	1	0	0	1	H	0			Rn					Rd			0x0F809000
957	//	MLA	0	0	0	0	H	0			Rn					Rd			0x2F000000
958	//	UMLAL	0	0	1	0	H	0			Rn					Rd			0x2F002000
959	//	UMLAL2	0	0	1	0	H	0			Rn					Rd			0x2F002000
960	//	MLS	0	1	0	0	H	0			Rn					Rd			0x2F004000



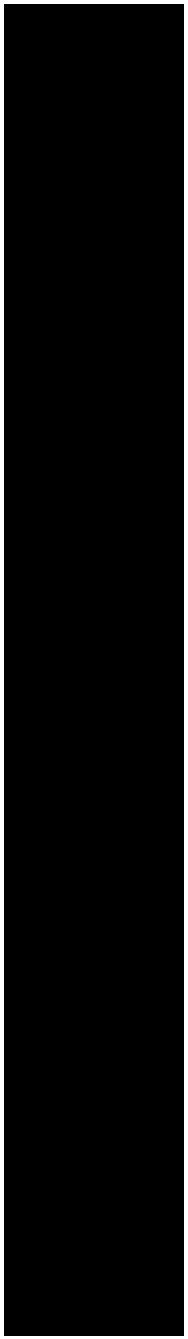
1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
961	//	UMLSL	0	1	1	0	H	0			Rn					Rd			0x2F006000
962	//	UMLSL2	0	1	1	0	H	0			Rn					Rd			0x2F006000
963	//	UMULL	1	0	1	0	H	0			Rn					Rd			0x2F00A000
964	//	UMULL2	1	0	1	0	H	0			Rn					Rd			0x2F00A000
965	//	FMULX	1	0	0	1	H	0			Rn					Rd			0x2F809000
966	//	**AdvSIMD modified immec**	**cmode**				**o2**	**1**	**d**	**e**	**f**	**g**	**h**			**Rd**			
967	//	MOVI	0	x	x	0	0	1	d	e	f	g	h			Rd			0x0F000400
968	//	ORR	0	x	x	1	0	1	d	e	f	g	h			Rd			0x0F001400
969	//	MOVI	1	0	x	0	0	1	d	e	f	g	h			Rd			0x0F008400
970	//	ORR	1	0	x	1	0	1	d	e	f	g	h			Rd			0x0F009400
971	//	MOVI	1	1	0	x	0	1	d	e	f	g	h			Rd			0x0F00C400
972	//	MOVI	1	1	1	0	0	1	d	e	f	g	h			Rd			0x0F00E400
973	//	FMOV	1	1	1	1	0	1	d	e	f	g	h			Rd			0x0F00F400
974	//	MVNI	0	x	x	0	0	1	d	e	f	g	h			Rd			0x2F000400
975	//	BIC	0	x	x	1	0	1	d	e	f	g	h			Rd			0x2F001400
976	//	MVNI	1	0	x	0	0	1	d	e	f	g	h			Rd			0x2F008400
977	//	BIC	1	0	x	1	0	1	d	e	f	g	h			Rd			0x2F009400
978	//	MVNI	1	1	0	x	0	1	d	e	f	g	h			Rd			0x2F00C400
979	//	MOVI	1	1	1	0	0	1	d	e	f	g	h			Rd			0x2F00E400
980	//	MOVI	1	1	1	0	0	1	d	e	f	g	h			Rd			0x6F00E400
981	//	FMOV	1	1	1	1	0	1	d	e	f	g	h			Rd			0x6F00F400
982	//	**AdvSIMD shift by immedi**	**opcode**					**1**			**Rn**					**Rd**			
983	//	SSHR	0	0	0	0	0	1			Rn					Rd			0x0F000400
984	//	SSRA	0	0	0	1	0	1			Rn					Rd			0x0F001400
985	//	SRSHR	0	0	1	0	0	1			Rn					Rd			0x0F002400
986	//	SRSRA	0	0	1	1	0	1			Rn					Rd			0x0F003400
987	//	SHL	0	1	0	1	0	1			Rn					Rd			0x0F005400
988	//	SQSHL	0	1	1	1	0	1			Rn					Rd			0x0F007400
989	//	SHRN	1	0	0	0	0	1			Rn					Rd			0x0F008400
990	//	SHRN2	1	0	0	0	0	1			Rn					Rd			0x0F008400
991	//	RSHRN	1	0	0	0	1	1			Rn					Rd			0x0F008C00
992	//	RSHRN2	1	0	0	0	1	1			Rn					Rd			0x0F008C00
993	//	SQSHRN	1	0	0	1	0	1			Rn					Rd			0x0F009400
994	//	SQSHRN2	1	0	0	1	0	1			Rn					Rd			0x0F009400

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
995	//	SQRSHRN	1	0	0	1	1	1			Rn					Rd			0x0F009C00
996	//	SQRSHRN2	1	0	0	1	1	1			Rn					Rd			0x0F009C00
997	//	SSHLL	1	0	1	0	0	1			Rn					Rd			0x0F00A400
998	//	SSHLL2	1	0	1	0	0	1			Rn					Rd			0x0F00A400
999	//	SCVTF	1	1	1	0	0	1			Rn					Rd			0x0F00E400
1000	//	FCVTZS	1	1	1	1	1	1			Rn					Rd			0x0F00FC00
1001	//	USHR	0	0	0	0	0	1			Rn					Rd			0x2F000400
1002	//	USRA	0	0	0	1	0	1			Rn					Rd			0x2F001400
1003	//	URSHR	0	0	1	0	0	1			Rn					Rd			0x2F002400
1004	//	URSRA	0	0	1	1	0	1			Rn					Rd			0x2F003400
1005	//	SRI	0	1	0	0	0	1			Rn					Rd			0x2F004400
1006	//	SLI	0	1	0	1	0	1			Rn					Rd			0x2F005400
1007	//	SQSHLU	0	1	1	0	0	1			Rn					Rd			0x2F006400
1008	//	UQSHL	0	1	1	1	0	1			Rn					Rd			0x2F007400
1009	//	SQSHRUN	1	0	0	0	0	1			Rn					Rd			0x2F008400
1010	//	SQSHRUN2	1	0	0	0	0	1			Rn					Rd			0x2F008400
1011	//	SQRSHRUN	1	0	0	0	1	1			Rn					Rd			0x2F008C00
1012	//	SQRSHRUN2	1	0	0	0	1	1			Rn					Rd			0x2F008C00
1013	//	UQSHRN	1	0	0	1	0	1			Rn					Rd			0x2F009400
1014	//	UQRSHRN	1	0	0	1	1	1			Rn					Rd			0x2F009C00
1015	//	UQRSHRN2	1	0	0	1	1	1			Rn					Rd			0x2F009C00
1016	//	USHLL	1	0	1	0	0	1			Rn					Rd			0x2F00A400
1017	//	USHLL2	1	0	1	0	0	1			Rn					Rd			0x2F00A400
1018	//	UCVTF	1	1	1	0	0	1			Rn					Rd			0x2F00E400
1019	//	FCVTZU	1	1	1	1	1	1			Rn					Rd			0x2F00FC00
1020	//	**AdvSIMD TBL/TBX**	**0**		**len**		**op**	**0**	**0**		**Rn**					**Rd**			
1021	//	TBL	0	0	0	0	0	0			Rn					Rd			0x0E000000
1022	//	TBX	0	0	0	1	0	0			Rn					Rd			0x0E001000
1023	//	TBL	0	0	1	0	0	0			Rn					Rd			0x0E002000
1024	//	TBX	0	0	1	1	0	0			Rn					Rd			0x0E003000
1025	//	TBL	0	1	0	0	0	0			Rn					Rd			0x0E004000
1026	//	TBX	0	1	0	1	0	0			Rn					Rd			0x0E005000
1027	//	TBL	0	1	1	0	0	0			Rn					Rd			0x0E006000
1028	//	TBX	0	1	1	1	0	0			Rn					Rd			0x0E007000

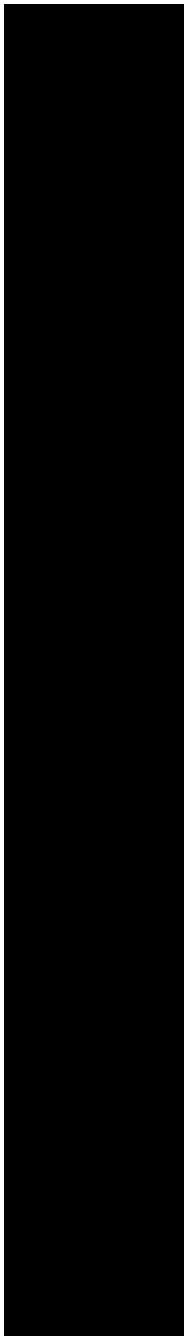
1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
1029	//	**AdvSIMD ZIP/UZP/TRN**	**0**	**opcode**			**1**	**0**			**Rn**					**Rd**			
1030	//	UZP1	0	0	0	1	1	0			Rn					Rd			0x0E001800
1031	//	TRN1	0	0	1	0	1	0			Rn					Rd			0x0E002800
1032	//	ZIP1	0	0	1	1	1	0			Rn					Rd			0x0E003800
1033	//	UZP2	0	1	0	1	1	0			Rn					Rd			0x0E005800
1034	//	TRN2	0	1	1	0	1	0			Rn					Rd			0x0E006800
1035	//	ZIP2	0	1	1	1	1	0			Rn					Rd			0x0E007800
1036	//	**AdvSIMD EXT**	**0**		**imm4**			**0**			**Rn**					**Rd**			
1037	//	EXT	0		imm4			0			Rn					Rd			0x2E000000
1038	//	**Loads and stores**																	
1039	//	**AdvSIMD load/store multi**		**opcode**			**size**				**Rn**					**Rt**			
1040	//	ST4	0	0	0	0	size				Rn					Rt			0x0C000000
1041	//	ST1	0	0	1	0	size				Rn					Rt			0x0C002000
1042	//	ST3	0	1	0	0	size				Rn					Rt			0x0C004000
1043	//	ST1	0	1	1	0	size				Rn					Rt			0x0C006000
1044	//	ST1	0	1	1	1	size				Rn					Rt			0x0C007000
1045	//	ST2	1	0	0	0	size				Rn					Rt			0x0C008000
1046	//	ST1	1	0	1	0	size				Rn					Rt			0x0C00A000
1047	//	LD4	0	0	0	0	size				Rn					Rt			0x0C400000
1048	//	LD1	0	0	1	0	size				Rn					Rt			0x0C402000
1049	//	LD3	0	1	0	0	size				Rn					Rt			0x0C404000
1050	//	LD1	0	1	1	0	size				Rn					Rt			0x0C406000
1051	//	LD1	0	1	1	1	size				Rn					Rt			0x0C407000
1052	//	LD2	1	0	0	0	size				Rn					Rt			0x0C408000
1053	//	LD1	1	0	1	0	size				Rn					Rt			0x0C40A000
1054	//	**AdvSIMD load/store multi**		**opcode**			**size**				**Rn**					**Rt**			
1055	//	ST4	0	0	0	0	size				Rn					Rt			0x0C800000
1056	//	ST1	0	0	1	0	size				Rn					Rt			0x0C802000
1057	//	ST3	0	1	0	0	size				Rn					Rt			0x0C804000
1058	//	ST1	0	1	1	0	size				Rn					Rt			0x0C806000
1059	//	ST1	0	1	1	1	size				Rn					Rt			0x0C807000
1060	//	ST2	1	0	0	0	size				Rn					Rt			0x0C808000
1061	//	ST1	1	0	1	0	size				Rn					Rt			0x0C80A000
1062	//	ST4	0	0	0	0	size				Rn					Rt			0x0C9F0000



1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
1063	//	ST1	0	0	1	0	size				Rn					Rt			0x0C9F2000
1064	//	ST3	0	1	0	0	size				Rn					Rt			0x0C9F4000
1065	//	ST1	0	1	1	0	size				Rn					Rt			0x0C9F6000
1066	//	ST1	0	1	1	1	size				Rn					Rt			0x0C9F7000
1067	//	ST2	1	0	0	0	size				Rn					Rt			0x0C9F8000
1068	//	ST1	1	0	1	0	size				Rn					Rt			0x0C9FA000
1069	//	LD4	0	0	0	0	size				Rn					Rt			0x0CC00000
1070	//	LD1	0	0	1	0	size				Rn					Rt			0x0CC02000
1071	//	LD3	0	1	0	0	size				Rn					Rt			0x0CC04000
1072	//	LD1	0	1	1	0	size				Rn					Rt			0x0CC06000
1073	//	LD1	0	1	1	1	size				Rn					Rt			0x0CC07000
1074	//	LD2	1	0	0	0	size				Rn					Rt			0x0CC08000
1075	//	LD1	1	0	1	0	size				Rn					Rt			0x0CC0A000
1076	//	LD4	0	0	0	0	size				Rn					Rt			0x0CDF0000
1077	//	LD1	0	0	1	0	size				Rn					Rt			0x0CDF2000
1078	//	LD3	0	1	0	0	size				Rn					Rt			0x0CDF4000
1079	//	LD1	0	1	1	0	size				Rn					Rt			0x0CDF6000
1080	//	LD1	0	1	1	1	size				Rn					Rt			0x0CDF7000
1081	//	LD2	1	0	0	0	size				Rn					Rt			0x0CDF8000
1082	//	LD1	1	0	1	0	size				Rn					Rt			0x0CDFA000
1083	//	**AdvSIMD load/store singl**	**opcode**			**S**	**size**				**Rn**					**Rt**			
1084	//	ST1	0	0	0	-	-	-			Rn					Rt			0x0D000000
1085	//	ST3	0	0	1	-	-	-			Rn					Rt			0x0D002000
1086	//	ST1	0	1	0	-	x	0			Rn					Rt			0x0D004000
1087	//	ST3	0	1	1	-	x	0			Rn					Rt			0x0D006000
1088	//	ST1	1	0	0	-	0	0			Rn					Rt			0x0D008000
1089	//	ST1	1	0	0	0	0	1			Rn					Rt			0x0D008400
1090	//	ST3	1	0	1	-	0	0			Rn					Rt			0x0D00A000
1091	//	ST3	1	0	1	0	0	1			Rn					Rt			0x0D00A400
1092	//	ST2	0	0	0	-	-	-			Rn					Rt			0x0D200000
1093	//	ST4	0	0	1	-	-	-			Rn					Rt			0x0D202000
1094	//	ST2	0	1	0	-	x	0			Rn					Rt			0x0D204000
1095	//	ST4	0	1	1	-	x	0			Rn					Rt			0x0D206000
1096	//	ST2	1	0	0	-	0	0			Rn					Rt			0x0D208000



1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
1097	//	ST2	1	0	0	0	0	1			Rn					Rt			0x0D208400
1098	//	ST4	1	0	1	-	0	0			Rn					Rt			0x0D20A000
1099	//	ST4	1	0	1	0	0	1			Rn					Rt			0x0D20A400
1100	//	LD1	0	0	0	-	-	-			Rn					Rt			0x0D400000
1101	//	LD3	0	0	1	-	-	-			Rn					Rt			0x0D402000
1102	//	LD1	0	1	0	-	x	0			Rn					Rt			0x0D404000
1103	//	LD3	0	1	1	-	x	0			Rn					Rt			0x0D406000
1104	//	LD1	1	0	0	-	0	0			Rn					Rt			0x0D408000
1105	//	LD1	1	0	0	0	0	1			Rn					Rt			0x0D408400
1106	//	LD3	1	0	1	-	0	0			Rn					Rt			0x0D40A000
1107	//	LD3	1	0	1	0	0	1			Rn					Rt			0x0D40A400
1108	//	LD1R	1	1	0	0	-	-			Rn					Rt			0x0D40C000
1109	//	LD3R	1	1	1	0	-	-			Rn					Rt			0x0D40E000
1110	//	LD2	0	0	0	-	-	-			Rn					Rt			0x0D600000
1111	//	LD4	0	0	1	-	-	-			Rn					Rt			0x0D602000
1112	//	LD2	0	1	0	-	x	0			Rn					Rt			0x0D604000
1113	//	LD4	0	1	1	-	x	0			Rn					Rt			0x0D606000
1114	//	LD2	1	0	0	-	0	0			Rn					Rt			0x0D608000
1115	//	LD2	1	0	0	0	0	1			Rn					Rt			0x0D608400
1116	//	LD4	1	0	1	-	0	0			Rn					Rt			0x0D60A000
1117	//	LD4	1	0	1	0	0	1			Rn					Rt			0x0D60A400
1118	//	LD2R	1	1	0	0	-	-			Rn					Rt			0x0D60C000
1119	//	LD4R	1	1	1	0	-	-			Rn					Rt			0x0D60E000
1120	//	**AdvSIMD load/store singl**	**opcode**			**S**		**size**			**Rn**					**Rt**			
1121	//	ST1	0	0	0	-	-	-			Rn					Rt			0x0D800000
1122	//	ST3	0	0	1	-	-	-			Rn					Rt			0x0D802000
1123	//	ST1	0	1	0	-	x	0			Rn					Rt			0x0D804000
1124	//	ST3	0	1	1	-	x	0			Rn					Rt			0x0D806000
1125	//	ST1	1	0	0	-	0	0			Rn					Rt			0x0D808000
1126	//	ST1	1	0	0	0	0	1			Rn					Rt			0x0D808400
1127	//	ST3	1	0	1	-	0	0			Rn					Rt			0x0D80A000
1128	//	ST3	1	0	1	0	0	1			Rn					Rt			0x0D80A400
1129	//	ST1	0	0	0	-	-	-			Rn					Rt			0x0D9F0000
1130	//	ST3	0	0	1	-	-	-			Rn					Rt			0x0D9F2000



1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
1131	//	ST1	0	1	0	-	x	0			Rn					Rt			0x0D9F4000
1132	//	ST3	0	1	1	-	x	0			Rn					Rt			0x0D9F6000
1133	//	ST1	1	0	0	-	0	0			Rn					Rt			0x0D9F8000
1134	//	ST1	1	0	0	0	0	1			Rn					Rt			0x0D9F8400
1135	//	ST3	1	0	1	-	0	0			Rn					Rt			0x0D9FA000
1136	//	ST3	1	0	1	0	0	1			Rn					Rt			0x0D9FA400
1137	//	ST2	0	0	0	-	-	-			Rn					Rt			0x0DA00000
1138	//	ST4	0	0	1	-	-	-			Rn					Rt			0x0DA02000
1139	//	ST2	0	1	0	-	x	0			Rn					Rt			0x0DA04000
1140	//	ST4	0	1	1	-	x	0			Rn					Rt			0x0DA06000
1141	//	ST2	1	0	0	-	0	0			Rn					Rt			0x0DA08000
1142	//	ST2	1	0	0	0	0	1			Rn					Rt			0x0DA08400
1143	//	ST4	1	0	1	-	0	0			Rn					Rt			0x0DA0A000
1144	//	ST4	1	0	1	0	0	1			Rn					Rt			0x0DA0A400
1145	//	ST2	0	0	0	-	-	-			Rn					Rt			0x0DBF0000
1146	//	ST4	0	0	1	-	-	-			Rn					Rt			0x0DBF2000
1147	//	ST2	0	1	0	-	x	0			Rn					Rt			0x0DBF4000
1148	//	ST4	0	1	1	-	x	0			Rn					Rt			0x0DBF6000
1149	//	ST2	1	0	0	-	0	0			Rn					Rt			0x0DBF8000
1150	//	ST2	1	0	0	0	0	1			Rn					Rt			0x0DBF8400
1151	//	ST4	1	0	1	-	0	0			Rn					Rt			0x0DBFA000
1152	//	ST4	1	0	1	0	0	1			Rn					Rt			0x0DBFA400
1153	//	LD1	0	0	0	-	-	-			Rn					Rt			0x0DC00000
1154	//	LD3	0	0	1	-	-	-			Rn					Rt			0x0DC02000
1155	//	LD1	0	1	0	-	x	0			Rn					Rt			0x0DC04000
1156	//	LD3	0	1	1	-	x	0			Rn					Rt			0x0DC06000
1157	//	LD1	1	0	0	-	0	0			Rn					Rt			0x0DC08000
1158	//	LD1	1	0	0	0	0	1			Rn					Rt			0x0DC08400
1159	//	LD3	1	0	1	-	0	0			Rn					Rt			0x0DC0A000
1160	//	LD3	1	0	1	0	0	1			Rn					Rt			0x0DC0A400
1161	//	LD1R	1	1	0	0	-	-			Rn					Rt			0x0DC0C000
1162	//	LD3R	1	1	1	0	-	-			Rn					Rt			0x0DC0E000
1163	//	LD1	0	0	0	-	-	-			Rn					Rt			0x0DDF0000
1164	//	LD3	0	0	1	-	-	-			Rn					Rt			0x0DDF2000

1	in_use	Opcode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Binary
1165	//	LD1	0	1	0	-	x	0			Rn					Rt			0x0DDF4000
1166	//	LD3	0	1	1	-	x	0			Rn					Rt			0x0DDF6000
1167	//	LD1	1	0	0	-	0	0			Rn					Rt			0x0DDF8000
1168	//	LD1	1	0	0	0	0	1			Rn					Rt			0x0DDF8400
1169	//	LD3	1	0	1	-	0	0			Rn					Rt			0x0DDFA000
1170	//	LD3	1	0	1	0	0	1			Rn					Rt			0x0DDFA400
1171	//	LD1R	1	1	0	0	-	-			Rn					Rt			0x0DDFC000
1172	//	LD3R	1	1	1	0	-	-			Rn					Rt			0x0DDFE000
1173	//	LD2	0	0	0	-	-	-			Rn					Rt			0x0DE00000
1174	//	LD4	0	0	1	-	-	-			Rn					Rt			0x0DE02000
1175	//	LD2	0	1	0	-	x	0			Rn					Rt			0x0DE04000
1176	//	LD4	0	1	1	-	x	0			Rn					Rt			0x0DE06000
1177	//	LD2	1	0	0	-	0	0			Rn					Rt			0x0DE08000
1178	//	LD2	1	0	0	0	0	1			Rn					Rt			0x0DE08400
1179	//	LD4	1	0	1	-	0	0			Rn					Rt			0x0DE0A000
1180	//	LD4	1	0	1	0	0	1			Rn					Rt			0x0DE0A400
1181	//	LD2R	1	1	0	0	-	-			Rn					Rt			0x0DE0C000
1182	//	LD4R	1	1	1	0	-	-			Rn					Rt			0x0DE0E000
1183	//	LD2	0	0	0	-	-	-			Rn					Rt			0x0DFF0000
1184	//	LD4	0	0	1	-	-	-			Rn					Rt			0x0DFF2000
1185	//	LD2	0	1	0	-	x	0			Rn					Rt			0x0DFF4000
1186	//	LD4	0	1	1	-	x	0			Rn					Rt			0x0DFF6000
1187	//	LD2	1	0	0	-	0	0			Rn					Rt			0x0DFF8000
1188	//	LD2	1	0	0	0	0	1			Rn					Rt			0x0DFF8400
1189	//	LD4	1	0	1	-	0	0			Rn					Rt			0x0DFFA000
1190	//	LD4	1	0	1	0	0	1			Rn					Rt			0x0DFFA400
1191	//	LD2R	1	1	0	0	-	-			Rn					Rt			0x0DFFC000
1192	//	LD4R	1	1	1	0	-	-			Rn					Rt			0x0DFFE000

	in_use	Opcode	//Opcode	BINARY	OI
2		**UNALLOCATED**	/* UNALLOCATED */		
3		BAD	ARM64Op_bad,	/* 0x00000000BAD	
4		**Branch,exception gener**	/* Branch,exception generation and system Instruction */		
5		**Compare _ Branch (imme**	/* Compare _ Branch (immediate) */		
6		CBZ	ARM64Op_cbz_32_bit,	/* 0x34000000CBZ	
7		CBNZ	ARM64Op_cbnz_32_bit,	/* 0x35000000CBN	
8		CBZ	ARM64Op_cbz_64_bit,	/* 0xB4000000CBZ	
9		CBNZ	ARM64Op_cbnz_64_bit,	/* 0xB5000000CBN	
10		**Test & branch (immediate**	/* Test & branch (immediate) */		
11		TBZ	ARM64Op_tbz,	/* 0x36000000TBZ	
12		TBNZ	ARM64Op_tbnz,	/* 0x37000000TBNZ	
13		**Conditional branch (imme**	/* Conditional branch (immediate) */		
14		B_cond	ARM64Op_b_cond,	/* 0x54000000B_cor	
15		**Exception generation**	/* Exception generation */		
16		SVC	ARM64Op_svc,	/* 0xD4000001SVC	
17		HVC	ARM64Op_hvc,	/* 0xD4000002HVC	
18		SMC	ARM64Op_smc,	/* 0xD4000003SMC	
19		BRK	ARM64Op_brk,	/* 0xD4200000BRK	
20		HLT	ARM64Op_hlt,	/* 0xD4400000HLT	
21		DCPS1	ARM64Op_dcps1,	/* 0xD4A00001DCPS	
22		DCPS2	ARM64Op_dcps2,	/* 0xD4A00002DCPS	
23		DCPS3	ARM64Op_dcps3,	/* 0xD4A00003DCPS	
24		**System**	/* System */		
25		MSR	ARM64Op_msr_immediate,	/* 0xD500401FM	
26		HINT	ARM64Op_hint,	/* 0xD503201FHINT	
27		CLREX	ARM64Op_clrex,	/* 0xD503305FCLREX	
28		DSB	ARM64Op_dsb,	/* 0xD503309FDSB	
29		DMB	ARM64Op_dmb,	/* 0xD50330BFDMB	
30		ISB	ARM64Op_isb,	/* 0xD50330DFISB	
31		SYS	ARM64Op_sys,	/* 0xD5080000SYS	
32		MSR	ARM64Op_msr_register,	/* 0xD5100000MSF	
33		SYSL	ARM64Op_sysl,	/* 0xD5280000SYSL	
34		MRS	ARM64Op_mrs,	/* 0xD5300000MRS	
35		**Unconditional branch (reg**	/* Unconditional branch (register) */		
36		BR	ARM64Op_br,	/* 0xD61F0000BR	
37		BLR	ARM64Op_blr,	/* 0xD63F0000BLR	
38		RET	ARM64Op_ret,	/* 0xD65F0000RET	
39		ERET	ARM64Op_eret,	/* 0xD69F03E0ERET	

	in_use	Opcode	//Opcode	BINARY	OI
40		DRPS	ARM64Op_drps,	/* 0xD6BF03E0	DRPS
41		**Unconditional branch (im**	/* Unconditional branch (immediate) */		
42		B	ARM64Op_b,	/* 0x14000000	B */
43		BL	ARM64Op_bl,	/* 0x94000000	BL *
44		**Loads and stores**	/* Loads and stores */		
45		**Load/store exclusive**	/* Load/store exclusive */		
46		STXRB	ARM64Op_stxrb,	/* 0x08000000	STXRB
47		STLXRB	ARM64Op_stlxrb,	/* 0x08008000	STLXRE
48		LDXRB	ARM64Op_ldxrb,	/* 0x08400000	LDXRB
49		LDAXRB	ARM64Op_ldaxrb,	/* 0x08408000	LDAXR
50		STLRB	ARM64Op_stlrb,	/* 0x08808000	STLRB
51		LDARB	ARM64Op_ldarb,	/* 0x08C08000	LDARB
52		STXRH	ARM64Op_stxrh,	/* 0x48000000	STXRH
53		STLXRH	ARM64Op_stlxrh,	/* 0x48008000	STLXRH
54		LDXRH	ARM64Op_ldxrh,	/* 0x48400000	LDXRH
55		LDAXRH	ARM64Op_ldaxrh,	/* 0x48408000	LDAXR
56		STLRH	ARM64Op_stlrh,	/* 0x48808000	STLRH
57		LDARH	ARM64Op_ldarh,	/* 0x48C08000	LDARH
58		STXR	ARM64Op_stxr_32_bit,	/* 0x88000000	STXR
59		STLXR	ARM64Op_stlxr_32_bit,	/* 0x88008000	STLX
60		STXP	ARM64Op_stxp_32_bit,	/* 0x88200000	STXF
61		STLXP	ARM64Op_stlxp_32_bit,	/* 0x88208000	STLX
62		LDXR	ARM64Op_ldxr_32_bit,	/* 0x88400000	LDXF
63		LDAXR	ARM64Op_ldaxr_32_bit,	/* 0x88408000	LDAX
64		LDXP	ARM64Op_ldxp_32_bit,	/* 0x88600000	LDXF
65		LDAXP	ARM64Op_ldaxp_32_bit,	/* 0x88608000	LDA
66		STLR	ARM64Op_stlr_32_bit,	/* 0x88808000	STLR
67		LDAR	ARM64Op_ldar_32_bit,	/* 0x88C08000	LDAF
68		STXR	ARM64Op_stxr_64_bit,	/* 0xC8000000	STXF
69		STLXR	ARM64Op_stlxr_64_bit,	/* 0xC8008000	STLX
70		STXP	ARM64Op_stxp_64_bit,	/* 0xC8200000	STXI
71		STLXP	ARM64Op_stlxp_64_bit,	/* 0xC8208000	STLX
72		LDXR	ARM64Op_ldxr_64_bit,	/* 0xC8400000	LDXF
73		LDAXR	ARM64Op_ldaxr_64_bit,	/* 0xC8408000	LDA
74		LDXP	ARM64Op_ldxp_64_bit,	/* 0xC8600000	LDXI
75		LDAXP	ARM64Op_ldaxp_64_bit,	/* 0xC8608000	LDA
76		STLR	ARM64Op_stlr_64_bit,	/* 0xC8808000	STLR
77		LDAR	ARM64Op_ldar_64_bit,	/* 0xC8C08000	LDAI

1	in_use	Opcode	//Opcode	BINARY	OI
78		**Load register (literal)**	/* Load register (literal) */		
79		LDR	ARM64Op_ldr_literal_32_bit,	/* 0x18000000LDR	
80		LDR	ARM64Op_ldr_literal_SIMD_FP_32_bit,	/* 0x1C0000(	
81		LDR	ARM64Op_ldr_literal_64_bit,	/* 0x58000000LDR	
82		LDR	ARM64Op_ldr_literal_SIMD_FP_64_bit,	/* 0x5C0000(	
83		LDRSW	ARM64Op_ldrsw_literal,	/* 0x98000000LDRS	
84		LDR	ARM64Op_ldr_literal_SIMD_FP_128_bit,	/* 0x9C0000	
85		PRFM	ARM64Op_prfm_literal,	/* 0xD8000000PRFM	
86		**Load/store no-allocate pa**	/* Load/store no-allocate pair (offset) */		
87		STNP	ARM64Op_stnp_32_bit,	/* 0x28000000STNI	
88		LDNP	ARM64Op_ldnp_32_bit,	/* 0x28400000LDNI	
89		STNP	ARM64Op_stnp_SIMD_FP_32_bit,	/* 0x2C00000	
90		LDNP	ARM64Op_ldnp_SIMD_FP_32_bit,	/* 0x2C40000	
91		STNP	ARM64Op_stnp_SIMD_FP_64_bit,	/* 0x6C00000	
92		LDNP	ARM64Op_ldnp_SIMD_FP_64_bit,	/* 0x6C40000	
93		STNP	ARM64Op_stnp_64_bit,	/* 0xA8000000STN	
94		LDNP	ARM64Op_ldnp_64_bit,	/* 0xA8400000LDN	
95		STNP	ARM64Op_stnp_SIMD_FP_128_bit,	/* 0xAC0000	
96		LDNP	ARM64Op_ldnp_SIMD_FP_128_bit,	/* 0xAC4000	
97		**Load/store register pair (p**	/* Load/store register pair (post-indexed) */		
98		STP	ARM64Op_stp_1_32_bit,	/* 0x28800000STP	
99		LDP	ARM64Op_ldp_1_32_bit,	/* 0x28C00000LDP	
100		STP	ARM64Op_stp_SIMD_FP_1_32_bit,	/* 0x2C8000(	
101		LDP	ARM64Op_ldp_SIMD_FP_1_32_bit,	/* 0x2CC000	
102		LDPSW	ARM64Op_ldpsw_Post_index,	/* 0x68C00000L	
103		STP	ARM64Op_stp_SIMD_FP_1_64_bit,	/* 0x6C8000(	
104		LDP	ARM64Op_ldp_SIMD_FP_1_64_bit,	/* 0x6CC000	
105		STP	ARM64Op_stp_1_64_bit,	/* 0xA8800000STP	
106		LDP	ARM64Op_ldp_1_64_bit,	/* 0xA8C00000LDP	
107		STP	ARM64Op_stp_SIMD_FP_1_128_bit,	/* 0xAC800(	
108		LDP	ARM64Op_ldp_SIMD_FP_1_128_bit,	/* 0xACC00(	
109		**Load/store register pair (o**	/* Load/store register pair (offset) */		
110		STP	ARM64Op_stp_2_32_bit,	/* 0x29000000STP	
111		LDP	ARM64Op_ldp_2_32_bit,	/* 0x29400000LDP	
112		STP	ARM64Op_stp_SIMD_FP_2_32_bit,	/* 0x2D0000(	

1	in_use	Opcode	//Opcode	BINARY OI
113		LDP	ARM64Op_ldp_SIMD_FP_2_32_bit,	/* 0x2D4000(
114		LDPSW	ARM64Op_ldpsw_Signed_offset,	/* 0x69400000L
115		STP	ARM64Op_stp_SIMD_FP_2_64_bit,	/* 0x6D0000(
116		LDP	ARM64Op_ldp_SIMD_FP_2_64_bit,	/* 0x6D4000(
117		STP	ARM64Op_stp_2_64_bit,	/* 0xA9000000STF
118		LDP	ARM64Op_ldp_2_64_bit,	/* 0xA9400000LDF
119		STP	ARM64Op_stp_SIMD_FP_2_128_bit,	/* 0xAD000(
120		LDP	ARM64Op_ldp_SIMD_FP_2_128_bit,	/* 0xAD400(
121		**Load/store register pair (p**	/* Load/store register pair (pre-indexed) */	
122		STP	ARM64Op_stp_3_32_bit,	/* 0x29800000STP
123		LDP	ARM64Op_ldp_3_32_bit,	/* 0x29C00000LDF
124		STP	ARM64Op_stp_SIMD_FP_3_32_bit,	/* 0x2D8000(
125		LDP	ARM64Op_ldp_SIMD_FP_3_32_bit,	/* 0x2DC000
126		LDPSW	ARM64Op_ldpsw_Pre_index,	/* 0x69C00000L
127		STP	ARM64Op_stp_SIMD_FP_3_64_bit,	/* 0x6D8000(
128		LDP	ARM64Op_ldp_SIMD_FP_3_64_bit,	/* 0x6DC000
129		STP	ARM64Op_stp_3_64_bit,	/* 0xA9800000STF
130		LDP	ARM64Op_ldp_3_64_bit,	/* 0xA9C00000LDF
131		STP	ARM64Op_stp_SIMD_FP_3_128_bit,	/* 0xAD800(
132		LDP	ARM64Op_ldp_SIMD_FP_3_128_bit,	/* 0xADC00(
133		**Load/store register (unsc**	/* Load/store register (unscaled immediate) */	
134		STURB	ARM64Op_sturb,	/* 0x38000000STURB
135		LDURB	ARM64Op_ldurb,	/* 0x38400000LDURB
136		LDURSB	ARM64Op_ldursb_64_bit,	/* 0x38800000LDU
137		LDURSB	ARM64Op_ldursb_32_bit,	/* 0x38C00000LDU
138		STUR	ARM64Op_stur_SIMD_FP_8_bit,	/* 0x3C00000(
139		LDUR	ARM64Op_ldur_SIMD_FP_8_bit,	/* 0x3C40000(
140		STUR	ARM64Op_stur_SIMD_FP_128_bit,	/* 0x3C8000(
141		LDUR	ARM64Op_ldur_SIMD_FP_128_bit,	/* 0x3CC000(
142		STURH	ARM64Op_sturh,	/* 0x78000000STURH
143		LDURH	ARM64Op_ldurh,	/* 0x78400000LDURH
144		LDURSH	ARM64Op_ldursh_64_bit,	/* 0x78800000LDU
145		LDURSH	ARM64Op_ldursh_32_bit,	/* 0x78C00000LDU
146		STUR	ARM64Op_stur_SIMD_FP_16_bit,	/* 0x7C0000(
147		LDUR	ARM64Op_ldur_SIMD_FP_16_bit,	/* 0x7C4000(
148		STUR	ARM64Op_stur_32_bit,	/* 0xB8000000STUF
149		LDUR	ARM64Op_ldur_32_bit,	/* 0xB8400000LDUF
150		LDURSW	ARM64Op_ldursw,	/* 0xB8800000LDURS

1	in_use	Opcode	//Opcode	BINARY OI
151		STUR	ARM64Op_stur_SIMD_FP_32_bit,	/* 0xBC00000
152		LDUR	ARM64Op_ldur_SIMD_FP_32_bit,	/* 0xBC40000
153		STUR	ARM64Op_stur_64_bit,	/* 0xF8000000STUF
154		LDUR	ARM64Op_ldur_64_bit,	/* 0xF8400000LDUF
155		PRFUM	ARM64Op_prfum,	/* 0xF8800000PRFUM
156		STUR	ARM64Op_stur_SIMD_FP_64_bit,	/* 0xFC00000
157		LDUR	ARM64Op_ldur_SIMD_FP_64_bit,	/* 0xFC40000
158		**Load/store register (imme**	/* Load/store register (immediate post-indexed) */	
159		STRB	ARM64Op_strb_immediate_Post_index,	/* 0x380004
160		LDRB	ARM64Op_ldrb_immediate_Post_index,	/* 0x384004
161		LDRSB	ARM64Op_ldrsb_immediate_1_64_bit,	/* 0x388004(
162		LDRSB	ARM64Op_ldrsb_immediate_1_32_bit,	/* 0x38C004(
163		STR	ARM64Op_str_immediate_SIMD_FP_1_8_bit,	/* 0x3C0
164		LDR	ARM64Op_ldr_immediate_SIMD_FP_1_8_bit,	/* 0x3C4
165		STR	ARM64Op_str_immediate_SIMD_FP_1_128_bit,	/* 0x3C
166		LDR	ARM64Op_ldr_immediate_SIMD_FP_1_128_bit,	/* 0x3C
167		STRH	ARM64Op_strh_immediate_Post_index,	/* 0x780004
168		LDRH	ARM64Op_ldrh_immediate_Post_index,	/* 0x784004
169		LDRSH	ARM64Op_ldrsh_immediate_1_64_bit,	/* 0x788004(
170		LDRSH	ARM64Op_ldrsh_immediate_1_32_bit,	/* 0x78C004(
171		STR	ARM64Op_str_immediate_SIMD_FP_1_16_bit,	/* 0x7C(
172		LDR	ARM64Op_ldr_immediate_SIMD_FP_1_16_bit,	/* 0x7C4
173		STR	ARM64Op_str_immediate_1_32_bit,	/* 0xB800040(
174		LDR	ARM64Op_ldr_immediate_1_32_bit,	/* 0xB840040(
175		LDRSW	ARM64Op_ldrsw_immediate_Post_index,	/* 0xB8800
176		STR	ARM64Op_str_immediate_SIMD_FP_1_32_bit,	/* 0xBC(
177		LDR	ARM64Op_ldr_immediate_SIMD_FP_1_32_bit,	/* 0xBC4
178		STR	ARM64Op_str_immediate_1_64_bit,	/* 0xF800040(
179		LDR	ARM64Op_ldr_immediate_1_64_bit,	/* 0xF840040(
180		STR	ARM64Op_str_immediate_SIMD_FP_1_64_bit,	/* 0xFC(
181		LDR	ARM64Op_ldr_immediate_SIMD_FP_1_64_bit,	/* 0xFC4
182		**Load/store register (unpri**	/* Load/store register (unprivileged) */	
183		STTRB	ARM64Op_sttrb,	/* 0x38000800STTRB
184		LDTRB	ARM64Op_ldtrb,	/* 0x38400800LDTRB
185		LDTRSB	ARM64Op_ldtrsb_64_bit,	/* 0x38800800LDTF
186		LDTRSB	ARM64Op_ldtrsb_32_bit,	/* 0x38C00800LDTI
187		STTRH	ARM64Op_sttrh,	/* 0x78000800STTRH
188		LDTRH	ARM64Op_ldtrh,	/* 0x78400800LDTRH

1	in_use	Opcode	//Opcode	BINARY OI
189		LDTRSH	ARM64Op_ldtrsh_64_bit,	/* 0x78800800LDTF
190		LDTRSH	ARM64Op_ldtrsh_32_bit,	/* 0x78C00800LDTI
191		STTR	ARM64Op_sttr_32_bit,	/* 0xB8000800STTR
192		LDTR	ARM64Op_ldtr_32_bit,	/* 0xB8400800LDTR
193		LDTRSW	ARM64Op_ldtrsw,	/* 0xB8800800LDTRS
194		STTR	ARM64Op_sttr_64_bit,	/* 0xF8000800STTR
195		LDTR	ARM64Op_ldtr_64_bit,	/* 0xF8400800LDTR
196		**Load/store register (imme**	/* Load/store register (immediate pre-indexed) */	
197		STRB	ARM64Op_strb_immediate_Pre_index,	/* 0x38000C
198		LDRB	ARM64Op_ldrb_immediate_Pre_index,	/* 0x38400C
199		LDRSB	ARM64Op_ldrsb_immediate_2_64_bit,	/* 0x38800C(
200		LDRSB	ARM64Op_ldrsb_immediate_2_32_bit,	/* 0x38C00C
201		STR	ARM64Op_str_immediate_SIMD_FP_2_8_bit,	/* 0x3C0
202		LDR	ARM64Op_ldr_immediate_SIMD_FP_2_8_bit,	/* 0x3C4
203		STR	ARM64Op_str_immediate_SIMD_FP_2_128_bit,	/* 0x3C
204		LDR	ARM64Op_ldr_immediate_SIMD_FP_2_128_bit,	/* 0x3C
205		STRH	ARM64Op_strh_immediate_Pre_index,	/* 0x78000C
206		LDRH	ARM64Op_ldrh_immediate_Pre_index,	/* 0x78400C
207		LDRSH	ARM64Op_ldrsh_immediate_2_64_bit,	/* 0x78800C(
208		LDRSH	ARM64Op_ldrsh_immediate_2_32_bit,	/* 0x78C00C
209		STR	ARM64Op_str_immediate_SIMD_FP_2_16_bit,	/* 0x7C(
210		LDR	ARM64Op_ldr_immediate_SIMD_FP_2_16_bit,	/* 0x7C4
211		STR	ARM64Op_str_immediate_2_32_bit,	/* 0xB8000C0
212		LDR	ARM64Op_ldr_immediate_2_32_bit,	/* 0xB8400C0
213		LDRSW	ARM64Op_ldrsw_immediate_Pre_index,	/* 0xB8800(
214		STR	ARM64Op_str_immediate_SIMD_FP_2_32_bit,	/* 0xBC(
215		LDR	ARM64Op_ldr_immediate_SIMD_FP_2_32_bit,	/* 0xBC4
216		STR	ARM64Op_str_immediate_2_64_bit,	/* 0xF8000C0
217		LDR	ARM64Op_ldr_immediate_2_64_bit,	/* 0xF8400C0
218		STR	ARM64Op_str_immediate_SIMD_FP_2_64_bit,	/* 0xFC(
219		LDR	ARM64Op_ldr_immediate_SIMD_FP_2_64_bit,	/* 0xFC4
220		**Load/store register (regis**	/* Load/store register (register offset) */	
221		STRB	ARM64Op_strb_register,	/* 0x38200800STRE
222		LDRB	ARM64Op_ldrb_register,	/* 0x38600800LDRE
223		LDRSB	ARM64Op_ldrsb_register_64_bit,	/* 0x38A00800LI
224		LDRSB	ARM64Op_ldrsb_register_32_bit,	/* 0x38E00800LI
225		STR	ARM64Op_str_register_SIMD_FP_8_bit,	/* 0x3C2008
226		LDR	ARM64Op_ldr_register_SIMD_FP_8_bit,	/* 0x3C6008

1	in_use	Opcode	//Opcode	BINARY OI
227		STR	ARM64Op_str_register_SIMD_FP_128_bit,	/* 0x3CA0(
228		LDR	ARM64Op_ldr_register_SIMD_FP_128_bit,	/* 0x3CE0(
229		STRH	ARM64Op_strh_register,	/* 0x78200800STRH
230		LDRH	ARM64Op_ldrh_register,	/* 0x78600800LDRH
231		LDRSH	ARM64Op_ldrsh_register_64_bit,	/* 0x78A00800LI
232		LDRSH	ARM64Op_ldrsh_register_32_bit,	/* 0x78E00800LI
233		STR	ARM64Op_str_register_SIMD_FP_16_bit,	/* 0x7C200
234		LDR	ARM64Op_ldr_register_SIMD_FP_16_bit,	/* 0x7C600
235		STR	ARM64Op_str_register_32_bit,	/* 0xB8200800ST
236		LDR	ARM64Op_ldr_register_32_bit,	/* 0xB8600800LD
237		LDRSW	ARM64Op_ldrsw_register,	/* 0xB8A00800LDR
238		STR	ARM64Op_str_register_SIMD_FP_32_bit,	/* 0xBC200
239		LDR	ARM64Op_ldr_register_SIMD_FP_32_bit,	/* 0xBC600
240		STR	ARM64Op_str_register_64_bit,	/* 0xF8200800ST
241		LDR	ARM64Op_ldr_register_64_bit,	/* 0xF8600800LD
242		PRFM	ARM64Op_prfm_register,	/* 0xF8A00800PRF
243		STR	ARM64Op_str_register_SIMD_FP_64_bit,	/* 0xFC200
244		LDR	ARM64Op_ldr_register_SIMD_FP_64_bit,	/* 0xFC600
245		**Load/store register (unsig**	/* Load/store register (unsigned immediate) */	
246		STRB	ARM64Op_strb_immediate_Unsigned_offset,	/* 0x3900(
247		LDRB	ARM64Op_ldrb_immediate_Unsigned_offset,	/* 0x3940(
248		LDRSB	ARM64Op_ldrsb_immediate_3_64_bit,	/* 0x398000(
249		LDRSB	ARM64Op_ldrsb_immediate_3_32_bit,	/* 0x39C000(
250		STR	ARM64Op_str_immediate_SIMD_FP_8_bit,	/* 0x3D00
251		LDR	ARM64Op_ldr_immediate_SIMD_FP_8_bit,	/* 0x3D40
252		STR	ARM64Op_str_immediate_SIMD_FP_128_bit,	/* 0x3D8
253		LDR	ARM64Op_ldr_immediate_SIMD_FP_128_bit,	/* 0x3DC
254		STRH	ARM64Op_strh_immediate_Unsigned_offset,	/* 0x7900(
255		LDRH	ARM64Op_ldrh_immediate_Unsigned_offset,	/* 0x7940(
256		LDRSH	ARM64Op_ldrsh_immediate_3_64_bit,	/* 0x798000(
257		LDRSH	ARM64Op_ldrsh_immediate_3_32_bit,	/* 0x79C000(
258		STR	ARM64Op_str_immediate_SIMD_FP_16_bit,	/* 0x7D0(
259		LDR	ARM64Op_ldr_immediate_SIMD_FP_16_bit,	/* 0x7D4(
260		STR	ARM64Op_str_immediate_3_32_bit,	/* 0xB900000(
261		LDR	ARM64Op_ldr_immediate_3_32_bit,	/* 0xB940000(
262		LDRSW	ARM64Op_ldrsw_immediate_Unsigned_offset,	/* 0xB98(
263		STR	ARM64Op_str_immediate_SIMD_FP_32_bit,	/* 0xBD0(
264		LDR	ARM64Op_ldr_immediate_SIMD_FP_32_bit,	/* 0xBD4(

1	in_use	Opcode	//Opcode	BINARY OI
265		STR	ARM64Op_str_immediate_3_64_bit,	/* 0xF900000(
266		LDR	ARM64Op_ldr_immediate_3_64_bit,	/* 0xF940000(
267		PRFM	ARM64Op_prfm_immediate,	/* 0xF9800000PI
268		STR	ARM64Op_str_immediate_SIMD_FP_64_bit,	/* 0xFD0(
269		LDR	ARM64Op_ldr_immediate_SIMD_FP_64_bit,	/* 0xFD4(
270		**Data processing – Imme**	/* Data processing – Immediate */	
271		**PC-rel. addressing**	/* PC-rel. addressing */	
272		ADR	ARM64Op_adr,	/* 0x10000000ADR
273		ADRP	ARM64Op_adrp,	/* 0x90000000ADRP
274		**Add/subtract (immediate)**	/* Add/subtract (immediate) */	
275		ADD	ARM64Op_add_immediate_32_bit,	/* 0x1100000(
276		ADDS	ARM64Op_adds_immediate_32_bit,	/* 0x3100000
277		SUB	ARM64Op_sub_immediate_32_bit,	/* 0x5100000(
278		SUBS	ARM64Op_subs_immediate_32_bit,	/* 0x7100000
279		ADD	ARM64Op_add_immediate_64_bit,	/* 0x9100000(
280		ADDS	ARM64Op_adds_immediate_64_bit,	/* 0xB100000
281		SUB	ARM64Op_sub_immediate_64_bit,	/* 0xD100000(
282		SUBS	ARM64Op_subs_immediate_64_bit,	/* 0xF100000
283		**Logical (immediate)**	/* Logical (immediate) */	
284		AND	ARM64Op_and_immediate_32_bit,	/* 0x1200000(
285		ORR	ARM64Op_orr_immediate_32_bit,	/* 0x32000000
286		EOR	ARM64Op_eor_immediate_32_bit,	/* 0x52000000
287		ANDS	ARM64Op_ands_immediate_32_bit,	/* 0x7200000
288		AND	ARM64Op_and_immediate_64_bit,	/* 0x9200000(
289		ORR	ARM64Op_orr_immediate_64_bit,	/* 0xB2000000
290		EOR	ARM64Op_eor_immediate_64_bit,	/* 0xD200000(
291		ANDS	ARM64Op_ands_immediate_64_bit,	/* 0xF200000
292		**Move wide (immediate)**	/* Move wide (immediate) */	
293		MOVN	ARM64Op_movn_32_bit,	/* 0x12800000MO
294		MOVZ	ARM64Op_movz_32_bit,	/* 0x52800000MO
295		MOVK	ARM64Op_movk_32_bit,	/* 0x72800000MO
296		MOVN	ARM64Op_movn_64_bit,	/* 0x92800000MO
297		MOVZ	ARM64Op_movz_64_bit,	/* 0xD2800000MO
298		MOVK	ARM64Op_movk_64_bit,	/* 0xF2800000MO
299		**Bitfield**	/* Bitfield */	
300		SBFM	ARM64Op_sbfm_32_bit,	/* 0x13000000SBF
301		BFM	ARM64Op_bfm_32_bit,	/* 0x33000000BFM
302		UBFM	ARM64Op_ubfm_32_bit,	/* 0x53000000UBF

1	in_use	Opcode	//Opcode	BINARY OI
303		SBFM	ARM64Op_sbfm_64_bit,	/* 0x93400000SBF
304		BFM	ARM64Op_bfm_64_bit,	/* 0xB3400000BFM
305		UBFM	ARM64Op_ubfm_64_bit,	/* 0xD3400000UBF
306		**Extract**	/* Extract */	
307		EXTR	ARM64Op_extr_32_bit,	/* 0x13800000EXTF
308		EXTR	ARM64Op_extr_64_bit,	/* 0x93C00000EXTF
309		**Data Processing – regis**	/* Data Processing – register */	
310		**Logical (shifted register)**	/* Logical (shifted register) */	
311		AND	ARM64Op_and_shifted_register_32_bit,	/* 0x0A00000
312		BIC	ARM64Op_bic_shifted_register_32_bit,	/* 0x0A20000(
313		ORR	ARM64Op_orr_shifted_register_32_bit,	/* 0x2A00000(
314		ORN	ARM64Op_orn_shifted_register_32_bit,	/* 0x2A20000(
315		EOR	ARM64Op_eor_shifted_register_32_bit,	/* 0x4A00000(
316		EON	ARM64Op_eon_shifted_register_32_bit,	/* 0x4A20000
317		ANDS	ARM64Op_ands_shifted_register_32_bit,	/* 0x6A00000(
318		BICS	ARM64Op_bics_shifted_register_32_bit,	/* 0x6A20000(
319		AND	ARM64Op_and_shifted_register_64_bit,	/* 0x8A00000
320		BIC	ARM64Op_bic_shifted_register_64_bit,	/* 0x8A20000(
321		ORR	ARM64Op_orr_shifted_register_64_bit,	/* 0xAA000000
322		ORN	ARM64Op_orn_shifted_register_64_bit,	/* 0xAA20000
323		EOR	ARM64Op_eor_shifted_register_64_bit,	/* 0xCA00000
324		EON	ARM64Op_eon_shifted_register_64_bit,	/* 0xCA2000(
325		ANDS	ARM64Op_ands_shifted_register_64_bit,	/* 0xEA0000(
326		BICS	ARM64Op_bics_shifted_register_64_bit,	/* 0xEA20000
327		**Add/subtract (shifted regi**	/* Add/subtract (shifted register) */	
328		ADD	ARM64Op_add_shifted_register_32_bit,	/* 0x0B00000
329		ADDS	ARM64Op_adds_shifted_register_32_bit,	/* 0x2B00000(
330		SUB	ARM64Op_sub_shifted_register_32_bit,	/* 0x4B00000
331		SUBS	ARM64Op_subs_shifted_register_32_bit,	/* 0x6B00000(
332		ADD	ARM64Op_add_shifted_register_64_bit,	/* 0x8B00000
333		ADDS	ARM64Op_adds_shifted_register_64_bit,	/* 0xAB0000(
334		SUB	ARM64Op_sub_shifted_register_64_bit,	/* 0xCB00000(
335		SUBS	ARM64Op_subs_shifted_register_64_bit,	/* 0xEB00000(
336		**Add/subtract (extended re**	/* Add/subtract (extended register) */	
337		ADD	ARM64Op_add_extended_register_32_bit,	/* 0x0B200(
338		ADDS	ARM64Op_adds_extended_register_32_bit,	/* 0x2B200
339		SUB	ARM64Op_sub_extended_register_32_bit,	/* 0x4B200(
340		SUBS	ARM64Op_subs_extended_register_32_bit,	/* 0x6B200

1	in_use	Opcode	//Opcode	BINARY Ol
341		ADD	ARM64Op_add_extended_register_64_bit,	/* 0x8B200
342		ADDS	ARM64Op_adds_extended_register_64_bit,	/* 0xAB20(
343		SUB	ARM64Op_sub_extended_register_64_bit,	/* 0xCB200
344		SUBS	ARM64Op_subs_extended_register_64_bit,	/* 0xEB20(
345		**Add/subtract (with carry)**	/* Add/subtract (with carry) */	
346		ADC	ARM64Op_adc_32_bit,	/* 0x1A000000ADC
347		ADCS	ARM64Op_adcs_32_bit,	/* 0x3A000000ADC
348		SBC	ARM64Op_sbc_32_bit,	/* 0x5A000000SBC
349		SBCS	ARM64Op_sbcs_32_bit,	/* 0x7A000000SBC
350		ADC	ARM64Op_adc_64_bit,	/* 0x9A000000ADC
351		ADCS	ARM64Op_adcs_64_bit,	/* 0xBA000000AD(
352		SBC	ARM64Op_sbc_64_bit,	/* 0xDA000000SBC
353		SBCS	ARM64Op_sbcs_64_bit,	/* 0xFA000000SBC
354		**Conditional compare (reg**	/* Conditional compare (register) */	
355		CCMN	ARM64Op_ccmn_register_32_bit,	/* 0x3A400000(
356		CCMN	ARM64Op_ccmn_register_64_bit,	/* 0xBA400000
357		CCMP	ARM64Op_ccmp_register_32_bit,	/* 0x7A400000(
358		CCMP	ARM64Op_ccmp_register_64_bit,	/* 0xFA400000
359		**Conditional compare (imr**	/* Conditional compare (immediate) */	
360		CCMN	ARM64Op_ccmn_immediate_32_bit,	/* 0x3A4008(
361		CCMN	ARM64Op_ccmn_immediate_64_bit,	/* 0xBA4008(
362		CCMP	ARM64Op_ccmp_immediate_32_bit,	/* 0x7A4008(
363		CCMP	ARM64Op_ccmp_immediate_64_bit,	/* 0xFA4008(
364		**Conditional select**	/* Conditional select */	
365		CSEL	ARM64Op_csel_32_bit,	/* 0x1A800000CSE
366		CSINC	ARM64Op_csinc_32_bit,	/* 0x1A800400CSI
367		CSINV	ARM64Op_csinv_32_bit,	/* 0x5A800000CSI
368		CSNEG	ARM64Op_csneg_32_bit,	/* 0x5A800400CS
369		CSEL	ARM64Op_csel_64_bit,	/* 0x9A800000CSE
370		CSINC	ARM64Op_csinc_64_bit,	/* 0x9A800400CSI
371		CSINV	ARM64Op_csinv_64_bit,	/* 0xDA800000CS
372		CSNEG	ARM64Op_csneg_64_bit,	/* 0xDA800400CS
373		**Data-processing (3 sourc**	/* Data-processing (3 source) */	
374		MADD	ARM64Op_madd_32_bit,	/* 0x1B000000MA
375		MADD	ARM64Op_madd_64_bit,	/* 0x9B000000MA
376		SMADDL	ARM64Op_smaddl,	/* 0x9B200000SMAD
377		UMADDL	ARM64Op_umaddl,	/* 0x9BA00000UMAD
378		MSUB	ARM64Op_msub_32_bit,	/* 0x1B008000MS

1	in_use	Opcode	//Opcode	BINARY OI
379		MSUB	ARM64Op_msub_64_bit,	/* 0x9B008000MS
380		SMSUBL	ARM64Op_smsubl,	/* 0x9B208000SMSU
381		UMSUBL	ARM64Op_umsubl,	/* 0x9BA08000UMSU
382		SMULH	ARM64Op_smulh,	/* 0x9B400000SMULI
383		UMULH	ARM64Op_umulh,	/* 0x9BC00000UMUL
384		**Data-processing (2 sourc**	/* Data-processing (2 source) */	
385		CRC32X	ARM64Op_crc32x,	/* 0x9AC04C00CRC3
386		CRC32CX	ARM64Op_crc32cx,	/* 0x9AC05C00CRC3
387		CRC32B	ARM64Op_crc32b,	/* 0x1AC04000CRC3
388		CRC32CB	ARM64Op_crc32cb,	/* 0x1AC05000CRC3
389		CRC32H	ARM64Op_crc32h,	/* 0x1AC04400CRC3
390		CRC32CH	ARM64Op_crc32ch,	/* 0x1AC05400CRC3
391		CRC32W	ARM64Op_crc32w,	/* 0x1AC04800CRC3
392		CRC32CW	ARM64Op_crc32cw,	/* 0x1AC05800CRC3
393		UDIV	ARM64Op_udiv_32_bit,	/* 0x1AC00800UDIV
394		UDIV	ARM64Op_udiv_64_bit,	/* 0x9AC00800UDIV
395		SDIV	ARM64Op_sdiv_32_bit,	/* 0x1AC00C00SDIV
396		SDIV	ARM64Op_sdiv_64_bit,	/* 0x9AC00C00SDIV
397		LSLV	ARM64Op_lslv_32_bit,	/* 0x1AC02000LSLV
398		LSLV	ARM64Op_lslv_64_bit,	/* 0x9AC02000LSLV
399		LSRV	ARM64Op_lsrv_32_bit,	/* 0x1AC02400LSRV
400		LSRV	ARM64Op_lsrv_64_bit,	/* 0x9AC02400LSRV
401		ASRV	ARM64Op_asrv_32_bit,	/* 0x1AC02800ASR
402		ASRV	ARM64Op_asrv_64_bit,	/* 0x9AC02800ASR
403		RORV	ARM64Op_rorv_32_bit,	/* 0x1AC02C00ROR
404		RORV	ARM64Op_rorv_64_bit,	/* 0x9AC02C00ROR
405		**Data-processing (1 sourc**	/* Data-processing (1 source) */	
406		RBIT	ARM64Op_rbit_32_bit,	/* 0x5AC00000RBIT
407		RBIT	ARM64Op_rbit_64_bit,	/* 0xDAC00000RBIT
408		CLZ	ARM64Op_clz_32_bit,	/* 0x5AC01000CLZ
409		CLZ	ARM64Op_clz_64_bit,	/* 0xDAC01000CLZ
410		CLS	ARM64Op_cls_32_bit,	/* 0x5AC01400CLS
411		CLS	ARM64Op_cls_64_bit,	/* 0xDAC01400CLS
412		REV	ARM64Op_rev_32_bit,	/* 0x5AC00800REV
413		REV	ARM64Op_rev_64_bit,	/* 0xDAC00C00REV
414		REV16	ARM64Op_rev16_64_bit,	/* 0xDAC00400RE
415		REV16	ARM64Op_rev16_32_bit,	/* 0x5AC00400REV
416		REV32	ARM64Op_rev32,	/* 0xDAC00800REV3

	in_use	Opcode	//Opcode	BINARY	OI
1	in_use	Opcode	//Opcode	BINARY	OI
417	//	**Data Processing – SIMD**	/* Data Processing – SIMD and floating point */		
418	//	**Floating-point<->fixed-po**	/* Floating-point<->fixed-point conversions */		
419	//	SCVTF	//ARM64Op_scvtf_scalar_fixed_point_32_bit_to_single_precision,	/*	
420	//	UCVTF	//ARM64Op_ucvtf_scalar_fixed_point_32_bit_to_single_precision,	/*	
421	//	FCVTZS	//ARM64Op_fcvtzs_scalar_fixed_point_Single_precision_to_32_bit,	/*	
422	//	FCVTZU	//ARM64Op_fcvtzu_scalar_fixed_point_Single_precision_to_32_bit,	/*	
423	//	SCVTF	//ARM64Op_scvtf_scalar_fixed_point_32_bit_to_double_precision,	/*	
424	//	UCVTF	//ARM64Op_ucvtf_scalar_fixed_point_32_bit_to_double_precision,	/*	
425	//	FCVTZS	//ARM64Op_fcvtzs_scalar_fixed_point_Double_precision_to_32_bit,	/	
426	//	FCVTZU	//ARM64Op_fcvtzu_scalar_fixed_point_Double_precision_to_32_bit,	/	
427	//	SCVTF	//ARM64Op_scvtf_scalar_fixed_point_64_bit_to_single_precision,	/*	
428	//	UCVTF	//ARM64Op_ucvtf_scalar_fixed_point_64_bit_to_single_precision,	/*	
429	//	FCVTZS	//ARM64Op_fcvtzs_scalar_fixed_point_Single_precision_to_64_bit,	/*	
430	//	FCVTZU	//ARM64Op_fcvtzu_scalar_fixed_point_Single_precision_to_64_bit,	/*	
431	//	SCVTF	//ARM64Op_scvtf_scalar_fixed_point_64_bit_to_double_precision,	/*	
432	//	UCVTF	//ARM64Op_ucvtf_scalar_fixed_point_64_bit_to_double_precision,	/*	
433	//	FCVTZS	//ARM64Op_fcvtzs_scalar_fixed_point_Double_precision_to_64_bit,	/	
434	//	FCVTZU	//ARM64Op_fcvtzu_scalar_fixed_point_Double_precision_to_64_bit,	/	
435	//	**Floating-point conditional**	/* Floating-point conditional compare */		
436	//	FCCMP	//ARM64Op_fccmp_Single_precision,	/* 0x1E2004	
437	//	FCCMPE	//ARM64Op_fccmpe_Single_precision,	/* 0x1E2004	
438	//	FCCMP	//ARM64Op_fccmp_Double_precision,	/* 0x1E6004	
439	//	FCCMPE	//ARM64Op_fccmpe_Double_precision,	/* 0x1E600	
440	//	**Floating-point data-proce**	/* Floating-point data-processing (2 source) */		
441	//	FMUL	//ARM64Op_fmul_scalar_Single_precision,	/* 0x1E200	
442	//	FDIV	//ARM64Op_fdiv_scalar_Single_precision,	/* 0x1E201	
443	//	FADD	//ARM64Op_fadd_scalar_Single_precision,	/* 0x1E202	
444	//	FSUB	//ARM64Op_fsub_scalar_Single_precision,	/* 0x1E203	
445	//	FMAX	//ARM64Op_fmax_scalar_Single_precision,	/* 0x1E20	
446	//	FMIN	//ARM64Op_fmin_scalar_Single_precision,	/* 0x1E205	
447	//	FMAXNM	//ARM64Op_fmaxnm_scalar_Single_precision,	/* 0x1E2	
448	//	FMINNM	//ARM64Op_fminnm_scalar_Single_precision,	/* 0x1E2	
449	//	FNMUL	//ARM64Op_fnmul_Single_precision,	/* 0x1E20880	
450	//	FMUL	//ARM64Op_fmul_scalar_Double_precision,	/* 0x1E60	

1	in_use	Opcode	//Opcode	BINARY OI
451	//	FDIV	//ARM64Op_fdiv_scalar_Double_precision,	/* 0x1E601
452	//	FADD	//ARM64Op_fadd_scalar_Double_precision,	/* 0x1E60
453	//	FSUB	//ARM64Op_fsub_scalar_Double_precision,	/* 0x1E60
454	//	FMAX	//ARM64Op_fmax_scalar_Double_precision,	/* 0x1E60
455	//	FMIN	//ARM64Op_fmin_scalar_Double_precision,	/* 0x1E60
456	//	FMAXNM	//ARM64Op_fmaxnm_scalar_Double_precision,	/* 0x1E
457	//	FMINNM	//ARM64Op_fminnm_scalar_Double_precision,	/* 0x1E6
458	//	FNMUL	//ARM64Op_fnmul_Double_precision,	/* 0x1E6088
459	//	**Floating-point conditional**	/* Floating-point conditional select */	
460	//	FCSEL	//ARM64Op_fcsel_Single_precision,	/* 0x1E200C0
461	//	FCSEL	//ARM64Op_fcsel_Double_precision,	/* 0x1E600C0
462	//	**Floating-point immediate**	/* Floating-point immediate */	
463	//	FMOV	//ARM64Op_fmov_scalar_immediate_Single_precision,	/* 0x1
464	//	FMOV	//ARM64Op_fmov_scalar_immediate_Double_precision,	/* 0x
465	//	**Floating-point compare**	/* Floating-point compare */	
466	//	FCMP	//ARM64Op_fcmp_Single_precision,	/* 0x1E20200
467	//	FCMP	//ARM64Op_fcmp_Single_precision_zero,	/* 0x1E202
468	//	FCMPE	//ARM64Op_fcmpe_Single_precision,	/* 0x1E2020
469	//	FCMPE	//ARM64Op_fcmpe_Single_precision_zero,	/* 0x1E202
470	//	FCMP	//ARM64Op_fcmp_Double_precision,	/* 0x1E6020
471	//	FCMP	//ARM64Op_fcmp_Double_precision_zero,	/* 0x1E60
472	//	FCMPE	//ARM64Op_fcmpe_Double_precision,	/* 0x1E602C
473	//	FCMPE	//ARM64Op_fcmpe_Double_precision_zero,	/* 0x1E60
474	//	**Floating-point data-proce**	/* Floating-point data-processing (1 source) */	
475	//	FMOV	//ARM64Op_fmov_register_Single_precision,	/* 0x1E20
476	//	FABS	//ARM64Op_fabs_scalar_Single_precision,	/* 0x1E20C
477	//	FNEG	//ARM64Op_fneg_scalar_Single_precision,	/* 0x1E214
478	//	FSQRT	//ARM64Op_fsqrt_scalar_Single_precision,	/* 0x1E21C0
479	//	FCVT	//ARM64Op_fcvt_Single_precision_to_double_precision,	/* 0x1
480	//	FCVT	//ARM64Op_fcvt_Single_precision_to_half_precision,	/* 0x1E2
481	//	FRINTN	//ARM64Op_frintn_scalar_Single_precision,	/* 0x1E2440
482	//	FRINTP	//ARM64Op_frintp_scalar_Single_precision,	/* 0x1E24C0
483	//	FRINTM	//ARM64Op_frintm_scalar_Single_precision,	/* 0x1E254
484	//	FRINTZ	//ARM64Op_frintz_scalar_Single_precision,	/* 0x1E25C0

1	in_use	Opcode	//Opcode	BINARY	OI
485	//	FRINTA	//ARM64Op_frinta_scalar_Single_precision,	/* 0x1E264(	
486	//	FRINTX	//ARM64Op_frintx_scalar_Single_precision,	/* 0x1E274(	
487	//	FRINTI	//ARM64Op_frinti_scalar_Single_precision,	/* 0x1E27C0	
488	//	FMOV	//ARM64Op_fmov_register_Double_precision,	/* 0x1E6(	
489	//	FABS	//ARM64Op_fabs_scalar_Double_precision,	/* 0x1E60(	
490	//	FNEG	//ARM64Op_fneg_scalar_Double_precision,	/* 0x1E61-	
491	//	FSQRT	//ARM64Op_fsqrt_scalar_Double_precision,	/* 0x1E61C	
492	//	FCVT	//ARM64Op_fcvt_Double_precision_to_single_precision,	/* 0x1	
493	//	FCVT	//ARM64Op_fcvt_Double_precision_to_half_precision,	/* 0x1E	
494	//	FRINTN	//ARM64Op_frintn_scalar_Double_precision,	/* 0x1E644	
495	//	FRINTP	//ARM64Op_frintp_scalar_Double_precision,	/* 0x1E64C	
496	//	FRINTM	//ARM64Op_frintm_scalar_Double_precision,	/* 0x1E65-	
497	//	FRINTZ	//ARM64Op_frintz_scalar_Double_precision,	/* 0x1E65C	
498	//	FRINTA	//ARM64Op_frinta_scalar_Double_precision,	/* 0x1E664	
499	//	FRINTX	//ARM64Op_frintx_scalar_Double_precision,	/* 0x1E674	
500	//	FRINTI	//ARM64Op_frinti_scalar_Double_precision,	/* 0x1E67C	
501	//	FCVT	//ARM64Op_fcvt_Half_precision_to_single_precision,	/* 0x1EE	
502	//	FCVT	//ARM64Op_fcvt_Half_precision_to_double_precision,	/* 0x1E	
503	//	**Floating-point<->integer c**	/* Floating-point<->integer conversions */		
504	//	FCVTNS	//ARM64Op_fcvtns_scalar_Single_precision_to_32_bit,	/* 0x1E	
505	//	FCVTNU	//ARM64Op_fcvtnu_scalar_Single_precision_to_32_bit,	/* 0x1E	
506	//	SCVTF	//ARM64Op_scvtf_scalar_integer_32_bit_to_single_precision,	/* 0x	
507	//	UCVTF	//ARM64Op_ucvtf_scalar_integer_32_bit_to_single_precision,	/* 0x	
508	//	FCVTAS	//ARM64Op_fcvtas_scalar_Single_precision_to_32_bit,	/* 0x1E	
509	//	FCVTAU	//ARM64Op_fcvtau_scalar_Single_precision_to_32_bit,	/* 0x1E	
510	//	FMOV	//ARM64Op_fmov_general_Single_precision_to_32_bit,	/* 0x1	
511	//	FMOV	//ARM64Op_fmov_general_32_bit_to_single_precision,	/* 0x1	
512	//	FCVTPS	//ARM64Op_fcvtps_scalar_Single_precision_to_32_bit,	/* 0x1E	
513	//	FCVTPU	//ARM64Op_fcvtpu_scalar_Single_precision_to_32_bit,	/* 0x1E	
514	//	FCVTMS	//ARM64Op_fcvtms_scalar_Single_precision_to_32_bit,	/* 0x1	
515	//	FCVTMU	//ARM64Op_fcvtmu_scalar_Single_precision_to_32_bit,	/* 0x1	
516	//	FCVTZS	//ARM64Op_fcvtzs_scalar_integer_Single_precision_to_32_bit,	/* 0	
517	//	FCVTZU	//ARM64Op_fcvtzu_scalar_integer_Single_precision_to_32_bit,	/* 0	
518	//	FCVTNS	//ARM64Op_fcvtns_scalar_Double_precision_to_32_bit,	/* 0x1	

1	in_use	Opcode	//Opcode	BINARY	OI
519	//	FCVTNU	//ARM64Op_fcvtnu_scalar_Double_precision_to_32_bit,	/* 0x1	
520	//	SCVTF	//ARM64Op_scvtf_scalar_integer_32_bit_to_double_precision,	/* 0	
521	//	UCVTF	//ARM64Op_ucvtf_scalar_integer_32_bit_to_double_precision,	/* 0	
522	//	FCVTAS	//ARM64Op_fcvtas_scalar_Double_precision_to_32_bit,	/* 0x1	
523	//	FCVTAU	//ARM64Op_fcvtau_scalar_Double_precision_to_32_bit,	/* 0x1	
524	//	FCVTPS	//ARM64Op_fcvtps_scalar_Double_precision_to_32_bit,	/* 0x1	
525	//	FCVTPU	//ARM64Op_fcvtpu_scalar_Double_precision_to_32_bit,	/* 0x1	
526	//	FCVTMS	//ARM64Op_fcvtms_scalar_Double_precision_to_32_bit,	/* 0x	
527	//	FCVTMU	//ARM64Op_fcvtmu_scalar_Double_precision_to_32_bit,	/* 0x	
528	//	FCVTZS	//ARM64Op_fcvtzs_scalar_integer_Double_precision_to_32_bit,	/* (	
529	//	FCVTZU	//ARM64Op_fcvtzu_scalar_integer_Double_precision_to_32_bit,	/* (	
530	//	FCVTNS	//ARM64Op_fcvtns_scalar_Single_precision_to_64_bit,	/* 0x9E	
531	//	FCVTNU	//ARM64Op_fcvtnu_scalar_Single_precision_to_64_bit,	/* 0x9E	
532	//	SCVTF	//ARM64Op_scvtf_scalar_integer_64_bit_to_single_precision,	/* 0x	
533	//	UCVTF	//ARM64Op_ucvtf_scalar_integer_64_bit_to_single_precision,	/* 0x	
534	//	FCVTAS	//ARM64Op_fcvtas_scalar_Single_precision_to_64_bit,	/* 0x9E	
535	//	FCVTAU	//ARM64Op_fcvtau_scalar_Single_precision_to_64_bit,	/* 0x9E	
536	//	FCVTPS	//ARM64Op_fcvtps_scalar_Single_precision_to_64_bit,	/* 0x9E	
537	//	FCVTPU	//ARM64Op_fcvtpu_scalar_Single_precision_to_64_bit,	/* 0x9E	
538	//	FCVTMS	//ARM64Op_fcvtms_scalar_Single_precision_to_64_bit,	/* 0x9	
539	//	FCVTMU	//ARM64Op_fcvtmu_scalar_Single_precision_to_64_bit,	/* 0x9	
540	//	FCVTZS	//ARM64Op_fcvtzs_scalar_integer_Single_precision_to_64_bit,	/* 0	
541	//	FCVTZU	//ARM64Op_fcvtzu_scalar_integer_Single_precision_to_64_bit,	/* 0	
542	//	FCVTNS	//ARM64Op_fcvtns_scalar_Double_precision_to_64_bit,	/* 0x9	
543	//	FCVTNU	//ARM64Op_fcvtnu_scalar_Double_precision_to_64_bit,	/* 0x9	
544	//	SCVTF	//ARM64Op_scvtf_scalar_integer_64_bit_to_double_precision,	/* 0	
545	//	UCVTF	//ARM64Op_ucvtf_scalar_integer_64_bit_to_double_precision,	/* 0	
546	//	FCVTAS	//ARM64Op_fcvtas_scalar_Double_precision_to_64_bit,	/* 0x9	
547	//	FCVTAU	//ARM64Op_fcvtau_scalar_Double_precision_to_64_bit,	/* 0x9	
548	//	FMOV	//ARM64Op_fmov_general_Double_precision_to_64_bit,	/* 0x	
549	//	FMOV	//ARM64Op_fmov_general_64_bit_to_double_precision,	/* 0x	
550	//	FCVTPS	//ARM64Op_fcvtps_scalar_Double_precision_to_64_bit,	/* 0x9	
551	//	FCVTPU	//ARM64Op_fcvtpu_scalar_Double_precision_to_64_bit,	/* 0x9	
552	//	FCVTMS	//ARM64Op_fcvtms_scalar_Double_precision_to_64_bit,	/* 0x	

1	in_use	Opcode	//Opcode	BINARY	Ol
553	//	FCVTMU	//ARM64Op_fcvtmu_scalar_Double_precision_to_64_bit,	/* 0x9	
554	//	FCVTZS	//ARM64Op_fcvtzs_scalar_integer_Double_precision_to_64_bit,	/* (	
555	//	FCVTZU	//ARM64Op_fcvtzu_scalar_integer_Double_precision_to_64_bit,	/* (	
556	//	FMOV	//ARM64Op_fmov_general_Top_half_of_128_bit_to_64_bit,	/* 0	
557	//	FMOV	//ARM64Op_fmov_general_64_bit_to_top_half_of_128_bit,	/* 0x	
558	//	**Floating-point data-proce**	/* Floating-point data-processing (3 source) */		
559	//	FMADD	//ARM64Op_fmadd_Single_precision,	/* 0x1F0000(	
560	//	FMSUB	//ARM64Op_fmsub_Single_precision,	/* 0x1F0080(	
561	//	FNMADD	//ARM64Op_fnmadd_Single_precision,	/* 0x1F2000	
562	//	FNMSUB	//ARM64Op_fnmsub_Single_precision,	/* 0x1F2080	
563	//	FMADD	//ARM64Op_fmadd_Double_precision,	/* 0x1F400(	
564	//	FMSUB	//ARM64Op_fmsub_Double_precision,	/* 0x1F4080	
565	//	FNMADD	//ARM64Op_fnmadd_Double_precision,	/* 0x1F600(	
566	//	FNMSUB	//ARM64Op_fnmsub_Double_precision,	/* 0x1F608(	
567	//	**AdvSIMD scalar three san**	/* AdvSIMD scalar three same */		
568	//	SQADD	//ARM64Op_sqadd_Scalar,	/* 0x5E200C00S	
569	//	SQSUB	//ARM64Op_sqsub_Scalar,	/* 0x5E202C00S(	
570	//	CMGT	//ARM64Op_cmgt_register_Scalar,	/* 0x5E20340(	
571	//	CMGE	//ARM64Op_cmge_register_Scalar,	/* 0x5E203C(	
572	//	SSHL	//ARM64Op_sshl_Scalar,	/* 0x5E204400SSI	
573	//	SQSHL	//ARM64Op_sqshl_register_Scalar,	/* 0x5E204C0(	
574	//	SRSHL	//ARM64Op_srshl_Scalar,	/* 0x5E205400SR	
575	//	SQRSHL	//ARM64Op_sqrshl_Scalar,	/* 0x5E205C00S(	
576	//	ADD	//ARM64Op_add_vector_Scalar,	/* 0x5E208400	
577	//	CMTST	//ARM64Op_cmtst_Scalar,	/* 0x5E208C00CI	
578	//	SQDMULH	//ARM64Op_sqdmulh_vector_Scalar,	/* 0x5E20B4	
579	//	FMULX	//ARM64Op_fmulx_Scalar,	/* 0x5E20DC00FI	
580	//	FCMEQ	//ARM64Op_fcmeq_register_Scalar,	/* 0x5E20E40	
581	//	FRECPS	//ARM64Op_frecps_Scalar,	/* 0x5E20FC00FF	
582	//	FRSQRTS	//ARM64Op_frsqrts_Scalar,	/* 0x5EA0FC00FF	
583	//	UQADD	//ARM64Op_uqadd_Scalar,	/* 0x7E200C00U	
584	//	UQSUB	//ARM64Op_uqsub_Scalar,	/* 0x7E202C00U	
585	//	CMHI	//ARM64Op_cmhi_register_Scalar,	/* 0x7E20340(	
586	//	CMHS	//ARM64Op_cmhs_register_Scalar,	/* 0x7E203C0	

1	in_use	Opcode	//Opcode	BINARY	OI
587	//	USHL	//ARM64Op_ushl_Scalar,	/* 0x7E204400US	
588	//	UQSHL	//ARM64Op_uqshl_register_Scalar,	/* 0x7E204C0(	
589	//	URSHL	//ARM64Op_urshl_Scalar,	/* 0x7E205400UR	
590	//	UQRSHL	//ARM64Op_uqrshl_Scalar,	/* 0x7E205C00U(	
591	//	SUB	//ARM64Op_sub_vector_Scalar,	/* 0x7E208400	
592	//	CMEQ	//ARM64Op_cmeq_register_Scalar,	/* 0x7E208C0	
593	//	SQRDMULH	//ARM64Op_sqrdmulh_vector_Scalar,	/* 0x7E20B4	
594	//	FCMGE	//ARM64Op_fcmge_register_Scalar,	/* 0x7E20E40	
595	//	FACGE	//ARM64Op_facge_Scalar,	/* 0x7E20EC00F/	
596	//	FABD	//ARM64Op_fabd_Scalar,	/* 0x7EA0D400FA	
597	//	FCMGT	//ARM64Op_fcmgt_register_Scalar,	/* 0x7EA0E40(	
598	//	FACGT	//ARM64Op_facgt_Scalar,	/* 0x7EA0EC00FA	
599	//	**AdvSIMD scalar three diff**	/* AdvSIMD scalar three different */		
600	//	SQDMLAL	//ARM64Op_sqdmlal_vector_Scalar,	/* 0x5E20900	
601	//	SQDMLAL2	//ARM64Op_sqdmlal2_vector_Scalar,	/* 0x5E2090(	
602	//	SQDMLSL	//ARM64Op_sqdmlsl_vector_Scalar,	/* 0x5E20B0(	
603	//	SQDMLSL2	//ARM64Op_sqdmlsl2_vector_Scalar,	/* 0x5E20B0(	
604	//	SQDMULL	//ARM64Op_sqdmull_vector_Scalar,	/* 0x5E20D0(	
605	//	SQDMULL2	//ARM64Op_sqdmull2_vector_Scalar,	/* 0x5E20D0	
606	//	**AdvSIMD scalar two-reg r**	/* AdvSIMD scalar two-reg misc */		
607	//	SUQADD	//ARM64Op_suqadd_Scalar,	/* 0x5E203800S	
608	//	SQABS	//ARM64Op_sqabs_Scalar,	/* 0x5E207800S(	
609	//	CMGT	//ARM64Op_cmgt_zero_Scalar,	/* 0x5E208800	
610	//	CMEQ	//ARM64Op_cmeq_zero_Scalar,	/* 0x5E20980(	
611	//	CMLT	//ARM64Op_cmlt_zero_Scalar,	/* 0x5E20A800(	
612	//	ABS	//ARM64Op_abs_Scalar,	/* 0x5E20B800AB	
613	//	SQXTN	//ARM64Op_sqxtn_Scalar,	/* 0x5E214800SC	
614	//	SQXTN2	//ARM64Op_sqxtn2_Scalar,	/* 0x5E214800S(	
615	//	FCVTNS	//ARM64Op_fcvtns_vector_Scalar,	/* 0x5E21A80(	
616	//	FCVTMS	//ARM64Op_fcvtms_vector_Scalar,	/* 0x5E21B80	
617	//	FCVTAS	//ARM64Op_fcvtas_vector_Scalar,	/* 0x5E21C80(	
618	//	SCVTF	//ARM64Op_scvtf_vector_integer_Scalar,	/* 0x5E21D8	
619	//	FCMGT	//ARM64Op_fcmgt_zero_Scalar,	/* 0x5EA0C80(	
620	//	FCMEQ	//ARM64Op_fcmeq_zero_Scalar,	/* 0x5EA0D80	

1	in_use	Opcode	//Opcode	BINARY OI
621	//	FCMLT	//ARM64Op_fcmlt_zero_Scalar,	/* 0x5EA0E800
622	//	FCVTPS	//ARM64Op_fcvtps_vector_Scalar,	/* 0x5EA1A80(
623	//	FCVTZS	//ARM64Op_fcvtzs_vector_integer_Scalar,	/* 0x5EA1B
624	//	FRECPE	//ARM64Op_frecpe_Scalar,	/* 0x5EA1D800FI
625	//	FRECPX	//ARM64Op_frecpx,	/* 0x5EA1F800FREC
626	//	USQADD	//ARM64Op_usqadd_Scalar,	/* 0x7E203800U
627	//	SQNEG	//ARM64Op_sqneg_Scalar,	/* 0x7E207800S(
628	//	CMGE	//ARM64Op_cmge_zero_Scalar,	/* 0x7E20880(
629	//	CMLE	//ARM64Op_cmle_zero_Scalar,	/* 0x7E209800
630	//	NEG	//ARM64Op_neg_vector_Scalar,	/* 0x7E20B800
631	//	SQXTUN	//ARM64Op_sqxtun_Scalar,	/* 0x7E212800S(
632	//	SQXTUN2	//ARM64Op_sqxtun2_Scalar,	/* 0x7E212800S
633	//	UQXTN	//ARM64Op_uqxtn_Scalar,	/* 0x7E214800UC
634	//	UQXTN2	//ARM64Op_uqxtn2_Scalar,	/* 0x7E214800U(
635	//	FCVTXN	//ARM64Op_fcvtxn_Scalar,	/* 0x7E216800FC
636	//	FCVTXN2	//ARM64Op_fcvtxn2_Scalar,	/* 0x7E216800F(
637	//	FCVTNU	//ARM64Op_fcvtnu_vector_Scalar,	/* 0x7E21A80(
638	//	FCVTMU	//ARM64Op_fcvtmu_vector_Scalar,	/* 0x7E21B80
639	//	FCVTAU	//ARM64Op_fcvtau_vector_Scalar,	/* 0x7E21C80(
640	//	UCVTF	//ARM64Op_ucvtf_vector_integer_Scalar,	/* 0x7E21D8
641	//	FCMGE	//ARM64Op_fcmge_zero_Scalar,	/* 0x7EA0C80
642	//	FCMLE	//ARM64Op_fcmle_zero_Scalar,	/* 0x7EA0D80(
643	//	FCVTPU	//ARM64Op_fcvtpu_vector_Scalar,	/* 0x7EA1A80(
644	//	FCVTZU	//ARM64Op_fcvtzu_vector_integer_Scalar,	/* 0x7EA1B
645	//	FRSQRTE	//ARM64Op_frsqrte_Scalar,	/* 0x7EA1D800FF
646	//	**AdvSIMD scalar pairwise**	/* AdvSIMD scalar pairwise */	
647	//	ADDP	//ARM64Op_addp_scalar,	/* 0x5E31B800AI
648	//	FMAXNMP	//ARM64Op_fmaxnmp_scalar,	/* 0x7E30C800
649	//	FADDP	//ARM64Op_faddp_scalar,	/* 0x7E30D800FA
650	//	FMAXP	//ARM64Op_fmaxp_scalar,	/* 0x7E30F800FM
651	//	FMINNMP	//ARM64Op_fminnmp_scalar,	/* 0x7EB0C800I
652	//	FMINP	//ARM64Op_fminp_scalar,	/* 0x7EB0F800FM
653	//	**AdvSIMD scalar copy**	/* AdvSIMD scalar copy */	
654	//	DUP	//ARM64Op_dup_element_Scalar,	/* 0x5E00040

1	in_use	Opcode	//Opcode	BINARY	Ol
655	//	**AdvSIMD scalar x indexe**	/* AdvSIMD scalar x indexed element */		
656	//	SQDMLAL	//ARM64Op_sqdmlal_by_element_Scalar,	/* 0x5F00	
657	//	SQDMLAL2	//ARM64Op_sqdmlal2_by_element_Scalar,	/* 0x5F00	
658	//	SQDMLSL	//ARM64Op_sqdmlsl_by_element_Scalar,	/* 0x5F007	
659	//	SQDMLSL2	//ARM64Op_sqdmlsl2_by_element_Scalar,	/* 0x5F00	
660	//	SQDMULL	//ARM64Op_sqdmull_by_element_Scalar,	/* 0x5F00E	
661	//	SQDMULL2	//ARM64Op_sqdmull2_by_element_Scalar,	/* 0x5F00	
662	//	SQDMULH	//ARM64Op_sqdmulh_by_element_Scalar,	/* 0x5F00	
663	//	SQRDMULH	//ARM64Op_sqrdmulh_by_element_Scalar,	/* 0x5F00	
664	//	FMLA	//ARM64Op_fmla_by_element_Scalar,	/* 0x5F8010	
665	//	FMLS	//ARM64Op_fmls_by_element_Scalar,	/* 0x5F8050	
666	//	FMUL	//ARM64Op_fmul_by_element_Scalar,	/* 0x5F8090	
667	//	FMULX	//ARM64Op_fmulx_by_element_Scalar,	/* 0x7F809(	
668	//	**AdvSIMD scalar shift by i**	/* AdvSIMD scalar shift by immediate */		
669	//	SSHR	//ARM64Op_sshr_Scalar,	/* 0x5F000400SSI	
670	//	SSRA	//ARM64Op_ssra_Scalar,	/* 0x5F001400SSI	
671	//	SRSHR	//ARM64Op_srshr_Scalar,	/* 0x5F002400SR	
672	//	SRSRA	//ARM64Op_srsra_Scalar,	/* 0x5F003400SR	
673	//	SHL	//ARM64Op_shl_Scalar,	/* 0x5F005400SHL	
674	//	SQSHL	//ARM64Op_sqshl_immediate_Scalar,	/* 0x5F0074	
675	//	SQSHRN	//ARM64Op_sqshrn_Scalar,	/* 0x5F009400S(	
676	//	SQSHRN2	//ARM64Op_sqshrn2_Scalar,	/* 0x5F009400S	
677	//	SQRSHRN	//ARM64Op_sqrshrn_Scalar,	/* 0x5F009C00S	
678	//	SQRSHRN2	//ARM64Op_sqrshrn2_Scalar,	/* 0x5F009C00S	
679	//	SCVTF	//ARM64Op_scvtf_vector_fixed_point_Scalar,	/* 0x5F00E	
680	//	FCVTZS	//ARM64Op_fcvtzs_vector_fixed_point_Scalar,	/* 0x5F00I	
681	//	USHR	//ARM64Op_ushr_Scalar,	/* 0x7F000400US	
682	//	USRA	//ARM64Op_usra_Scalar,	/* 0x7F001400US	
683	//	URSHR	//ARM64Op_urshr_Scalar,	/* 0x7F002400UR	
684	//	URSRA	//ARM64Op_ursra_Scalar,	/* 0x7F003400UR	
685	//	SRI	//ARM64Op_sri_Scalar,	/* 0x7F004400SRI	
686	//	SLI	//ARM64Op_sli_Scalar,	/* 0x7F005400SLI	
687	//	SQSHLU	//ARM64Op_sqshlu_Scalar,	/* 0x7F006400S(	
688	//	UQSHL	//ARM64Op_uqshl_immediate_Scalar,	/* 0x7F0074	

1	in_use	Opcode	//Opcode	BINARY OI
689	//	SQSHRUN	//ARM64Op_sqshrun_Scalar,	/* 0x7F008400S
690	//	SQSHRUN2	//ARM64Op_sqshrun2_Scalar,	/* 0x7F008400S
691	//	SQRSHRUN	//ARM64Op_sqrshrun_Scalar,	/* 0x7F008C00S
692	//	SQRSHRUN2	//ARM64Op_sqrshrun2_Scalar,	/* 0x7F008C00S
693	//	UQSHRN	//ARM64Op_uqshrn_Scalar,	/* 0x7F009400U(
694	//	UQRSHRN	//ARM64Op_uqrshrn_Scalar,	/* 0x7F009C00U
695	//	UQRSHRN2	//ARM64Op_uqrshrn2_Scalar,	/* 0x7F009C00U
696	//	UCVTF	//ARM64Op_ucvtf_vector_fixed_point_Scalar,	/* 0x7F00E
697	//	FCVTZU	//ARM64Op_fcvtzu_vector_fixed_point_Scalar,	/* 0x7F00I
698	//	**Crypto three-reg SHA**	/* Crypto three-reg SHA */	
699	//	SHA1C	//ARM64Op_sha1c,	/* 0x5E000000SHA1
700	//	SHA1P	//ARM64Op_sha1p,	/* 0x5E001000SHA1
701	//	SHA1M	//ARM64Op_sha1m,	/* 0x5E002000SHA
702	//	SHA1SU0	//ARM64Op_sha1su0,	/* 0x5E003000SHA
703	//	SHA256H	//ARM64Op_sha256h,	/* 0x5E004000SHA
704	//	SHA256H2	//ARM64Op_sha256h2,	/* 0x5E005000SH.
705	//	SHA256SU1	//ARM64Op_sha256su1,	/* 0x5E006000SH
706	//	**Crypto two-reg SHA**	/* Crypto two-reg SHA */	
707	//	SHA1H	//ARM64Op_sha1h,	/* 0x5E280800SHA1
708	//	SHA1SU1	//ARM64Op_sha1su1,	/* 0x5E281800SHA
709	//	SHA256SU0	//ARM64Op_sha256su0,	/* 0x5E282800SH
710	//	**Crypto AES**	/* Crypto AES */	
711	//	AESE	//ARM64Op_aese,	/* 0x4E284800AESE
712	//	AESD	//ARM64Op_aesd,	/* 0x4E285800AESD
713	//	AESMC	//ARM64Op_aesmc,	/* 0x4E286800AES
714	//	AESIMC	//ARM64Op_aesimc,	/* 0x4E287800AESI
715	//	**AdvSIMD three same**	/* AdvSIMD three same */	
716	//	SHADD	//ARM64Op_shadd,	/* 0x0E200400SHAI
717	//	SQADD	//ARM64Op_sqadd_Vector,	/* 0x0E200C00S
718	//	SRHADD	//ARM64Op_srhadd,	/* 0x0E201400SRH/
719	//	SHSUB	//ARM64Op_shsub,	/* 0x0E202400SHSU
720	//	SQSUB	//ARM64Op_sqsub_Vector,	/* 0x0E202C00S
721	//	CMGT	//ARM64Op_cmgt_register_Vector,	/* 0x0E20340(
722	//	CMGE	//ARM64Op_cmge_register_Vector,	/* 0x0E203C(

1	in_use	Opcode	//Opcode	BINARY OI
723	//	SSHL Vector	//ARM64Op_sshl vector,	/* 0x0E204400SSH
724	//	SQSHL	//ARM64Op_sqshl_register_Vector,	/* 0x0E204C0(
725	//	SRSHL	//ARM64Op_srshl_Vector,	/* 0x0E205400SR
726	//	SQRSHL	//ARM64Op_sqrshl_Vector,	/* 0x0E205C00S(
727	//	SMAX	//ARM64Op_smax,	/* 0x0E206400SMA)
728	//	SMIN	//ARM64Op_smin,	/* 0x0E206C00SMIN
729	//	SABD	//ARM64Op_sabd,	/* 0x0E207400SABD
730	//	SABA	//ARM64Op_saba,	/* 0x0E207C00SABA
731	//	ADD	//ARM64Op_add_vector_Vector,	/* 0x0E208400
732	//	CMTST	//ARM64Op_cmtst_Vector,	/* 0x0E208C00CI
733	//	MLA	//ARM64Op_mla_vector,	/* 0x0E209400ML/
734	//	MUL	//ARM64Op_mul_vector,	/* 0x0E209C00MU
735	//	SMAXP	//ARM64Op_smaxp,	/* 0x0E20A400SMA
736	//	SMINP	//ARM64Op_sminp,	/* 0x0E20AC00SMI
737	//	SQDMULH	//ARM64Op_sqdmulh_vector_Vector,	/* 0x0E20B4
738	//	ADDP	//ARM64Op_addp_vector,	/* 0x0E20BC00AI
739	//	FMAXNM	//ARM64Op_fmaxnm_vector,	/* 0x0E20C400I
740	//	FMLA	//ARM64Op_fmla_vector,	/* 0x0E20CC00FM
741	//	FADD	//ARM64Op_fadd_vector,	/* 0x0E20D400FA
742	//	FMULX	//ARM64Op_fmulx_Vector,	/* 0x0E20DC00FI
743	//	FCMEQ	//ARM64Op_fcmeq_register_Vector,	/* 0x0E20E4(
744	//	FMAX	//ARM64Op_fmax_vector,	/* 0x0E20F400FM
745	//	FRECPS	//ARM64Op_frecps_Vector,	/* 0x0E20FC00FI
746	//	AND	//ARM64Op_and_vector,	/* 0x0E201C00AN
747	//	BIC	//ARM64Op_bic_vector_register,	/* 0x0E601C00I
748	//	FMINNM	//ARM64Op_fminnm_vector,	/* 0x0EA0C400F
749	//	FMLS	//ARM64Op_fmls_vector,	/* 0x0EA0CC00FM
750	//	FSUB	//ARM64Op_fsub_vector,	/* 0x0EA0D400FS
751	//	FMIN	//ARM64Op_fmin_vector,	/* 0x0EA0F400FM
752	//	FRSQRTS	//ARM64Op_frsqrts_Vector,	/* 0x0EA0FC00FI
753	//	ORR	//ARM64Op_orr_vector_register,	/* 0x0EA01C00(
754	//	ORN	//ARM64Op_orn_vector,	/* 0x0EE01C00OR
755	//	UHADD	//ARM64Op_uhadd,	/* 0x2E200400UHAI
756	//	UQADD	//ARM64Op_uqadd_Vector,	/* 0x2E200C00U

1	in_use	Opcode	//Opcode	BINARY OI
757	//	URHADD	//ARM64Op_urhadd,	/* 0x2E201400URH
758	//	UHSUB	//ARM64Op_uhsub,	/* 0x2E202400UHSI
759	//		//ARM64Op_ _Vector,	/* 0x2E202C00
760	//	CMHI	//ARM64Op_cmhi_register_Vector,	/* 0x2E20340(
761	//	CMHS	//ARM64Op_cmhs_register_Vector,	/* 0x2E203C(
762	//	USHL	//ARM64Op_ushl_Vector,	/* 0x2E204400US
763	//	UQSHL	//ARM64Op_uqshl_register_Vector,	/* 0x2E204C0(
764	//	URSHL	//ARM64Op_urshl_Vector,	/* 0x2E205400UR
765	//	UQRSHL	//ARM64Op_uqrshl_Vector,	/* 0x2E205C00U(
766	//	UMAX	//ARM64Op_umax,	/* 0x2E206400UMA
767	//	UMIN	//ARM64Op_umin,	/* 0x2E206C00UMIN
768	//	UABD	//ARM64Op_uabd,	/* 0x2E207400UABD
769	//	UABA	//ARM64Op_uaba,	/* 0x2E207C00UABA
770	//	SUB	//ARM64Op_sub_vector_Vector,	/* 0x2E208400
771	//	CMEQ	//ARM64Op_cmeq_register_Vector,	/* 0x2E208C0(
772	//	MLS	//ARM64Op_mls_vector,	/* 0x2E209400ML
773	//	PMUL	//ARM64Op_pmul,	/* 0x2E209C00PMUI
774	//	UMAXP	//ARM64Op_umaxp,	/* 0x2E20A400UMA
775	//	UMINP	//ARM64Op_uminp,	/* 0x2E20AC00UMII
776	//	SQRDMULH	//ARM64Op_sqrdmulh_vector_Vector,	/* 0x2E20B4
777	//	FMAXNMP	//ARM64Op_fmaxnmp_vector,	/* 0x2E20B400
778	//	FADDP	//ARM64Op_faddp_vector,	/* 0x2E20D400FA
779	//	FMUL	//ARM64Op_fmul_vector,	/* 0x2E20DC00FM
780	//	FCMGE	//ARM64Op_fcmge_register_Vector,	/* 0x2E20E40(
781	//	FACGE	//ARM64Op_facge_Vector,	/* 0x2E20EC00FA
782	//	FMAXP	//ARM64Op_fmaxp_vector,	/* 0x2E20F400FM
783	//	FDIV	//ARM64Op_fdiv_vector,	/* 0x2E20FC00FDI
784	//	EOR	//ARM64Op_eor_vector,	/* 0x2E201C00EO
785	//	BSL	//ARM64Op_bsl,	/* 0x2E601C00BSL
786	//	FMINNMP	//ARM64Op_fminnmp_vector,	/* 0x2EA0C400
787	//	FABD	//ARM64Op_fabd_Vector,	/* 0x2EA0D400FA
788	//	FCMGT	//ARM64Op_fcmgt_register_Vector,	/* 0x2EA0E40
789	//	FACGT	//ARM64Op_facgt_Vector,	/* 0x2EA0EC00FA
790	//	FMINP	//ARM64Op_fminp_vector,	/* 0x2EA0F400FM

1	in_use	Opcode	//Opcode	BINARY OI
791	//	BIT	//ARM64Op_bit,	/* 0x2EA01C00BIT
792	//	BIF	//ARM64Op_bif,	/* 0x2EE01C00BIF
793	//	**AdvSIMD three different**	/* AdvSIMD three different */	
794	//	SADDL	//ARM64Op_saddl,	/* 0x0E200000SADD
795	//	SADDL2	//ARM64Op_saddl2,	/* 0x4E200000SADI
796	//	SADDW	//ARM64Op_saddw,	/* 0x0E201000SAD
797	//	SADDW2	//ARM64Op_saddw2,	/* 0x4E201000SAI
798	//	SSUBL	//ARM64Op_ssubl,	/* 0x0E202000SSUB
799	//	SSUBL2	//ARM64Op_ssubl2,	/* 0x4E202000SSUI
800	//	SSUBW	//ARM64Op_ssubw,	/* 0x0E203000SSUI
801	//	SSUBW2	//ARM64Op_ssubw2,	/* 0x4E203000SSU
802	//	ADDHN	//ARM64Op_addhn,	/* 0x0E204000ADDI
803	//	ADDHN2	//ARM64Op_addhn2,	/* 0x4E204000ADD
804	//	SABAL	//ARM64Op_sabal,	/* 0x0E205000SABA
805	//	SABAL2	//ARM64Op_sabal2,	/* 0x4E205000SABA
806	//	SUBHN	//ARM64Op_subhn,	/* 0x0E206000SUBH
807	//	SUBHN2	//ARM64Op_subhn2,	/* 0x4E206000SUB
808	//	SABDL	//ARM64Op_sabdl,	/* 0x0E207000SABD
809	//	SABDL2	//ARM64Op_sabdl2,	/* 0x4E207000SABI
810	//	SMLAL	//ARM64Op_smlal_vector,	/* 0x0E208000SM
811	//	SMLAL2	//ARM64Op_smlal2_vector,	/* 0x4E208000SM
812	//	SQDMLAL	//ARM64Op_sqdmlal_vector_Vector,	/* 0x0E2090(
813	//	SQDMLAL2	//ARM64Op_sqdmlal2_vector_Vector,	/* 0x4E2090(
814	//	SMLSL	//ARM64Op_smlsl_vector,	/* 0x0E20A000SM
815	//	SMLSL2	//ARM64Op_smlsl2_vector,	/* 0x4E20A000SM
816	//	SQDMLSL	//ARM64Op_sqdmlsl_vector_Vector,	/* 0x0E20B0(
817	//	SQDMLSL2	//ARM64Op_sqdmlsl2_vector_Vector,	/* 0x4E20B0(
818	//	SMULL	//ARM64Op_smull_vector,	/* 0x0E20C000SM
819	//	SMULL2	//ARM64Op_smull2_vector,	/* 0x4E20C000SM
820	//	SQDMULL	//ARM64Op_sqdmull_vector_Vector,	/* 0x0E20D0(
821	//	SQDMULL2	//ARM64Op_sqdmull2_vector_Vector,	/* 0x4E20D0
822	//	PMULL	//ARM64Op_pmull,	/* 0x0E20E000PMUL
823	//	PMULL2	//ARM64Op_pmull2,	/* 0x4E20E000PMU
824	//	UADDL	//ARM64Op_uaddl,	/* 0x2E200000UADD

1	in_use	Opcode	//Opcode	BINARY OI
825	//	UADDL2	//ARM64Op_uaddl2,	/* 0x6E200000UADI
826	//	UADDW	//ARM64Op_uaddw,	/* 0x2E201000UAD
827	//	UADDW2	//ARM64Op_uaddw2,	/* 0x6E201000UAI
828	//	USUBL	//ARM64Op_usubl,	/* 0x2E202000USUB
829	//	USUBL2	//ARM64Op_usubl2,	/* 0x6E202000USUI
830	//	USUBW	//ARM64Op_usubw,	/* 0x2E203000USU
831	//	USUBW2	//ARM64Op_usubw2,	/* 0x6E203000USU
832	//	RADDHN	//ARM64Op_raddhn,	/* 0x2E204000RAD
833	//	RADDHN2	//ARM64Op_raddhn2,	/* 0x6E204000RAI
834	//	UABAL	//ARM64Op_uabal,	/* 0x2E205000UABA
835	//	UABAL2	//ARM64Op_uabal2,	/* 0x6E205000UABA
836	//	RSUBHN	//ARM64Op_rsubhn,	/* 0x2E206000RSUI
837	//	RSUBHN2	//ARM64Op_rsubhn2,	/* 0x6E206000RSU
838	//	UABDL	//ARM64Op_uabdl,	/* 0x2E207000UABI
839	//	UABDL2	//ARM64Op_uabdl2,	/* 0x6E207000UABI
840	//	UMLAL	//ARM64Op_umlal_vector,	/* 0x2E208000UM
841	//	UMLAL2	//ARM64Op_umlal2_vector,	/* 0x6E208000UM
842	//	UMLSL	//ARM64Op_umlsl_vector,	/* 0x2E20A000UM
843	//	UMLSL2	//ARM64Op_umlsl2_vector,	/* 0x6E20A000UM
844	//	UMULL	//ARM64Op_umull_vector,	/* 0x2E20C000UM
845	//	UMULL2	//ARM64Op_umull2_vector,	/* 0x6E20C000UI
846	//	**AdvSIMD two-reg misc**	/* AdvSIMD two-reg misc */	
847	//	REV64	//ARM64Op_rev64,	/* 0x0E200800REV6
848	//	REV16	//ARM64Op_rev16_vector,	/* 0x0E201800RE
849	//	SADDLP	//ARM64Op_saddlp,	/* 0x0E202800SADI
850	//	SUQADD	//ARM64Op_suqadd_Vector,	/* 0x0E203800S
851	//	CLS	//ARM64Op_cls_vector,	/* 0x0E204800CLS
852	//	CNT	//ARM64Op_cnt,	/* 0x0E205800CNT
853	//	SADALP	//ARM64Op_sadalp,	/* 0x0E206800SADA
854	//	SQABS	//ARM64Op_sqabs_Vector,	/* 0x0E207800S(
855	//	CMGT	//ARM64Op_cmgt_zero_Vector,	/* 0x0E208800
856	//	CMEQ	//ARM64Op_cmeq_zero_Vector,	/* 0x0E20980(
857	//	CMLT	//ARM64Op_cmlt_zero_Vector,	/* 0x0E20A800(
858	//	ABS	//ARM64Op_abs_Vector,	/* 0x0E20B800AB

1	in_use	Opcode	//Opcode	BINARY OI
859	//	XTN	//ARM64Op_xtn,	/* 0x0E212800XTN
860	//	XTN2	//ARM64Op_xtn2,	/* 0x0E212800XTN2
861	//	SQXTN	//ARM64Op_sqxtn_Vector,	/* 0x0E214800SC
862	//	SQXTN2	//ARM64Op_sqxtn2_Vector,	/* 0x0E214800S(
863	//	FCVTN	//ARM64Op_fcvtn,	/* 0x0E216800FCVTN
864	//	FCVTN2	//ARM64Op_fcvtn2,	/* 0x0E216800FCVT
865	//	FCVTL	//ARM64Op_fcvtl,	/* 0x0E217800FCVTL
866	//	FCVTL2	//ARM64Op_fcvtl2,	/* 0x0E217800FCVTL
867	//	FRINTN	//ARM64Op_frintn_vector,	/* 0x0E218800FRI
868	//	FRINTM	//ARM64Op_frintm_vector,	/* 0x0E219800FR
869	//	FCVTNS	//ARM64Op_fcvtns_vector_Vector,	/* 0x0E21A80(
870	//	FCVTMS	//ARM64Op_fcvtms_vector_Vector,	/* 0x0E21B80
871	//	FCVTAS	//ARM64Op_fcvtas_vector_Vector,	/* 0x0E21C80(
872	//	SCVTF	//ARM64Op_scvtf_vector_integer_Vector,	/* 0x0E21D8
873	//	FCMGT	//ARM64Op_fcmgt_zero_Vector,	/* 0x0EA0C80(
874	//	FCMEQ	//ARM64Op_fcmeq_zero_Vector,	/* 0x0EA0D80
875	//	FCMLT	//ARM64Op_fcmlt_zero_Vector,	/* 0x0EA0E800
876	//	FABS	//ARM64Op_fabs_vector,	/* 0x0EA0F800FA
877	//	FRINTP	//ARM64Op_frintp_vector,	/* 0x0EA18800FRI
878	//	FRINTZ	//ARM64Op_frintz_vector,	/* 0x0EA19800FRI
879	//	FCVTPS	//ARM64Op_fcvtps_vector_Vector,	/* 0x0EA1A80(
880	//	FCVTZS	//ARM64Op_fcvtzs_vector_integer_Vector,	/* 0x0EA1B
881	//	URECPE	//ARM64Op_urecpe,	/* 0x0EA1C800URE
882	//	FRECPE	//ARM64Op_frecpe_Vector,	/* 0x0EA1D800F
883	//	REV32	//ARM64Op_rev32_vector,	/* 0x2E200800RE
884	//	UADDLP	//ARM64Op_uaddlp,	/* 0x2E202800UADI
885	//	USQADD	//ARM64Op_usqadd_Vector,	/* 0x2E203800U
886	//	CLZ	//ARM64Op_clz_vector,	/* 0x2E204800CLZ
887	//	UADALP	//ARM64Op_uadalp,	/* 0x2E206800UAD/
888	//	SQNEG	//ARM64Op_sqneg_Vector,	/* 0x2E207800S(
889	//	CMGE	//ARM64Op_cmge_zero_Vector,	/* 0x2E20880(
890	//	CMLE	//ARM64Op_cmle_zero_Vector,	/* 0x2E209800
891	//	NEG	//ARM64Op_neg_vector_Vector,	/* 0x2E20B80(
892	//	SQXTUN	//ARM64Op_sqxtun_Vector,	/* 0x2E212800S(

1	in_use	Opcode	//Opcode	BINARY OI
893	//	SQXTUN2	//ARM64Op_sqxtun2_Vector,	/* 0x2E212800S
894	//	SHLL	//ARM64Op_shll,	/* 0x2E213800SHLL
895	//	SHLL2	//ARM64Op_shll2,	/* 0x2E213800SHLL2
896	//	UQXTN	//ARM64Op_uqxtn_Vector,	/* 0x2E214800UC
897	//	UQXTN2	//ARM64Op_uqxtn2_Vector,	/* 0x2E214800U
898	//	FCVTXN	//ARM64Op_fcvtxn_Vector,	/* 0x2E216800FC
899	//	FCVTXN2	//ARM64Op_fcvtxn2_Vector,	/* 0x2E216800F(
900	//	FRINTA	//ARM64Op_frinta_vector,	/* 0x2E218800FRI
901	//	FRINTX	//ARM64Op_frintx_vector,	/* 0x2E219800FRI
902	//	FCVTNU	//ARM64Op_fcvtnu_vector_Vector,	/* 0x2E21A80(
903	//	FCVTMU	//ARM64Op_fcvtmu_vector_Vector,	/* 0x2E21B80
904	//	FCVTAU	//ARM64Op_fcvtau_vector_Vector,	/* 0x2E21C80(
905	//	UCVTF	//ARM64Op_ucvtf_vector_integer_Vector,	/* 0x2E21D8
906	//	NOT	//ARM64Op_not,	/* 0x2E205800NOT
907	//	RBIT	//ARM64Op_rbit_vector,	/* 0x2E605800RBIT
908	//	FCMGE	//ARM64Op_fcmge_zero_Vector,	/* 0x2EA0C80
909	//	FCMLE	//ARM64Op_fcmle_zero_Vector,	/* 0x2EA0D80(
910	//	FNEG	//ARM64Op_fneg_vector,	/* 0x2EA0F800FN
911	//	FRINTI	//ARM64Op_frinti_vector,	/* 0x2EA19800FRIN
912	//	FCVTPU	//ARM64Op_fcvtpu_vector_Vector,	/* 0x2EA1A80(
913	//	FCVTZU	//ARM64Op_fcvtzu_vector_integer_Vector,	/* 0x2EA1B
914	//	URSQRTE	//ARM64Op_ursqrte,	/* 0x2EA1C800URS(
915	//	FRSQRTE	//ARM64Op_frsqrte_Vector,	/* 0x2EA1D800FF
916	//	FSQRT	//ARM64Op_fsqrt_vector,	/* 0x2EA1F800FS(
917	//	**AdvSIMD across lanes**	/* AdvSIMD across lanes */	
918	//	SADDLV	//ARM64Op_saddlv,	/* 0x0E303800SADL
919	//	SMAXV	//ARM64Op_smaxv,	/* 0x0E30A800SMA
920	//	SMINV	//ARM64Op_sminv,	/* 0x0E31A800SMIN
921	//	ADDV	//ARM64Op_addv,	/* 0x0E31B800ADDV
922	//	UADDLV	//ARM64Op_uaddlv,	/* 0x2E303800UADL
923	//	UMAXV	//ARM64Op_umaxv,	/* 0x2E30A800UMA
924	//	UMINV	//ARM64Op_uminv,	/* 0x2E31A800UMIN
925	//	FMAXNMV	//ARM64Op_fmaxnmv,	/* 0x2E30C800FM
926	//	FMAXV	//ARM64Op_fmaxv,	/* 0x2E30F800FMA

1	in_use	Opcode	//Opcode	BINARY OI
927	//	FMINNMV	//ARM64Op_fminnmv,	/* 0x2EB0C800FM
928	//	FMINV	//ARM64Op_fminv,	/* 0x2EB0F800FMIN
929	//	**AdvSIMD copy**	/* AdvSIMD copy */	
930	//	DUP	//ARM64Op_dup_element_Vector,	/* 0x0E00040
931	//	DUP	//ARM64Op_dup_general,	/* 0x0E000C00DU
932	//	SMOV	//ARM64Op_smov_32_bit,	/* 0x0E002C00SI
933	//	UMOV	//ARM64Op_umov_32_bit,	/* 0x0E003C00U
934	//	INS	//ARM64Op_ins_general,	/* 0x4E001C00INS
935	//	SMOV	//ARM64Op_smov_64_bit,	/* 0x4E002C00SI
936	//	UMOV	//ARM64Op_umov_64_bit,	/* 0x4E003C00U
937	//	INS	//ARM64Op_ins_element,	/* 0x6E000400INS
938	//	**AdvSIMD vector x indexe**	/* AdvSIMD vector x indexed element */	
939	//	SMLAL	//ARM64Op_smlal_by_element,	/* 0x0F002000
940	//	SMLAL2	//ARM64Op_smlal2_by_element,	/* 0x0F00200(
941	//	SQDMLAL	//ARM64Op_sqdmlal_by_element_Vector,	/* 0x0F003
942	//	SQDMLAL2	//ARM64Op_sqdmlal2_by_element_Vector,	/* 0x0F00
943	//	SMLSL	//ARM64Op_smlsl_by_element,	/* 0x0F006000
944	//	SMLSL2	//ARM64Op_smlsl2_by_element,	/* 0x0F00600(
945	//	SQDMLSL	//ARM64Op_sqdmlsl_by_element_Vector,	/* 0x0F007
946	//	SQDMLSL2	//ARM64Op_sqdmlsl2_by_element_Vector,	/* 0x0F00
947	//	MUL	//ARM64Op_mul_by_element,	/* 0x0F008000I
948	//	SMULL	//ARM64Op_smull_by_element,	/* 0x0F00A000
949	//	SMULL2	//ARM64Op_smull2_by_element,	/* 0x0F00A00(
950	//	SQDMULL	//ARM64Op_sqdmull_by_element_Vector,	/* 0x0F00E
951	//	SQDMULL2	//ARM64Op_sqdmull2_by_element_Vector,	/* 0x0F00
952	//	SQDMULH	//ARM64Op_sqdmulh_by_element_Vector,	/* 0x0F00
953	//	SQRDMULH	//ARM64Op_sqrdmulh_by_element_Vector,	/* 0x0F0(
954	//	FMLA	//ARM64Op_fmla_by_element_Vector,	/* 0x0F801(
955	//	FMLS	//ARM64Op_fmls_by_element_Vector,	/* 0x0F805(
956	//	FMUL	//ARM64Op_fmul_by_element_Vector,	/* 0x0F809(
957	//	MLA	//ARM64Op_mla_by_element,	/* 0x2F000000I
958	//	UMLAL	//ARM64Op_umlal_by_element,	/* 0x2F002000
959	//	UMLAL2	//ARM64Op_umlal2_by_element,	/* 0x2F00200(
960	//	MLS	//ARM64Op_mls_by_element,	/* 0x2F004000I

1	in_use	Opcode	//Opcode	BINARY OI
961	//	UMLSL	//ARM64Op_umlsl_by_element,	/* 0x2F006000
962	//	UMLSL2	//ARM64Op_umlsl2_by_element,	/* 0x2F006000
963	//	UMULL	//ARM64Op_umull_by_element,	/* 0x2F00A000
964	//	UMULL2	//ARM64Op_umull2_by_element,	/* 0x2F00A000
965	//	FMULX	//ARM64Op_fmulx_by_element_Vector,	/* 0x2F809(
966	//	**AdvSIMD modified immed**	/* AdvSIMD modified immediate */	
967	//	MOVI	//ARM64Op_movi_32_bit_shifted_immediate,	/* 0x0F00
968	//	ORR	//ARM64Op_orr_vector_immediate_32_bit,	/* 0x0F001
969	//	MOVI	//ARM64Op_movi_16_bit_shifted_immediate,	/* 0x0F00
970	//	ORR	//ARM64Op_orr_vector_immediate_16_bit,	/* 0x0F009
971	//	MOVI	//ARM64Op_movi_32_bit_shifting_ones,	/* 0x0F00C4
972	//	MOVI	//ARM64Op_movi_8_bit,	/* 0x0F00E400MO
973	//	FMOV	//ARM64Op_fmov_vector_immediate_Single_precision,	/* 0x(
974	//	MVNI	//ARM64Op_mvni_32_bit_shifted_immediate,	/* 0x2F00
975	//	BIC	//ARM64Op_bic_vector_immediate_32_bit,	/* 0x2F001
976	//	MVNI	//ARM64Op_mvni_16_bit_shifted_immediate,	/* 0x2F00
977	//	BIC	//ARM64Op_bic_vector_immediate_16_bit,	/* 0x2F009
978	//	MVNI	//ARM64Op_mvni_32_bit_shifting_ones,	/* 0x2F00C4
979	//	MOVI	//ARM64Op_movi_64_bit_scalar,	/* 0x2F00E400
980	//	MOVI	//ARM64Op_movi_64_bit_vector,	/* 0x6F00E400
981	//	FMOV	//ARM64Op_fmov_vector_immediate_Double_precision,	/* 0x
982	//	**AdvSIMD shift by immedi**	/* AdvSIMD shift by immediate */	
983	//	SSHR	//ARM64Op_sshr_Vector,	/* 0x0F000400SS
984	//	SSRA	//ARM64Op_ssra_Vector,	/* 0x0F001400SS
985	//	SRSHR	//ARM64Op_srshr_Vector,	/* 0x0F002400SR
986	//	SRSRA	//ARM64Op_srsra_Vector,	/* 0x0F003400SR
987	//	SHL	//ARM64Op_shl_Vector,	/* 0x0F005400SHL
988	//	SQSHL	//ARM64Op_sqshl_immediate_Vector,	/* 0x0F0074
989	//	SHRN	//ARM64Op_shrn,	/* 0x0F008400SHRN
990	//	SHRN2	//ARM64Op_shrn2,	/* 0x0F008400SHRN
991	//	RSHRN	//ARM64Op_rshrn,	/* 0x0F008C00RSHR
992	//	RSHRN2	//ARM64Op_rshrn2,	/* 0x0F008C00RSHR
993	//	SQSHRN	//ARM64Op_sqshrn_Vector,	/* 0x0F009400S(
994	//	SQSHRN2	//ARM64Op_sqshrn2_Vector,	/* 0x0F009400S

1	in_use	Opcode	//Opcode	BINARY Ol
995	//	SQRSHRN	//ARM64Op_sqrshrn_Vector,	/* 0x0F009C00S
996	//	SQRSHRN2	//ARM64Op_sqrshrn2_Vector,	/* 0x0F009C00
997	//	SSHLL	//ARM64Op_sshll,	/* 0x0F00A400SSHLL
998	//	SSHLL2	//ARM64Op_sshll2,	/* 0x0F00A400SSHL
999	//	SCVTF	//ARM64Op_scvtf_vector_fixed_point_Vector,	/* 0x0F00E
1000	//	FCVTZS	//ARM64Op_fcvtzs_vector_fixed_point_Vector,	/* 0x0F00
1001	//	USHR	//ARM64Op_ushr_Vector,	/* 0x2F000400US
1002	//	USRA	//ARM64Op_usra_Vector,	/* 0x2F001400US
1003	//	URSHR	//ARM64Op_urshr_Vector,	/* 0x2F002400UR
1004	//	URSRA	//ARM64Op_ursra_Vector,	/* 0x2F003400UR
1005	//	SRI	//ARM64Op_sri_Vector,	/* 0x2F004400SRI
1006	//	SLI	//ARM64Op_sli_Vector,	/* 0x2F005400SLI
1007	//	SQSHLU	//ARM64Op_sqshlu_Vector,	/* 0x2F006400S
1008	//	UQSHL	//ARM64Op_uqshl_immediate_Vector,	/* 0x2F0074
1009	//	SQSHRUN	//ARM64Op_sqshrun_Vector,	/* 0x2F008400S
1010	//	SQSHRUN2	//ARM64Op_sqshrun2_Vector,	/* 0x2F00840
1011	//	SQRSHRUN	//ARM64Op_sqrshrun_Vector,	/* 0x2F008C00
1012	//	SQRSHRUN2	//ARM64Op_sqrshrun2_Vector,	/* 0x2F008C00
1013	//	UQSHRN	//ARM64Op_uqshrn_Vector,	/* 0x2F009400U
1014	//	UQRSHRN	//ARM64Op_uqrshrn_Vector,	/* 0x2F009C00U
1015	//	UQRSHRN2	//ARM64Op_uqrshrn2_Vector,	/* 0x2F009C00
1016	//	USHLL	//ARM64Op_ushll,	/* 0x2F00A400USHLL
1017	//	USHLL2	//ARM64Op_ushll2,	/* 0x2F00A400USHL
1018	//	UCVTF	//ARM64Op_ucvtf_vector_fixed_point_Vector,	/* 0x2F00E
1019	//	FCVTZU	//ARM64Op_fcvtzu_vector_fixed_point_Vector,	/* 0x2F00
1020	//	**AdvSIMD TBL/TBX**	/* AdvSIMD TBL/TBX */	
1021	//	TBL	//ARM64Op_tbl_Single_register_table,	/* 0x0E00000
1022	//	TBX	//ARM64Op_tbx_Single_register_table,	/* 0x0E00100
1023	//	TBL	//ARM64Op_tbl_Two_register_table,	/* 0x0E00200
1024	//	TBX	//ARM64Op_tbx_Two_register_table,	/* 0x0E00300
1025	//	TBL	//ARM64Op_tbl_Three_register_table,	/* 0x0E00400
1026	//	TBX	//ARM64Op_tbx_Three_register_table,	/* 0x0E00500
1027	//	TBL	//ARM64Op_tbl_Four_register_table,	/* 0x0E00600
1028	//	TBX	//ARM64Op_tbx_Four_register_table,	/* 0x0E00700

```
1     in_use      Opcode                  //Opcode                                        BINARY   Ol
1029  //        AdvSIMD ZIP/UZP/TRN       /* AdvSIMD ZIP/UZP/TRN */
1030  //            UZP1                  //ARM64Op_uzp1,                                 /* 0x0E001800UZP1
1031  //            TRN1                  //ARM64Op_trn1,                                /* 0x0E002800TRN1
1032  //            ZIP1                  //ARM64Op_zip1,                                 /* 0x0E003800ZIP1
1033  //            UZP2                  //ARM64Op_uzp2,                                 /* 0x0E005800UZP2
1034  //            TRN2                  //ARM64Op_trn2,                                /* 0x0E006800TRN2
1035  //            ZIP2                  //ARM64Op_zip2,                                 /* 0x0E007800ZIP2
1036  //        AdvSIMD EXT               /* AdvSIMD EXT */
1037  //            EXT                   //ARM64Op_ext,                                 /* 0x2E000000EXT
1038  //      Loads and stores            /* Loads and stores */
1039  //        AdvSIMD load/store multi  /* AdvSIMD load/store multiple structures */
1040  //            ST4                   //ARM64Op_st4_multiple_structures_No_offset,                /* 0x0C0
1041  //            ST1                   //ARM64Op_st1_multiple_structures_Four_registers,            /* 0x0C
1042  //            ST3                   //ARM64Op_st3_multiple_structures_No_offset,                /* 0x0C0
1043  //            ST1                   //ARM64Op_st1_multiple_structures_Three_registers,           /* 0x0C
1044  //            ST1                   //ARM64Op_st1_multiple_structures_One_register,              /* 0x0C
1045  //            ST2                   //ARM64Op_st2_multiple_structures_No_offset,                /* 0x0C0
1046  //            ST1                   //ARM64Op_st1_multiple_structures_Two_registers,             /* 0x0C
1047  //            LD4                   //ARM64Op_ld4_multiple_structures_No_offset,                /* 0x0C4
1048  //            LD1                   //ARM64Op_ld1_multiple_structures_Four_registers,            /* 0x0C4
1049  //            LD3                   //ARM64Op_ld3_multiple_structures_No_offset,                /* 0x0C4
1050  //            LD1                   //ARM64Op_ld1_multiple_structures_Three_registers,           /* 0x0C
1051  //            LD1                   //ARM64Op_ld1_multiple_structures_One_register,              /* 0x0C4
1052  //            LD2                   //ARM64Op_ld2_multiple_structures_No_offset,                /* 0x0C4
1053  //            LD1                   //ARM64Op_ld1_multiple_structures_Two_registers,             /* 0x0C
1054  //        AdvSIMD load/store multi  /* AdvSIMD load/store multiple structures (post-indexed) */
1055  //            ST4                   //ARM64Op_st4_multiple_structures_Register_offset,          /* 0x0C8
1056  //            ST1                   //ARM64Op_st1_multiple_structures_Four_registers_register_offset, /*
1057  //            ST3                   //ARM64Op_st3_multiple_structures_Register_offset,          /* 0x0C8
1058  //            ST1                   //ARM64Op_st1_multiple_structures_Three_registers_register_offset,/*
1059  //            ST1                   //ARM64Op_st1_multiple_structures_One_register_register_offset,  /*
1060  //            ST2                   //ARM64Op_st2_multiple_structures_Register_offset,          /* 0x0C8
1061  //            ST1                   //ARM64Op_st1_multiple_structures_Two_registers_register_offset,  /*
1062  //            ST4                   //ARM64Op_st4_multiple_structures_Immediate_offset,         /* 0x0
```

```
1     in_use      Opcode                  //Opcode                                        BINARY   Ol
1063  //          ST1                     //ARM64Op_st1_multiple_structures_Four_registers_immediate_offset,
1064  //          ST3                     //ARM64Op_st3_multiple_structures_Immediate_offset,             /* 0x0(
1065  //          ST1                     //ARM64Op_st1_multiple_structures_Three_registers_immediate_offse
1066  //          ST1                     //ARM64Op_st1_multiple_structures_One_register_immediate_offset,  /
1067  //          ST2                     //ARM64Op_st2_multiple_structures_Immediate_offset,             /* 0x0(
1068  //          ST1                     //ARM64Op_st1_multiple_structures_Two_registers_immediate_offset,
1069  //          LD4                     //ARM64Op_ld4_multiple_structures_Register_offset,              /* 0x0C(
1070  //          LD1                     //ARM64Op_ld1_multiple_structures_Four_registers_register_offset, /*
1071  //          LD3                     //ARM64Op_ld3_multiple_structures_Register_offset,              /* 0x0C(
1072  //          LD1                     //ARM64Op_ld1_multiple_structures_Three_registers_register_offset,/*
1073  //          LD1                     //ARM64Op_ld1_multiple_structures_One_register_register_offset,  /* (
1074  //          LD2                     //ARM64Op_ld2_multiple_structures_Register_offset,              /* 0x0C(
1075  //          LD1                     //ARM64Op_ld1_multiple_structures_Two_registers_register_offset,  /*
1076  //          LD4                     //ARM64Op_ld4_multiple_structures_Immediate_offset,             /* 0x0(
1077  //          LD1                     //ARM64Op_ld1_multiple_structures_Four_registers_immediate_offset,
1078  //          LD3                     //ARM64Op_ld3_multiple_structures_Immediate_offset,             /* 0x0(
1079  //          LD1                     //ARM64Op_ld1_multiple_structures_Three_registers_immediate_offse
1080  //          LD1                     //ARM64Op_ld1_multiple_structures_One_register_immediate_offset,  /
1081  //          LD2                     //ARM64Op_ld2_multiple_structures_Immediate_offset,             /* 0x0(
1082  //          LD1                     //ARM64Op_ld1_multiple_structures_Two_registers_immediate_offset,
1083  //        AdvSIMD load/store singl  /* AdvSIMD load/store single structure */
1084  //          ST1                     //ARM64Op_st1_single_structure_8_bit,                  /* 0x0D0000(
1085  //          ST3                     //ARM64Op_st3_single_structure_8_bit,                  /* 0x0D0020(
1086  //          ST1                     //ARM64Op_st1_single_structure_16_bit,                  /* 0x0D0040
1087  //          ST3                     //ARM64Op_st3_single_structure_16_bit,                  /* 0x0D0060
1088  //          ST1                     //ARM64Op_st1_single_structure_32_bit,                  /* 0x0D0080
1089  //          ST1                     //ARM64Op_st1_single_structure_64_bit,                  /* 0x0D0084
1090  //          ST3                     //ARM64Op_st3_single_structure_32_bit,                  /* 0x0D00A(
1091  //          ST3                     //ARM64Op_st3_single_structure_64_bit,                  /* 0x0D00A4
1092  //          ST2                     //ARM64Op_st2_single_structure_8_bit,                  /* 0x0D2000(
1093  //          ST4                     //ARM64Op_st4_single_structure_8_bit,                  /* 0x0D2020(
1094  //          ST2                     //ARM64Op_st2_single_structure_16_bit,                  /* 0x0D2040
1095  //          ST4                     //ARM64Op_st4_single_structure_16_bit,                  /* 0x0D2060
1096  //          ST2                     //ARM64Op_st2_single_structure_32_bit,                  /* 0x0D2080
```

1	in_use	Opcode	//Opcode	BINARY OI
1097	//	ST2	//ARM64Op_st2_single_structure_64_bit,	/* 0x0D2084
1098	//	ST4	//ARM64Op_st4_single_structure_32_bit,	/* 0x0D20A0
1099	//	ST4	//ARM64Op_st4_single_structure_64_bit,	/* 0x0D20A4
1100	//	LD1	//ARM64Op_ld1_single_structure_8_bit,	/* 0x0D4000(
1101	//	LD3	//ARM64Op_ld3_single_structure_8_bit,	/* 0x0D4020(
1102	//	LD1	//ARM64Op_ld1_single_structure_16_bit,	/* 0x0D4040
1103	//	LD3	//ARM64Op_ld3_single_structure_16_bit,	/* 0x0D4060
1104	//	LD1	//ARM64Op_ld1_single_structure_32_bit,	/* 0x0D4080
1105	//	LD1	//ARM64Op_ld1_single_structure_64_bit,	/* 0x0D4084
1106	//	LD3	//ARM64Op_ld3_single_structure_32_bit,	/* 0x0D40A0
1107	//	LD3	//ARM64Op_ld3_single_structure_64_bit,	/* 0x0D40A4
1108	//	LD1R	//ARM64Op_ld1r_No_offset,	/* 0x0D40C000LI
1109	//	LD3R	//ARM64Op_ld3r_No_offset,	/* 0x0D40E000LI
1110	//	LD2	//ARM64Op_ld2_single_structure_8_bit,	/* 0x0D6000(
1111	//	LD4	//ARM64Op_ld4_single_structure_8_bit,	/* 0x0D6020(
1112	//	LD2	//ARM64Op_ld2_single_structure_16_bit,	/* 0x0D6040
1113	//	LD4	//ARM64Op_ld4_single_structure_16_bit,	/* 0x0D6060
1114	//	LD2	//ARM64Op_ld2_single_structure_32_bit,	/* 0x0D6080
1115	//	LD2	//ARM64Op_ld2_single_structure_64_bit,	/* 0x0D6084
1116	//	LD4	//ARM64Op_ld4_single_structure_32_bit,	/* 0x0D60A0
1117	//	LD4	//ARM64Op_ld4_single_structure_64_bit,	/* 0x0D60A4
1118	//	LD2R	//ARM64Op_ld2r_No_offset,	/* 0x0D60C000LI
1119	//	LD4R	//ARM64Op_ld4r_No_offset,	/* 0x0D60E000LI
1120	//	**AdvSIMD load/store singl**	/* AdvSIMD load/store single structure (post-indexed) */	
1121	//	ST1	//ARM64Op_st1_single_structure_8_bit_register_offset,	/* 0x0D
1122	//	ST3	//ARM64Op_st3_single_structure_8_bit_register_offset,	/* 0x0D
1123	//	ST1	//ARM64Op_st1_single_structure_16_bit_register_offset,	/* 0x0I
1124	//	ST3	//ARM64Op_st3_single_structure_16_bit_register_offset,	/* 0x0I
1125	//	ST1	//ARM64Op_st1_single_structure_32_bit_register_offset,	/* 0x0I
1126	//	ST1	//ARM64Op_st1_single_structure_64_bit_register_offset,	/* 0x0I
1127	//	ST3	//ARM64Op_st3_single_structure_32_bit_register_offset,	/* 0x0I
1128	//	ST3	//ARM64Op_st3_single_structure_64_bit_register_offset,	/* 0x0I
1129	//	ST1	//ARM64Op_st1_single_structure_8_bit_immediate_offset,	/* 0x(
1130	//	ST3	//ARM64Op_st3_single_structure_8_bit_immediate_offset,	/* 0x(

1	in_use	Opcode	//Opcode	BINARY	OI
1131	//	ST1	//ARM64Op_st1_single_structure_16_bit_immediate_offset,		/* 0x
1132	//	ST3	//ARM64Op_st3_single_structure_16_bit_immediate_offset,		/* 0x
1133	//	ST1	//ARM64Op_st1_single_structure_32_bit_immediate_offset,		/* 0x
1134	//	ST1	//ARM64Op_st1_single_structure_64_bit_immediate_offset,		/* 0x
1135	//	ST3	//ARM64Op_st3_single_structure_32_bit_immediate_offset,		/* 0x
1136	//	ST3	//ARM64Op_st3_single_structure_64_bit_immediate_offset,		/* 0x
1137	//	ST2	//ARM64Op_st2_single_structure_8_bit_register_offset,		/* 0x0D
1138	//	ST4	//ARM64Op_st4_single_structure_8_bit_register_offset,		/* 0x0D
1139	//	ST2	//ARM64Op_st2_single_structure_16_bit_register_offset,		/* 0x0E
1140	//	ST4	//ARM64Op_st4_single_structure_16_bit_register_offset,		/* 0x0E
1141	//	ST2	//ARM64Op_st2_single_structure_32_bit_register_offset,		/* 0x0E
1142	//	ST2	//ARM64Op_st2_single_structure_64_bit_register_offset,		/* 0x0E
1143	//	ST4	//ARM64Op_st4_single_structure_32_bit_register_offset,		/* 0x0E
1144	//	ST4	//ARM64Op_st4_single_structure_64_bit_register_offset,		/* 0x0E
1145	//	ST2	//ARM64Op_st2_single_structure_8_bit_immediate_offset,		/* 0x(
1146	//	ST4	//ARM64Op_st4_single_structure_8_bit_immediate_offset,		/* 0x(
1147	//	ST2	//ARM64Op_st2_single_structure_16_bit_immediate_offset,		/* 0x
1148	//	ST4	//ARM64Op_st4_single_structure_16_bit_immediate_offset,		/* 0x
1149	//	ST2	//ARM64Op_st2_single_structure_32_bit_immediate_offset,		/* 0x
1150	//	ST2	//ARM64Op_st2_single_structure_64_bit_immediate_offset,		/* 0x
1151	//	ST4	//ARM64Op_st4_single_structure_32_bit_immediate_offset,		/* 0x
1152	//	ST4	//ARM64Op_st4_single_structure_64_bit_immediate_offset,		/* 0x
1153	//	LD1	//ARM64Op_ld1_single_structure_8_bit_register_offset,		/* 0x0D
1154	//	LD3	//ARM64Op_ld3_single_structure_8_bit_register_offset,		/* 0x0D
1155	//	LD1	//ARM64Op_ld1_single_structure_16_bit_register_offset,		/* 0x0E
1156	//	LD3	//ARM64Op_ld3_single_structure_16_bit_register_offset,		/* 0x0E
1157	//	LD1	//ARM64Op_ld1_single_structure_32_bit_register_offset,		/* 0x0E
1158	//	LD1	//ARM64Op_ld1_single_structure_64_bit_register_offset,		/* 0x0E
1159	//	LD3	//ARM64Op_ld3_single_structure_32_bit_register_offset,		/* 0x0E
1160	//	LD3	//ARM64Op_ld3_single_structure_64_bit_register_offset,		/* 0x0E
1161	//	LD1R	//ARM64Op_ld1r_Register_offset,		/* 0x0DC0C00(
1162	//	LD3R	//ARM64Op_ld3r_Register_offset,		/* 0x0DC0E000
1163	//	LD1	//ARM64Op_ld1_single_structure_8_bit_immediate_offset,		/* 0x(
1164	//	LD3	//ARM64Op_ld3_single_structure_8_bit_immediate_offset,		/* 0x(

1	in_use	Opcode	//Opcode	BINARY	Ol
1165	//	LD1	//ARM64Op_ld1_single_structure_16_bit_immediate_offset,		/* 0x
1166	//	LD3	//ARM64Op_ld3_single_structure_16_bit_immediate_offset,		/* 0x
1167	//	LD1	//ARM64Op_ld1_single_structure_32_bit_immediate_offset,		/* 0x
1168	//	LD1	//ARM64Op_ld1_single_structure_64_bit_immediate_offset,		/* 0x
1169	//	LD3	//ARM64Op_ld3_single_structure_32_bit_immediate_offset,		/* 0x
1170	//	LD3	//ARM64Op_ld3_single_structure_64_bit_immediate_offset,		/* 0x
1171	//	LD1R	//ARM64Op_ld1r_Immediate_offset,		/* 0x0DDFC0(
1172	//	LD3R	//ARM64Op_ld3r_Immediate_offset,		/* 0x0DDFE0(
1173	//	LD2	//ARM64Op_ld2_single_structure_8_bit_register_offset,		/* 0x0D
1174	//	LD4	//ARM64Op_ld4_single_structure_8_bit_register_offset,		/* 0x0D
1175	//	LD2	//ARM64Op_ld2_single_structure_16_bit_register_offset,		/* 0x0[
1176	//	LD4	//ARM64Op_ld4_single_structure_16_bit_register_offset,		/* 0x0[
1177	//	LD2	//ARM64Op_ld2_single_structure_32_bit_register_offset,		/* 0x0[
1178	//	LD2	//ARM64Op_ld2_single_structure_64_bit_register_offset,		/* 0x0[
1179	//	LD4	//ARM64Op_ld4_single_structure_32_bit_register_offset,		/* 0x0[
1180	//	LD4	//ARM64Op_ld4_single_structure_64_bit_register_offset,		/* 0x0[
1181	//	LD2R	//ARM64Op_ld2r_Register_offset,		/* 0x0DE0C000
1182	//	LD4R	//ARM64Op_ld4r_Register_offset,		/* 0x0DE0E000
1183	//	LD2	//ARM64Op_ld2_single_structure_8_bit_immediate_offset,		/* 0x(
1184	//	LD4	//ARM64Op_ld4_single_structure_8_bit_immediate_offset,		/* 0x(
1185	//	LD2	//ARM64Op_ld2_single_structure_16_bit_immediate_offset,		/* 0x
1186	//	LD4	//ARM64Op_ld4_single_structure_16_bit_immediate_offset,		/* 0x
1187	//	LD2	//ARM64Op_ld2_single_structure_32_bit_immediate_offset,		/* 0x
1188	//	LD2	//ARM64Op_ld2_single_structure_64_bit_immediate_offset,		/* 0x
1189	//	LD4	//ARM64Op_ld4_single_structure_32_bit_immediate_offset,		/* 0x
1190	//	LD4	//ARM64Op_ld4_single_structure_64_bit_immediate_offset,		/* 0x
1191	//	LD2R	//ARM64Op_ld2r_Immediate_offset,		/* 0x0DFFC0(
1192	//	LD4R	//ARM64Op_ld4r_Immediate_offset,		/* 0x0DFFE0(

	in_use	Opcode	//BINARY	Opcode	Opcodecomments
2		**UNALLOCATED**	/* UNALLOCATED */		
3		BAD	0x00000000,/* BAD	ARM64Op_bad	invalid operation */
4		**Branch,exception gener**	/* Branch,exception generation and system Instruction */		
5		**Compare _ Branch (imme**	/* Compare _ Branch (immediate) */		
6		CBZ	0x34000000,/* CBZ	ARM64Op_cbz	_32_bit */
7		CBNZ	0x35000000,/* CBNZ	ARM64Op_cbnz	_32_bit */
8		CBZ	0xB4000000,/* CBZ	ARM64Op_cbz	_64_bit */
9		CBNZ	0xB5000000,/* CBNZ	ARM64Op_cbnz	_64_bit */
10		**Test & branch (immediate**	/* Test & branch (immediate) */		
11		TBZ	0x36000000,/* TBZ	ARM64Op_tbz	*/
12		TBNZ	0x37000000,/* TBNZ	ARM64Op_tbnz	*/
13		**Conditional branch (imme**	/* Conditional branch (immediate) */		
14		B_cond	0x54000000,/* B_cond	ARM64Op_b_cond	*/
15		**Exception generation**	/* Exception generation */		
16		SVC	0xD4000001,/* SVC	ARM64Op_svc	*/
17		HVC	0xD4000002,/* HVC	ARM64Op_hvc	*/
18		SMC	0xD4000003,/* SMC	ARM64Op_smc	*/
19		BRK	0xD4200000,/* BRK	ARM64Op_brk	*/
20		HLT	0xD4400000,/* HLT	ARM64Op_hlt	*/
21		DCPS1	0xD4A00001,/* DCPS1	ARM64Op_dcps1	*/
22		DCPS2	0xD4A00002,/* DCPS2	ARM64Op_dcps2	*/
23		DCPS3	0xD4A00003,/* DCPS3	ARM64Op_dcps3	*/
24		**System**	/* System */		
25		MSR	0xD500401F,/* MSR	ARM64Op_msr_immediate	*/
26		HINT	0xD503201F,/* HINT	ARM64Op_hint	*/
27		CLREX	0xD503305F,/* CLREX	ARM64Op_clrex	*/
28		DSB	0xD503309F,/* DSB	ARM64Op_dsb	*/
29		DMB	0xD50330BF,/* DMB	ARM64Op_dmb	*/
30		ISB	0xD50330DF,/* ISB	ARM64Op_isb	*/
31		SYS	0xD5080000,/* SYS	ARM64Op_sys	*/
32		MSR	0xD5100000,/* MSR	ARM64Op_msr_register	*/
33		SYSL	0xD5280000,/* SYSL	ARM64Op_sysl	*/
34		MRS	0xD5300000,/* MRS	ARM64Op_mrs	*/
35		**Unconditional branch (reg**	/* Unconditional branch (register) */		
36		BR	0xD61F0000,/* BR	ARM64Op_br	*/
37		BLR	0xD63F0000,/* BLR	ARM64Op_blr	*/
38		RET	0xD65F0000,/* RET	ARM64Op_ret	*/
39		ERET	0xD69F03E0,/* ERET	ARM64Op_eret	*/

1	in_use	Opcode	//BINARY Opcode Opcodecomments
40		DRPS	0xD6BF03E0,/* DRPS ARM64Op_drps */
41		**Unconditional branch (im**	/* Unconditional branch (immediate) */
42		B	0x14000000,/* B ARM64Op_b */
43		BL	0x94000000,/* BL ARM64Op_bl */
44		**Loads and stores**	/* Loads and stores */
45		**Load/store exclusive**	/* Load/store exclusive */
46		STXRB	0x08000000,/* STXRB ARM64Op_stxrb */
47		STLXRB	0x08008000,/* STLXRB ARM64Op_stlxrb */
48		LDXRB	0x08400000,/* LDXRB ARM64Op_ldxrb */
49		LDAXRB	0x08408000,/* LDAXRB ARM64Op_ldaxrb */
50		STLRB	0x08808000,/* STLRB ARM64Op_stlrb */
51		LDARB	0x08C08000,/* LDARB ARM64Op_ldarb */
52		STXRH	0x48000000,/* STXRH ARM64Op_stxrh */
53		STLXRH	0x48008000,/* STLXRH ARM64Op_stlxrh */
54		LDXRH	0x48400000,/* LDXRH ARM64Op_ldxrh */
55		LDAXRH	0x48408000,/* LDAXRH ARM64Op_ldaxrh */
56		STLRH	0x48808000,/* STLRH ARM64Op_stlrh */
57		LDARH	0x48C08000,/* LDARH ARM64Op_ldarh */
58		STXR	0x88000000,/* STXR ARM64Op_stxr_32_bit */
59		STLXR	0x88008000,/* STLXR ARM64Op_stlxr_32_bit */
60		STXP	0x88200000,/* STXP ARM64Op_stxp_32_bit */
61		STLXP	0x88208000,/* STLXP ARM64Op_stlxp_32_bit */
62		LDXR	0x88400000,/* LDXR ARM64Op_ldxr_32_bit */
63		LDAXR	0x88408000,/* LDAXR ARM64Op_ldaxr_32_bit */
64		LDXP	0x88600000,/* LDXP ARM64Op_ldxp_32_bit */
65		LDAXP	0x88608000,/* LDAXP ARM64Op_ldaxp_32_bit */
66		STLR	0x88808000,/* STLR ARM64Op_stlr_32_bit */
67		LDAR	0x88C08000,/* LDAR ARM64Op_ldar_32_bit */
68		STXR	0xC8000000,/* STXR ARM64Op_stxr_64_bit */
69		STLXR	0xC8008000,/* STLXR ARM64Op_stlxr_64_bit */
70		STXP	0xC8200000,/* STXP ARM64Op_stxp_64_bit */
71		STLXP	0xC8208000,/* STLXP ARM64Op_stlxp_64_bit */
72		LDXR	0xC8400000,/* LDXR ARM64Op_ldxr_64_bit */
73		LDAXR	0xC8408000,/* LDAXR ARM64Op_ldaxr_64_bit */
74		LDXP	0xC8600000,/* LDXP ARM64Op_ldxp_64_bit */
75		LDAXP	0xC8608000,/* LDAXP ARM64Op_ldaxp_64_bit */
76		STLR	0xC8808000,/* STLR ARM64Op_stlr_64_bit */
77		LDAR	0xC8C08000,/* LDAR ARM64Op_ldar_64_bit */

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
78		**Load register (literal)**	/* Load register (literal) */		
79		LDR	0x18000000,/* LDR	ARM64Op_ldr_literal_32_bit */	
80		LDR	0x1C000000,/* LDR	ARM64Op_ldr_literal_SIMD_FP_32_bit */	
81		LDR	0x58000000,/* LDR	ARM64Op_ldr_literal_64_bit */	
82		LDR	0x5C000000,/* LDR	ARM64Op_ldr_literal_SIMD_FP_64_bit */	
83		LDRSW	0x98000000,/* LDRSW	ARM64Op_ldrsw_literal */	
84		LDR	0x9C000000,/* LDR	ARM64Op_ldr_literal_SIMD_FP_128_bit */	
85		PRFM	0xD8000000,/* PRFM	ARM64Op_prfm_literal */	
86		**Load/store no-allocate pa**	/* Load/store no-allocate pair (offset) */		
87		STNP	0x28000000,/* STNP	ARM64Op_stnp_32_bit */	
88		LDNP	0x28400000,/* LDNP	ARM64Op_ldnp_32_bit */	
89		STNP	0x2C000000,/* STNP	ARM64Op_stnp_SIMD_FP_32_bit */	
90		LDNP	0x2C400000,/* LDNP	ARM64Op_ldnp_SIMD_FP_32_bit */	
91		STNP	0x6C000000,/* STNP	ARM64Op_stnp_SIMD_FP_64_bit */	
92		LDNP	0x6C400000,/* LDNP	ARM64Op_ldnp_SIMD_FP_64_bit */	
93		STNP	0xA8000000,/* STNP	ARM64Op_stnp_64_bit */	
94		LDNP	0xA8400000,/* LDNP	ARM64Op_ldnp_64_bit */	
95		STNP	0xAC000000,/* STNP	ARM64Op_stnp_SIMD_FP_128_bit */	
96		LDNP	0xAC400000,/* LDNP	ARM64Op_ldnp_SIMD_FP_128_bit */	
97		**Load/store register pair (p**	/* Load/store register pair (post-indexed) */		
98		STP	0x28800000,/* STP	ARM64Op_stp_1_32_bit */	
99		LDP	0x28C00000,/* LDP	ARM64Op_ldp_1_32_bit */	
100		STP	0x2C800000,/* STP	ARM64Op_stp_SIMD_FP_1_32_bit */	
101		LDP	0x2CC00000,/* LDP	ARM64Op_ldp_SIMD_FP_1_32_bit */	
102		LDPSW	0x68C00000,/* LDPSW	ARM64Op_ldpsw_Post_index */	
103		STP	0x6C800000,/* STP	ARM64Op_stp_SIMD_FP_1_64_bit */	
104		LDP	0x6CC00000,/* LDP	ARM64Op_ldp_SIMD_FP_1_64_bit */	
105		STP	0xA8800000,/* STP	ARM64Op_stp_1_64_bit */	
106		LDP	0xA8C00000,/* LDP	ARM64Op_ldp_1_64_bit */	
107		STP	0xAC800000,/* STP	ARM64Op_stp_SIMD_FP_1_128_bit */	
108		LDP	0xACC00000,/* LDP	ARM64Op_ldp_SIMD_FP_1_128_bit */	
109		**Load/store register pair (o**	/* Load/store register pair (offset) */		
110		STP	0x29000000,/* STP	ARM64Op_stp_2_32_bit */	
111		LDP	0x29400000,/* LDP	ARM64Op_ldp_2_32_bit */	
112		STP	0x2D000000,/* STP	ARM64Op_stp_SIMD_FP_2_32_bit */	

1	in_use	Opcode	//BINARY Opcode Opcodecomments
113		LDP	0x2D400000,/* LDP ARM64Op_ldp_SIMD_FP_2_32_bit */
114		LDPSW	0x69400000,/* LDPSW ARM64Op_ldpsw_Signed_offset */
115		STP	0x6D000000,/* STP ARM64Op_stp_SIMD_FP_2_64_bit */
116		LDP	0x6D400000,/* LDP ARM64Op_ldp_SIMD_FP_2_64_bit */
117		STP	0xA9000000,/* STP ARM64Op_stp_2_64_bit */
118		LDP	0xA9400000,/* LDP ARM64Op_ldp_2_64_bit */
119		STP	0xAD000000,/* STP ARM64Op_stp_SIMD_FP_2_128_bit */
120		LDP	0xAD400000,/* LDP ARM64Op_ldp_SIMD_FP_2_128_bit */
121		**Load/store register pair (p**	/* Load/store register pair (pre-indexed) */
122		STP	0x29800000,/* STP ARM64Op_stp_3_32_bit */
123		LDP	0x29C00000,/* LDP ARM64Op_ldp_3_32_bit */
124		STP	0x2D800000,/* STP ARM64Op_stp_SIMD_FP_3_32_bit */
125		LDP	0x2DC00000,/* LDP ARM64Op_ldp_SIMD_FP_3_32_bit */
126		LDPSW	0x69C00000,/* LDPSW ARM64Op_ldpsw_Pre_index */
127		STP	0x6D800000,/* STP ARM64Op_stp_SIMD_FP_3_64_bit */
128		LDP	0x6DC00000,/* LDP ARM64Op_ldp_SIMD_FP_3_64_bit */
129		STP	0xA9800000,/* STP ARM64Op_stp_3_64_bit */
130		LDP	0xA9C00000,/* LDP ARM64Op_ldp_3_64_bit */
131		STP	0xAD800000,/* STP ARM64Op_stp_SIMD_FP_3_128_bit */
132		LDP	0xADC00000,/* LDP ARM64Op_ldp_SIMD_FP_3_128_bit */
133		**Load/store register (unsc**	/* Load/store register (unscaled immediate) */
134		STURB	0x38000000,/* STURB ARM64Op_sturb */
135		LDURB	0x38400000,/* LDURB ARM64Op_ldurb */
136		LDURSB	0x38800000,/* LDURSB ARM64Op_ldursb_64_bit */
137		LDURSB	0x38C00000,/* LDURSB ARM64Op_ldursb_32_bit */
138		STUR	0x3C000000,/* STUR ARM64Op_stur_SIMD_FP_8_bit */
139		LDUR	0x3C400000,/* LDUR ARM64Op_ldur_SIMD_FP_8_bit */
140		STUR	0x3C800000,/* STUR ARM64Op_stur_SIMD_FP_128_bit */
141		LDUR	0x3CC00000,/* LDUR ARM64Op_ldur_SIMD_FP_128_bit */
142		STURH	0x78000000,/* STURH ARM64Op_sturh */
143		LDURH	0x78400000,/* LDURH ARM64Op_ldurh */
144		LDURSH	0x78800000,/* LDURSH ARM64Op_ldursh_64_bit */
145		LDURSH	0x78C00000,/* LDURSH ARM64Op_ldursh_32_bit */
146		STUR	0x7C000000,/* STUR ARM64Op_stur_SIMD_FP_16_bit */
147		LDUR	0x7C400000,/* LDUR ARM64Op_ldur_SIMD_FP_16_bit */
148		STUR	0xB8000000,/* STUR ARM64Op_stur_32_bit */
149		LDUR	0xB8400000,/* LDUR ARM64Op_ldur_32_bit */
150		LDURSW	0xB8800000,/* LDURSW ARM64Op_ldursw */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
151		STUR	0xBC000000,/* STUR ARM64Op_stur_SIMD_FP_32_bit */
152		LDUR	0xBC400000,/* LDUR ARM64Op_ldur_SIMD_FP_32_bit */
153		STUR	0xF8000000,/* STUR ARM64Op_stur_64_bit */
154		LDUR	0xF8400000,/* LDUR ARM64Op_ldur_64_bit */
155		PRFUM	0xF8800000,/* PRFUM ARM64Op_prfum */
156		STUR	0xFC000000,/* STUR ARM64Op_stur_SIMD_FP_64_bit */
157		LDUR	0xFC400000,/* LDUR ARM64Op_ldur_SIMD_FP_64_bit */
158		**Load/store register (imme**	/* Load/store register (immediate post-indexed) */
159		STRB	0x38000400,/* STRB ARM64Op_strb_immediate_Post_index */
160		LDRB	0x38400400,/* LDRB ARM64Op_ldrb_immediate_Post_index */
161		LDRSB	0x38800400,/* LDRSB ARM64Op_ldrsb_immediate_1_64_bit */
162		LDRSB	0x38C00400,/* LDRSB ARM64Op_ldrsb_immediate_1_32_bit */
163		STR	0x3C000400,/* STR ARM64Op_str_immediate_SIMD_FP_1_8_bit */
164		LDR	0x3C400400,/* LDR ARM64Op_ldr_immediate_SIMD_FP_1_8_bit */
165		STR	0x3C800400,/* STR ARM64Op_str_immediate_SIMD_FP_1_128_bit */
166		LDR	0x3CC00400,/* LDR ARM64Op_ldr_immediate_SIMD_FP_1_128_bit */
167		STRH	0x78000400,/* STRH ARM64Op_strh_immediate_Post_index */
168		LDRH	0x78400400,/* LDRH ARM64Op_ldrh_immediate_Post_index */
169		LDRSH	0x78800400,/* LDRSH ARM64Op_ldrsh_immediate_1_64_bit */
170		LDRSH	0x78C00400,/* LDRSH ARM64Op_ldrsh_immediate_1_32_bit */
171		STR	0x7C000400,/* STR ARM64Op_str_immediate_SIMD_FP_1_16_bit */
172		LDR	0x7C400400,/* LDR ARM64Op_ldr_immediate_SIMD_FP_1_16_bit */
173		STR	0xB8000400,/* STR ARM64Op_str_immediate_1_32_bit */
174		LDR	0xB8400400,/* LDR ARM64Op_ldr_immediate_1_32_bit */
175		LDRSW	0xB8800400,/* LDRSW ARM64Op_ldrsw_immediate_Post_index */
176		STR	0xBC000400,/* STR ARM64Op_str_immediate_SIMD_FP_1_32_bit */
177		LDR	0xBC400400,/* LDR ARM64Op_ldr_immediate_SIMD_FP_1_32_bit */
178		STR	0xF8000400,/* STR ARM64Op_str_immediate_1_64_bit */
179		LDR	0xF8400400,/* LDR ARM64Op_ldr_immediate_1_64_bit */
180		STR	0xFC000400,/* STR ARM64Op_str_immediate_SIMD_FP_1_64_bit */
181		LDR	0xFC400400,/* LDR ARM64Op_ldr_immediate_SIMD_FP_1_64_bit */
182		**Load/store register (unpri**	/* Load/store register (unprivileged) */
183		STTRB	0x38000800,/* STTRB ARM64Op_sttrb */
184		LDTRB	0x38400800,/* LDTRB ARM64Op_ldtrb */
185		LDTRSB	0x38800800,/* LDTRSB ARM64Op_ldtrsb_64_bit */
186		LDTRSB	0x38C00800,/* LDTRSB ARM64Op_ldtrsb_32_bit */
187		STTRH	0x78000800,/* STTRH ARM64Op_sttrh */
188		LDTRH	0x78400800,/* LDTRH ARM64Op_ldtrh */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
189		LDTRSH	0x78800800,/* LDTRSH ARM64Op_ldtrsh_64_bit */
190		LDTRSH	0x78C00800,/* LDTRSH ARM64Op_ldtrsh_32_bit */
191		STTR	0xB8000800,/* STTR ARM64Op_sttr_32_bit */
192		LDTR	0xB8400800,/* LDTR ARM64Op_ldtr_32_bit */
193		LDTRSW	0xB8800800,/* LDTRSW ARM64Op_ldtrsw */
194		STTR	0xF8000800,/* STTR ARM64Op_sttr_64_bit */
195		LDTR	0xF8400800,/* LDTR ARM64Op_ldtr_64_bit */
196		**Load/store register (imme**	/* Load/store register (immediate pre-indexed) */
197		STRB	0x38000C00,/* STRB ARM64Op_strb_immediate_Pre_index */
198		LDRB	0x38400C00,/* LDRB ARM64Op_ldrb_immediate_Pre_index */
199		LDRSB	0x38800C00,/* LDRSB ARM64Op_ldrsb_immediate_2_64_bit */
200		LDRSB	0x38C00C00,/* LDRSB ARM64Op_ldrsb_immediate_2_32_bit */
201		STR	0x3C000C00,/* STR ARM64Op_str_immediate_SIMD_FP_2_8_bit */
202		LDR	0x3C400C00,/* LDR ARM64Op_ldr_immediate_SIMD_FP_2_8_bit */
203		STR	0x3C800C00,/* STR ARM64Op_str_immediate_SIMD_FP_2_128_bit */
204		LDR	0x3CC00C00,/* LDR ARM64Op_ldr_immediate_SIMD_FP_2_128_bit */
205		STRH	0x78000C00,/* STRH ARM64Op_strh_immediate_Pre_index */
206		LDRH	0x78400C00,/* LDRH ARM64Op_ldrh_immediate_Pre_index */
207		LDRSH	0x78800C00,/* LDRSH ARM64Op_ldrsh_immediate_2_64_bit */
208		LDRSH	0x78C00C00,/* LDRSH ARM64Op_ldrsh_immediate_2_32_bit */
209		STR	0x7C000C00,/* STR ARM64Op_str_immediate_SIMD_FP_2_16_bit */
210		LDR	0x7C400C00,/* LDR ARM64Op_ldr_immediate_SIMD_FP_2_16_bit */
211		STR	0xB8000C00,/* STR ARM64Op_str_immediate_2_32_bit */
212		LDR	0xB8400C00,/* LDR ARM64Op_ldr_immediate_2_32_bit */
213		LDRSW	0xB8800C00,/* LDRSW ARM64Op_ldrsw_immediate_Pre_index */
214		STR	0xBC000C00,/* STR ARM64Op_str_immediate_SIMD_FP_2_32_bit */
215		LDR	0xBC400C00,/* LDR ARM64Op_ldr_immediate_SIMD_FP_2_32_bit */
216		STR	0xF8000C00,/* STR ARM64Op_str_immediate_2_64_bit */
217		LDR	0xF8400C00,/* LDR ARM64Op_ldr_immediate_2_64_bit */
218		STR	0xFC000C00,/* STR ARM64Op_str_immediate_SIMD_FP_2_64_bit */
219		LDR	0xFC400C00,/* LDR ARM64Op_ldr_immediate_SIMD_FP_2_64_bit */
220		**Load/store register (regis**	/* Load/store register (register offset) */
221		STRB	0x38200800,/* STRB ARM64Op_strb_register */
222		LDRB	0x38600800,/* LDRB ARM64Op_ldrb_register */
223		LDRSB	0x38A00800,/* LDRSB ARM64Op_ldrsb_register_64_bit */
224		LDRSB	0x38E00800,/* LDRSB ARM64Op_ldrsb_register_32_bit */
225		STR	0x3C200800,/* STR ARM64Op_str_register_SIMD_FP_8_bit */
226		LDR	0x3C600800,/* LDR ARM64Op_ldr_register_SIMD_FP_8_bit */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
227		STR	0x3CA00800,/* STR ARM64Op_str_register_SIMD_FP_128_bit */
228		LDR	0x3CE00800,/* LDR ARM64Op_ldr_register_SIMD_FP_128_bit */
229		STRH	0x78200800,/* STRH ARM64Op_strh_register */
230		LDRH	0x78600800,/* LDRH ARM64Op_ldrh_register */
231		LDRSH	0x78A00800,/* LDRSH ARM64Op_ldrsh_register_64_bit */
232		LDRSH	0x78E00800,/* LDRSH ARM64Op_ldrsh_register_32_bit */
233		STR	0x7C200800,/* STR ARM64Op_str_register_SIMD_FP_16_bit */
234		LDR	0x7C600800,/* LDR ARM64Op_ldr_register_SIMD_FP_16_bit */
235		STR	0xB8200800,/* STR ARM64Op_str_register_32_bit */
236		LDR	0xB8600800,/* LDR ARM64Op_ldr_register_32_bit */
237		LDRSW	0xB8A00800,/* LDRSW ARM64Op_ldrsw_register */
238		STR	0xBC200800,/* STR ARM64Op_str_register_SIMD_FP_32_bit */
239		LDR	0xBC600800,/* LDR ARM64Op_ldr_register_SIMD_FP_32_bit */
240		STR	0xF8200800,/* STR ARM64Op_str_register_64_bit */
241		LDR	0xF8600800,/* LDR ARM64Op_ldr_register_64_bit */
242		PRFM	0xF8A00800,/* PRFM ARM64Op_prfm_register */
243		STR	0xFC200800,/* STR ARM64Op_str_register_SIMD_FP_64_bit */
244		LDR	0xFC600800,/* LDR ARM64Op_ldr_register_SIMD_FP_64_bit */
245		**Load/store register (unsig**	/* Load/store register (unsigned immediate) */
246		STRB	0x39000000,/* STRB ARM64Op_strb_immediate_Unsigned_offset */
247		LDRB	0x39400000,/* LDRB ARM64Op_ldrb_immediate_Unsigned_offset */
248		LDRSB	0x39800000,/* LDRSB ARM64Op_ldrsb_immediate_3_64_bit */
249		LDRSB	0x39C00000,/* LDRSB ARM64Op_ldrsb_immediate_3_32_bit */
250		STR	0x3D000000,/* STR ARM64Op_str_immediate_SIMD_FP_8_bit */
251		LDR	0x3D400000,/* LDR ARM64Op_ldr_immediate_SIMD_FP_8_bit */
252		STR	0x3D800000,/* STR ARM64Op_str_immediate_SIMD_FP_128_bit */
253		LDR	0x3DC00000,/* LDR ARM64Op_ldr_immediate_SIMD_FP_128_bit */
254		STRH	0x79000000,/* STRH ARM64Op_strh_immediate_Unsigned_offset */
255		LDRH	0x79400000,/* LDRH ARM64Op_ldrh_immediate_Unsigned_offset */
256		LDRSH	0x79800000,/* LDRSH ARM64Op_ldrsh_immediate_3_64_bit */
257		LDRSH	0x79C00000,/* LDRSH ARM64Op_ldrsh_immediate_3_32_bit */
258		STR	0x7D000000,/* STR ARM64Op_str_immediate_SIMD_FP_16_bit */
259		LDR	0x7D400000,/* LDR ARM64Op_ldr_immediate_SIMD_FP_16_bit */
260		STR	0xB9000000,/* STR ARM64Op_str_immediate_3_32_bit */
261		LDR	0xB9400000,/* LDR ARM64Op_ldr_immediate_3_32_bit */
262		LDRSW	0xB9800000,/* LDRSW ARM64Op_ldrsw_immediate_Unsigned_offset */
263		STR	0xBD000000,/* STR ARM64Op_str_immediate_SIMD_FP_32_bit */
264		LDR	0xBD400000,/* LDR ARM64Op_ldr_immediate_SIMD_FP_32_bit */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
265		STR	0xF9000000,/* STR ARM64Op_str_immediate_3_64_bit */
266		LDR	0xF9400000,/* LDR ARM64Op_ldr_immediate_3_64_bit */
267		PRFM	0xF9800000,/* PRFM ARM64Op_prfm_immediate */
268		STR	0xFD000000,/* STR ARM64Op_str_immediate_SIMD_FP_64_bit */
269		LDR	0xFD400000,/* LDR ARM64Op_ldr_immediate_SIMD_FP_64_bit */
270		**Data processing – Imme**	/* Data processing – Immediate */
271		**PC-rel. addressing**	/* PC-rel. addressing */
272		ADR	0x10000000,/* ADR ARM64Op_adr */
273		ADRP	0x90000000,/* ADRP ARM64Op_adrp */
274		**Add/subtract (immediate)**	/* Add/subtract (immediate) */
275		ADD	0x11000000,/* ADD ARM64Op_add_immediate_32_bit */
276		ADDS	0x31000000,/* ADDS ARM64Op_adds_immediate_32_bit */
277		SUB	0x51000000,/* SUB ARM64Op_sub_immediate_32_bit */
278		SUBS	0x71000000,/* SUBS ARM64Op_subs_immediate_32_bit */
279		ADD	0x91000000,/* ADD ARM64Op_add_immediate_64_bit */
280		ADDS	0xB1000000,/* ADDS ARM64Op_adds_immediate_64_bit */
281		SUB	0xD1000000,/* SUB ARM64Op_sub_immediate_64_bit */
282		SUBS	0xF1000000,/* SUBS ARM64Op_subs_immediate_64_bit */
283		**Logical (immediate)**	/* Logical (immediate) */
284		AND	0x12000000,/* AND ARM64Op_and_immediate_32_bit */
285		ORR	0x32000000,/* ORR ARM64Op_orr_immediate_32_bit */
286		EOR	0x52000000,/* EOR ARM64Op_eor_immediate_32_bit */
287		ANDS	0x72000000,/* ANDS ARM64Op_ands_immediate_32_bit */
288		AND	0x92000000,/* AND ARM64Op_and_immediate_64_bit */
289		ORR	0xB2000000,/* ORR ARM64Op_orr_immediate_64_bit */
290		EOR	0xD2000000,/* EOR ARM64Op_eor_immediate_64_bit */
291		ANDS	0xF2000000,/* ANDS ARM64Op_ands_immediate_64_bit */
292		**Move wide (immediate)**	/* Move wide (immediate) */
293		MOVN	0x12800000,/* MOVN ARM64Op_movn_32_bit */
294		MOVZ	0x52800000,/* MOVZ ARM64Op_movz_32_bit */
295		MOVK	0x72800000,/* MOVK ARM64Op_movk_32_bit */
296		MOVN	0x92800000,/* MOVN ARM64Op_movn_64_bit */
297		MOVZ	0xD2800000,/* MOVZ ARM64Op_movz_64_bit */
298		MOVK	0xF2800000,/* MOVK ARM64Op_movk_64_bit */
299		**Bitfield**	/* Bitfield */
300		SBFM	0x13000000,/* SBFM ARM64Op_sbfm_32_bit */
301		BFM	0x33000000,/* BFM ARM64Op_bfm_32_bit */
302		UBFM	0x53000000,/* UBFM ARM64Op_ubfm_32_bit */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
303		SBFM	0x93400000,/* SBFM ARM64Op_sbfm_64_bit */
304		BFM	0xB3400000,/* BFM ARM64Op_bfm_64_bit */
305		UBFM	0xD3400000,/* UBFM ARM64Op_ubfm_64_bit */
306		**Extract**	/* Extract */
307		EXTR	0x13800000,/* EXTR ARM64Op_extr_32_bit */
308		EXTR	0x93C00000,/* EXTR ARM64Op_extr_64_bit */
309		**Data Processing – regis**	/* Data Processing – register */
310		**Logical (shifted register)**	/* Logical (shifted register) */
311		AND	0x0A000000,/* AND ARM64Op_and_shifted_register_32_bit */
312		BIC	0x0A200000,/* BIC ARM64Op_bic_shifted_register_32_bit */
313		ORR	0x2A000000,/* ORR ARM64Op_orr_shifted_register_32_bit */
314		ORN	0x2A200000,/* ORN ARM64Op_orn_shifted_register_32_bit */
315		EOR	0x4A000000,/* EOR ARM64Op_eor_shifted_register_32_bit */
316		EON	0x4A200000,/* EON ARM64Op_eon_shifted_register_32_bit */
317		ANDS	0x6A000000,/* ANDS ARM64Op_ands_shifted_register_32_bit */
318		BICS	0x6A200000,/* BICS ARM64Op_bics_shifted_register_32_bit */
319		AND	0x8A000000,/* AND ARM64Op_and_shifted_register_64_bit */
320		BIC	0x8A200000,/* BIC ARM64Op_bic_shifted_register_64_bit */
321		ORR	0xAA000000,/* ORR ARM64Op_orr_shifted_register_64_bit */
322		ORN	0xAA200000,/* ORN ARM64Op_orn_shifted_register_64_bit */
323		EOR	0xCA000000,/* EOR ARM64Op_eor_shifted_register_64_bit */
324		EON	0xCA200000,/* EON ARM64Op_eon_shifted_register_64_bit */
325		ANDS	0xEA000000,/* ANDS ARM64Op_ands_shifted_register_64_bit */
326		BICS	0xEA200000,/* BICS ARM64Op_bics_shifted_register_64_bit */
327		**Add/subtract (shifted regi**	/* Add/subtract (shifted register) */
328		ADD	0x0B000000,/* ADD ARM64Op_add_shifted_register_32_bit */
329		ADDS	0x2B000000,/* ADDS ARM64Op_adds_shifted_register_32_bit */
330		SUB	0x4B000000,/* SUB ARM64Op_sub_shifted_register_32_bit */
331		SUBS	0x6B000000,/* SUBS ARM64Op_subs_shifted_register_32_bit */
332		ADD	0x8B000000,/* ADD ARM64Op_add_shifted_register_64_bit */
333		ADDS	0xAB000000,/* ADDS ARM64Op_adds_shifted_register_64_bit */
334		SUB	0xCB000000,/* SUB ARM64Op_sub_shifted_register_64_bit */
335		SUBS	0xEB000000,/* SUBS ARM64Op_subs_shifted_register_64_bit */
336		**Add/subtract (extended re**	/* Add/subtract (extended register) */
337		ADD	0x0B200000,/* ADD ARM64Op_add_extended_register_32_bit */
338		ADDS	0x2B200000,/* ADDS ARM64Op_adds_extended_register_32_bit */
339		SUB	0x4B200000,/* SUB ARM64Op_sub_extended_register_32_bit */
340		SUBS	0x6B200000,/* SUBS ARM64Op_subs_extended_register_32_bit */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
341		ADD	0x8B200000,/* ADD ARM64Op_add_extended_register_64_bit */
342		ADDS	0xAB200000,/* ADDS ARM64Op_adds_extended_register_64_bit */
343		SUB	0xCB200000,/* SUB ARM64Op_sub_extended_register_64_bit */
344		SUBS	0xEB200000,/* SUBS ARM64Op_subs_extended_register_64_bit */
345		**Add/subtract (with carry)**	/* Add/subtract (with carry) */
346		ADC	0x1A000000,/* ADC ARM64Op_adc_32_bit */
347		ADCS	0x3A000000,/* ADCS ARM64Op_adcs_32_bit */
348		SBC	0x5A000000,/* SBC ARM64Op_sbc_32_bit */
349		SBCS	0x7A000000,/* SBCS ARM64Op_sbcs_32_bit */
350		ADC	0x9A000000,/* ADC ARM64Op_adc_64_bit */
351		ADCS	0xBA000000,/* ADCS ARM64Op_adcs_64_bit */
352		SBC	0xDA000000,/* SBC ARM64Op_sbc_64_bit */
353		SBCS	0xFA000000,/* SBCS ARM64Op_sbcs_64_bit */
354		**Conditional compare (reg**	/* Conditional compare (register) */
355		CCMN	0x3A400000,/* CCMN ARM64Op_ccmn_register_32_bit */
356		CCMN	0xBA400000,/* CCMN ARM64Op_ccmn_register_64_bit */
357		CCMP	0x7A400000,/* CCMP ARM64Op_ccmp_register_32_bit */
358		CCMP	0xFA400000,/* CCMP ARM64Op_ccmp_register_64_bit */
359		**Conditional compare (imr**	/* Conditional compare (immediate) */
360		CCMN	0x3A400800,/* CCMN ARM64Op_ccmn_immediate_32_bit */
361		CCMN	0xBA400800,/* CCMN ARM64Op_ccmn_immediate_64_bit */
362		CCMP	0x7A400800,/* CCMP ARM64Op_ccmp_immediate_32_bit */
363		CCMP	0xFA400800,/* CCMP ARM64Op_ccmp_immediate_64_bit */
364		**Conditional select**	/* Conditional select */
365		CSEL	0x1A800000,/* CSEL ARM64Op_csel_32_bit */
366		CSINC	0x1A800400,/* CSINC ARM64Op_csinc_32_bit */
367		CSINV	0x5A800000,/* CSINV ARM64Op_csinv_32_bit */
368		CSNEG	0x5A800400,/* CSNEG ARM64Op_csneg_32_bit */
369		CSEL	0x9A800000,/* CSEL ARM64Op_csel_64_bit */
370		CSINC	0x9A800400,/* CSINC ARM64Op_csinc_64_bit */
371		CSINV	0xDA800000,/* CSINV ARM64Op_csinv_64_bit */
372		CSNEG	0xDA800400,/* CSNEG ARM64Op_csneg_64_bit */
373		**Data-processing (3 sourc**	/* Data-processing (3 source) */
374		MADD	0x1B000000,/* MADD ARM64Op_madd_32_bit */
375		MADD	0x9B000000,/* MADD ARM64Op_madd_64_bit */
376		SMADDL	0x9B200000,/* SMADDL ARM64Op_smaddl */
377		UMADDL	0x9BA00000,/* UMADDL ARM64Op_umaddl */
378		MSUB	0x1B008000,/* MSUB ARM64Op_msub_32_bit */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
379		MSUB	0x9B008000,/* MSUB ARM64Op_msub_64_bit */
380		SMSUBL	0x9B208000,/* SMSUBL ARM64Op_smsubl */
381		UMSUBL	0x9BA08000,/* UMSUBL ARM64Op_umsubl */
382		SMULH	0x9B400000,/* SMULH ARM64Op_smulh */
383		UMULH	0x9BC00000,/* UMULH ARM64Op_umulh */
384		**Data-processing (2 sourc**	/* Data-processing (2 source) */
385		CRC32X	0x9AC04C00,/* CRC32X ARM64Op_crc32x */
386		CRC32CX	0x9AC05C00,/* CRC32CX ARM64Op_crc32cx */
387		CRC32B	0x1AC04000,/* CRC32B ARM64Op_crc32b */
388		CRC32CB	0x1AC05000,/* CRC32CB ARM64Op_crc32cb */
389		CRC32H	0x1AC04400,/* CRC32H ARM64Op_crc32h */
390		CRC32CH	0x1AC05400,/* CRC32CH ARM64Op_crc32ch */
391		CRC32W	0x1AC04800,/* CRC32W ARM64Op_crc32w */
392		CRC32CW	0x1AC05800,/* CRC32CW ARM64Op_crc32cw */
393		UDIV	0x1AC00800,/* UDIV ARM64Op_udiv_32_bit */
394		UDIV	0x9AC00800,/* UDIV ARM64Op_udiv_64_bit */
395		SDIV	0x1AC00C00,/* SDIV ARM64Op_sdiv_32_bit */
396		SDIV	0x9AC00C00,/* SDIV ARM64Op_sdiv_64_bit */
397		LSLV	0x1AC02000,/* LSLV ARM64Op_lslv_32_bit */
398		LSLV	0x9AC02000,/* LSLV ARM64Op_lslv_64_bit */
399		LSRV	0x1AC02400,/* LSRV ARM64Op_lsrv_32_bit */
400		LSRV	0x9AC02400,/* LSRV ARM64Op_lsrv_64_bit */
401		ASRV	0x1AC02800,/* ASRV ARM64Op_asrv_32_bit */
402		ASRV	0x9AC02800,/* ASRV ARM64Op_asrv_64_bit */
403		RORV	0x1AC02C00,/* RORV ARM64Op_rorv_32_bit */
404		RORV	0x9AC02C00,/* RORV ARM64Op_rorv_64_bit */
405		**Data-processing (1 sourc**	/* Data-processing (1 source) */
406		RBIT	0x5AC00000,/* RBIT ARM64Op_rbit_32_bit */
407		RBIT	0xDAC00000,/* RBIT ARM64Op_rbit_64_bit */
408		CLZ	0x5AC01000,/* CLZ ARM64Op_clz_32_bit */
409		CLZ	0xDAC01000,/* CLZ ARM64Op_clz_64_bit */
410		CLS	0x5AC01400,/* CLS ARM64Op_cls_32_bit */
411		CLS	0xDAC01400,/* CLS ARM64Op_cls_64_bit */
412		REV	0x5AC00800,/* REV ARM64Op_rev_32_bit */
413		REV	0xDAC00C00,/* REV ARM64Op_rev_64_bit */
414		REV16	0xDAC00400,/* REV16 ARM64Op_rev16_64_bit */
415		REV16	0x5AC00400,/* REV16 ARM64Op_rev16_32_bit */
416		REV32	0xDAC00800,/* REV32 ARM64Op_rev32 */

```
1   in_use      Opcode                  //BINARY    Opcode    Opcodecomments
417 //      Data Processing – SIMD      /* Data Processing – SIMD and floating point */
418 //          Floating-point<->fixed-po   /* Floating-point<->fixed-point conversions */
419 //              SCVTF                   //0x1E020000,/* SCVTF     ARM64Op_scvtf_scalar_fixed_point_32_bit_to_single_
420 //              UCVTF                   //0x1E030000,/* UCVTF     ARM64Op_ucvtf_scalar_fixed_point_32_bit_to_single_
421 //              FCVTZS                  //0x1ED80000,/* FCVTZS    ARM64Op_fcvtzs_scalar_fixed_point_Single_precisio
422 //              FCVTZU                  //0x1ED90000,/* FCVTZU    ARM64Op_fcvtzu_scalar_fixed_point_Single_precisio
423 //              SCVTF                   //0x1E020000,/* SCVTF     ARM64Op_scvtf_scalar_fixed_point_32_bit_to_double
424 //              UCVTF                   //0x1E030000,/* UCVTF     ARM64Op_ucvtf_scalar_fixed_point_32_bit_to_double
425 //              FCVTZS                  //0x1ED80000,/* FCVTZS    ARM64Op_fcvtzs_scalar_fixed_point_Double_precisi
426 //              FCVTZU                  //0x1ED90000,/* FCVTZU    ARM64Op_fcvtzu_scalar_fixed_point_Double_precis
427 //              SCVTF                   //0x9E020000,/* SCVTF     ARM64Op_scvtf_scalar_fixed_point_64_bit_to_single_
428 //              UCVTF                   //0x9E030000,/* UCVTF     ARM64Op_ucvtf_scalar_fixed_point_64_bit_to_single_
429 //              FCVTZS                  //0x9ED80000,/* FCVTZS    ARM64Op_fcvtzs_scalar_fixed_point_Single_precisio
430 //              FCVTZU                  //0x9ED90000,/* FCVTZU    ARM64Op_fcvtzu_scalar_fixed_point_Single_precisio
431 //              SCVTF                   //0x9E020000,/* SCVTF     ARM64Op_scvtf_scalar_fixed_point_64_bit_to_double
432 //              UCVTF                   //0x9E030000,/* UCVTF     ARM64Op_ucvtf_scalar_fixed_point_64_bit_to_double
433 //              FCVTZS                  //0x9ED80000,/* FCVTZS    ARM64Op_fcvtzs_scalar_fixed_point_Double_precisi
434 //              FCVTZU                  //0x9ED90000,/* FCVTZU    ARM64Op_fcvtzu_scalar_fixed_point_Double_precis
435 //          Floating-point conditional  /* Floating-point conditional compare */
436 //              FCCMP                   //0x1E200400,/* FCCMP     ARM64Op_fccmp_Single_precision */
437 //              FCCMPE                  //0x1E200410,/* FCCMPE    ARM64Op_fccmpe_Single_precision */
438 //              FCCMP                   //0x1E600400,/* FCCMP     ARM64Op_fccmp_Double_precision */
439 //              FCCMPE                  //0x1E600410,/* FCCMPE    ARM64Op_fccmpe_Double_precision */
440 //          Floating-point data-proce   /* Floating-point data-processing (2 source) */
441 //              FMUL                    //0x1E200800,/* FMUL      ARM64Op_fmul_scalar_Single_precision */
442 //              FDIV                    //0x1E201800,/* FDIV      ARM64Op_fdiv_scalar_Single_precision */
443 //              FADD                    //0x1E202800,/* FADD      ARM64Op_fadd_scalar_Single_precision */
444 //              FSUB                    //0x1E203800,/* FSUB      ARM64Op_fsub_scalar_Single_precision */
445 //              FMAX                    //0x1E204800,/* FMAX      ARM64Op_fmax_scalar_Single_precision */
446 //              FMIN                    //0x1E205800,/* FMIN      ARM64Op_fmin_scalar_Single_precision */
447 //              FMAXNM                  //0x1E206800,/* FMAXNM    ARM64Op_fmaxnm_scalar_Single_precision */
448 //              FMINNM                  //0x1E207800,/* FMINNM    ARM64Op_fminnm_scalar_Single_precision */
449 //              FNMUL                   //0x1E208800,/* FNMUL     ARM64Op_fnmul_Single_precision */
450 //              FMUL                    //0x1E600800,/* FMUL      ARM64Op_fmul_scalar_Double_precision */
```

1	in_use	Opcode	//BINARY Opcode Opcodecomments
451	//	FDIV	//0x1E601800,/* FDIV ARM64Op_fdiv_scalar_Double_precision */
452	//	FADD	//0x1E602800,/* FADD ARM64Op_fadd_scalar_Double_precision */
453	//	FSUB	//0x1E603800,/* FSUB ARM64Op_fsub_scalar_Double_precision */
454	//	FMAX	//0x1E604800,/* FMAX ARM64Op_fmax_scalar_Double_precision */
455	//	FMIN	//0x1E605800,/* FMIN ARM64Op_fmin_scalar_Double_precision */
456	//	FMAXNM	//0x1E606800,/* FMAXNM ARM64Op_fmaxnm_scalar_Double_precision */
457	//	FMINNM	//0x1E607800,/* FMINNM ARM64Op_fminnm_scalar_Double_precision */
458	//	FNMUL	//0x1E608800,/* FNMUL ARM64Op_fnmul_Double_precision */
459	//	**Floating-point conditional**	/* Floating-point conditional select */
460	//	FCSEL	//0x1E200C00,/* FCSEL ARM64Op_fcsel_Single_precision */
461	//	FCSEL	//0x1E600C00,/* FCSEL ARM64Op_fcsel_Double_precision */
462	//	**Floating-point immediate**	/* Floating-point immediate */
463	//	FMOV	//0x1E201000,/* FMOV ARM64Op_fmov_scalar_immediate_Single_precision
464	//	FMOV	//0x1E601000,/* FMOV ARM64Op_fmov_scalar_immediate_Double_precision
465	//	**Floating-point compare**	/* Floating-point compare */
466	//	FCMP	//0x1E202000,/* FCMP ARM64Op_fcmp_Single_precision */
467	//	FCMP	//0x1E202008,/* FCMP ARM64Op_fcmp_Single_precision_zero */
468	//	FCMPE	//0x1E202010,/* FCMPE ARM64Op_fcmpe_Single_precision */
469	//	FCMPE	//0x1E202018,/* FCMPE ARM64Op_fcmpe_Single_precision_zero */
470	//	FCMP	//0x1E602000,/* FCMP ARM64Op_fcmp_Double_precision */
471	//	FCMP	//0x1E602008,/* FCMP ARM64Op_fcmp_Double_precision_zero */
472	//	FCMPE	//0x1E602010,/* FCMPE ARM64Op_fcmpe_Double_precision */
473	//	FCMPE	//0x1E602018,/* FCMPE ARM64Op_fcmpe_Double_precision_zero */
474	//	**Floating-point data-proce**	/* Floating-point data-processing (1 source) */
475	//	FMOV	//0x1E204000,/* FMOV ARM64Op_fmov_register_Single_precision */
476	//	FABS	//0x1E20C000,/* FABS ARM64Op_fabs_scalar_Single_precision */
477	//	FNEG	//0x1E214000,/* FNEG ARM64Op_fneg_scalar_Single_precision */
478	//	FSQRT	//0x1E21C000,/* FSQRT ARM64Op_fsqrt_scalar_Single_precision */
479	//	FCVT	//0x1E22C000,/* FCVT ARM64Op_fcvt_Single_precision_to_double_precision
480	//	FCVT	//0x1E23C000,/* FCVT ARM64Op_fcvt_Single_precision_to_half_precision */
481	//	FRINTN	//0x1E244000,/* FRINTN ARM64Op_frintn_scalar_Single_precision */
482	//	FRINTP	//0x1E24C000,/* FRINTP ARM64Op_frintp_scalar_Single_precision */
483	//	FRINTM	//0x1E254000,/* FRINTM ARM64Op_frintm_scalar_Single_precision */
484	//	FRINTZ	//0x1E25C000,/* FRINTZ ARM64Op_frintz_scalar_Single_precision */

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
485	//	FRINTA	//0x1E264000,/*	FRINTA	ARM64Op_frinta_scalar_Single_precision */
486	//	FRINTX	//0x1E274000,/*	FRINTX	ARM64Op_frintx_scalar_Single_precision */
487	//	FRINTI	//0x1E27C000,/*	FRINTI	ARM64Op_frinti_scalar_Single_precision */
488	//	FMOV	//0x1E604000,/*	FMOV	ARM64Op_fmov_register_Double_precision */
489	//	FABS	//0x1E60C000,/*	FABS	ARM64Op_fabs_scalar_Double_precision */
490	//	FNEG	//0x1E614000,/*	FNEG	ARM64Op_fneg_scalar_Double_precision */
491	//	FSQRT	//0x1E61C000,/*	FSQRT	ARM64Op_fsqrt_scalar_Double_precision */
492	//	FCVT	//0x1E62C000,/*	FCVT	ARM64Op_fcvt_Double_precision_to_single_precision
493	//	FCVT	//0x1E63C000,/*	FCVT	ARM64Op_fcvt_Double_precision_to_half_precision */
494	//	FRINTN	//0x1E644000,/*	FRINTN	ARM64Op_frintn_scalar_Double_precision */
495	//	FRINTP	//0x1E64C000,/*	FRINTP	ARM64Op_frintp_scalar_Double_precision */
496	//	FRINTM	//0x1E654000,/*	FRINTM	ARM64Op_frintm_scalar_Double_precision */
497	//	FRINTZ	//0x1E65C000,/*	FRINTZ	ARM64Op_frintz_scalar_Double_precision */
498	//	FRINTA	//0x1E664000,/*	FRINTA	ARM64Op_frinta_scalar_Double_precision */
499	//	FRINTX	//0x1E674000,/*	FRINTX	ARM64Op_frintx_scalar_Double_precision */
500	//	FRINTI	//0x1E67C000,/*	FRINTI	ARM64Op_frinti_scalar_Double_precision */
501	//	FCVT	//0x1EE24000,/*	FCVT	ARM64Op_fcvt_Half_precision_to_single_precision */
502	//	FCVT	//0x1EE2C000,/*	FCVT	ARM64Op_fcvt_Half_precision_to_double_precision */
503	//	**Floating-point<->integer c**	/* Floating-point<->integer conversions */		
504	//	FCVTNS	//0x1E200000,/*	FCVTNS	ARM64Op_fcvtns_scalar_Single_precision_to_32_bit
505	//	FCVTNU	//0x1E210000,/*	FCVTNU	ARM64Op_fcvtnu_scalar_Single_precision_to_32_bi
506	//	SCVTF	//0x1E220000,/*	SCVTF	ARM64Op_scvtf_scalar_integer_32_bit_to_single_pre
507	//	UCVTF	//0x1E230000,/*	UCVTF	ARM64Op_ucvtf_scalar_integer_32_bit_to_single_pre
508	//	FCVTAS	//0x1E240000,/*	FCVTAS	ARM64Op_fcvtas_scalar_Single_precision_to_32_bit
509	//	FCVTAU	//0x1E250000,/*	FCVTAU	ARM64Op_fcvtau_scalar_Single_precision_to_32_bit
510	//	FMOV	//0x1E260000,/*	FMOV	ARM64Op_fmov_general_Single_precision_to_32_bit
511	//	FMOV	//0x1E270000,/*	FMOV	ARM64Op_fmov_general_32_bit_to_single_precision
512	//	FCVTPS	//0x1E280000,/*	FCVTPS	ARM64Op_fcvtps_scalar_Single_precision_to_32_bit
513	//	FCVTPU	//0x1E290000,/*	FCVTPU	ARM64Op_fcvtpu_scalar_Single_precision_to_32_bit
514	//	FCVTMS	//0x1E300000,/*	FCVTMS	ARM64Op_fcvtms_scalar_Single_precision_to_32_b
515	//	FCVTMU	//0x1E310000,/*	FCVTMU	ARM64Op_fcvtmu_scalar_Single_precision_to_32_b
516	//	FCVTZS	//0x1E380000,/*	FCVTZS	ARM64Op_fcvtzs_scalar_integer_Single_precision_to
517	//	FCVTZU	//0x1E390000,/*	FCVTZU	ARM64Op_fcvtzu_scalar_integer_Single_precision_to
518	//	FCVTNS	//0x1E600000,/*	FCVTNS	ARM64Op_fcvtns_scalar_Double_precision_to_32_b

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
519	//	FCVTNU	//0x1E610000,/*	FCVTNU	ARM64Op_fcvtnu_scalar_Double_precision_to_32_b
520	//	SCVTF	//0x1E620000,/*	SCVTF	ARM64Op_scvtf_scalar_integer_32_bit_to_double_pre
521	//	UCVTF	//0x1E630000,/*	UCVTF	ARM64Op_ucvtf_scalar_integer_32_bit_to_double_pr
522	//	FCVTAS	//0x1E640000,/*	FCVTAS	ARM64Op_fcvtas_scalar_Double_precision_to_32_b
523	//	FCVTAU	//0x1E650000,/*	FCVTAU	ARM64Op_fcvtau_scalar_Double_precision_to_32_b
524	//	FCVTPS	//0x1E680000,/*	FCVTPS	ARM64Op_fcvtps_scalar_Double_precision_to_32_b
525	//	FCVTPU	//0x1E690000,/*	FCVTPU	ARM64Op_fcvtpu_scalar_Double_precision_to_32_b
526	//	FCVTMS	//0x1E700000,/*	FCVTMS	ARM64Op_fcvtms_scalar_Double_precision_to_32_l
527	//	FCVTMU	//0x1E710000,/*	FCVTMU	ARM64Op_fcvtmu_scalar_Double_precision_to_32_
528	//	FCVTZS	//0x1E780000,/*	FCVTZS	ARM64Op_fcvtzs_scalar_integer_Double_precision_t
529	//	FCVTZU	//0x1E790000,/*	FCVTZU	ARM64Op_fcvtzu_scalar_integer_Double_precision_
530	//	FCVTNS	//0x9E200000,/*	FCVTNS	ARM64Op_fcvtns_scalar_Single_precision_to_64_bit
531	//	FCVTNU	//0x9E210000,/*	FCVTNU	ARM64Op_fcvtnu_scalar_Single_precision_to_64_bi
532	//	SCVTF	//0x9E220000,/*	SCVTF	ARM64Op_scvtf_scalar_integer_64_bit_to_single_pre
533	//	UCVTF	//0x9E230000,/*	UCVTF	ARM64Op_ucvtf_scalar_integer_64_bit_to_single_pre
534	//	FCVTAS	//0x9E244000,/*	FCVTAS	ARM64Op_fcvtas_scalar_Single_precision_to_64_bit
535	//	FCVTAU	//0x9E250000,/*	FCVTAU	ARM64Op_fcvtau_scalar_Single_precision_to_64_bit
536	//	FCVTPS	//0x9E280000,/*	FCVTPS	ARM64Op_fcvtps_scalar_Single_precision_to_64_bit
537	//	FCVTPU	//0x9E290000,/*	FCVTPU	ARM64Op_fcvtpu_scalar_Single_precision_to_64_bit
538	//	FCVTMS	//0x9E300000,/*	FCVTMS	ARM64Op_fcvtms_scalar_Single_precision_to_64_b
539	//	FCVTMU	//0x9E318000,/*	FCVTMU	ARM64Op_fcvtmu_scalar_Single_precision_to_64_b
540	//	FCVTZS	//0x9E380000,/*	FCVTZS	ARM64Op_fcvtzs_scalar_integer_Single_precision_to
541	//	FCVTZU	//0x9E390000,/*	FCVTZU	ARM64Op_fcvtzu_scalar_integer_Single_precision_to
542	//	FCVTNS	//0x9E200000,/*	FCVTNS	ARM64Op_fcvtns_scalar_Double_precision_to_64_b
543	//	FCVTNU	//0x9E210000,/*	FCVTNU	ARM64Op_fcvtnu_scalar_Double_precision_to_64_b
544	//	SCVTF	//0x9E620000,/*	SCVTF	ARM64Op_scvtf_scalar_integer_64_bit_to_double_pre
545	//	UCVTF	//0x9E630000,/*	UCVTF	ARM64Op_ucvtf_scalar_integer_64_bit_to_double_pr
546	//	FCVTAS	//0x9E640000,/*	FCVTAS	ARM64Op_fcvtas_scalar_Double_precision_to_64_b
547	//	FCVTAU	//0x9E650000,/*	FCVTAU	ARM64Op_fcvtau_scalar_Double_precision_to_64_b
548	//	FMOV	//0x9E660000,/*	FMOV	ARM64Op_fmov_general_Double_precision_to_64_bit
549	//	FMOV	//0x9E670000,/*	FMOV	ARM64Op_fmov_general_64_bit_to_double_precision
550	//	FCVTPS	//0x9E680000,/*	FCVTPS	ARM64Op_fcvtps_scalar_Double_precision_to_64_b
551	//	FCVTPU	//0x9E690000,/*	FCVTPU	ARM64Op_fcvtpu_scalar_Double_precision_to_64_b
552	//	FCVTMS	//0x9E700000,/*	FCVTMS	ARM64Op_fcvtms_scalar_Double_precision_to_64_l

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
553	//	FCVTMU	//0x9E710000,/* FCVTMU	ARM64Op_fcvtmu_scalar_Double_precision_to_64_	
554	//	FCVTZS	//0x9E780000,/* FCVTZS	ARM64Op_fcvtzs_scalar_integer_Double_precision_t	
555	//	FCVTZU	//0x9E790000,/* FCVTZU	ARM64Op_fcvtzu_scalar_integer_Double_precision_	
556	//	FMOV	//0x9EAE0000,/* FMOV	ARM64Op_fmov_general_Top_half_of_128_bit_to_64	
557	//	FMOV	//0x9EAF0000,/* FMOV	ARM64Op_fmov_general_64_bit_to_top_half_of_128_	
558	//	**Floating-point data-proce**	/* Floating-point data-processing (3 source) */		
559	//	FMADD	//0x1F000000,/* FMADD	ARM64Op_fmadd_Single_precision */	
560	//	FMSUB	//0x1F008000,/* FMSUB	ARM64Op_fmsub_Single_precision */	
561	//	FNMADD	//0x1F200000,/* FNMADD	ARM64Op_fnmadd_Single_precision */	
562	//	FNMSUB	//0x1F208000,/* FNMSUB	ARM64Op_fnmsub_Single_precision */	
563	//	FMADD	//0x1F400000,/* FMADD	ARM64Op_fmadd_Double_precision */	
564	//	FMSUB	//0x1F408000,/* FMSUB	ARM64Op_fmsub_Double_precision */	
565	//	FNMADD	//0x1F600000,/* FNMADD	ARM64Op_fnmadd_Double_precision */	
566	//	FNMSUB	//0x1F608000,/* FNMSUB	ARM64Op_fnmsub_Double_precision */	
567	//	**AdvSIMD scalar three san**	/* AdvSIMD scalar three same */		
568	//	SQADD	//0x5E200C00,/* SQADD	ARM64Op_sqadd_Scalar */	
569	//	SQSUB	//0x5E202C00,/* SQSUB	ARM64Op_sqsub_Scalar */	
570	//	CMGT	//0x5E203400,/* CMGT	ARM64Op_cmgt_register_Scalar */	
571	//	CMGE	//0x5E203C00,/* CMGE	ARM64Op_cmge_register_Scalar */	
572	//	SSHL	//0x5E204400,/* SSHL	ARM64Op_sshl_Scalar */	
573	//	SQSHL	//0x5E204C00,/* SQSHL	ARM64Op_sqshl_register_Scalar */	
574	//	SRSHL	//0x5E205400,/* SRSHL	ARM64Op_srshl_Scalar */	
575	//	SQRSHL	//0x5E205C00,/* SQRSHL	ARM64Op_sqrshl_Scalar */	
576	//	ADD	//0x5E208400,/* ADD	ARM64Op_add_vector_Scalar */	
577	//	CMTST	//0x5E208C00,/* CMTST	ARM64Op_cmtst_Scalar */	
578	//	SQDMULH	//0x5E20B400,/* SQDMULH	ARM64Op_sqdmulh_vector_Scalar */	
579	//	FMULX	//0x5E20DC00,/* FMULX	ARM64Op_fmulx_Scalar */	
580	//	FCMEQ	//0x5E20E400,/* FCMEQ	ARM64Op_fcmeq_register_Scalar */	
581	//	FRECPS	//0x5E20FC00,/* FRECPS	ARM64Op_frecps_Scalar */	
582	//	FRSQRTS	//0x5EA0FC00,/* FRSQRTS	ARM64Op_frsqrts_Scalar */	
583	//	UQADD	//0x7E200C00,/* UQADD	ARM64Op_uqadd_Scalar */	
584	//	UQSUB	//0x7E202C00,/* UQSUB	ARM64Op_uqsub_Scalar */	
585	//	CMHI	//0x7E203400,/* CMHI	ARM64Op_cmhi_register_Scalar */	
586	//	CMHS	//0x7E203C00,/* CMHS	ARM64Op_cmhs_register_Scalar */	

1	in_use	Opcode	//BINARY Opcode Opcodecomments
587	//	USHL	//0x7E204400,/* USHL ARM64Op_ushl_Scalar */
588	//	UQSHL	//0x7E204C00,/* UQSHL ARM64Op_uqshl_register_Scalar */
589	//	URSHL	//0x7E205400,/* URSHL ARM64Op_urshl_Scalar */
590	//	UQRSHL	//0x7E205C00,/* UQRSHL ARM64Op_uqrshl_Scalar */
591	//	SUB	//0x7E208400,/* SUB ARM64Op_sub_vector_Scalar */
592	//	CMEQ	//0x7E208C00,/* CMEQ ARM64Op_cmeq_register_Scalar */
593	//	SQRDMULH	//0x7E20B400,/* SQRDMULH ARM64Op_sqrdmulh_vector_Scalar */
594	//	FCMGE	//0x7E20E400,/* FCMGE ARM64Op_fcmge_register_Scalar */
595	//	FACGE	//0x7E20EC00,/* FACGE ARM64Op_facge_Scalar */
596	//	FABD	//0x7EA0D400,/* FABD ARM64Op_fabd_Scalar */
597	//	FCMGT	//0x7EA0E400,/* FCMGT ARM64Op_fcmgt_register_Scalar */
598	//	FACGT	//0x7EA0EC00,/* FACGT ARM64Op_facgt_Scalar */
599	//	**AdvSIMD scalar three diff**	/* AdvSIMD scalar three different */
600	//	SQDMLAL	//0x5E209000,/* SQDMLAL ARM64Op_sqdmlal_vector_Scalarwrites to low half
601	//	SQDMLAL2	//0x5E209000,/* SQDMLAL2 ARM64Op_sqdmlal2_vector_Scalarwrites to high ha
602	//	SQDMLSL	//0x5E20B000,/* SQDMLSL ARM64Op_sqdmlsl_vector_Scalarwrites to low half
603	//	SQDMLSL2	//0x5E20B000,/* SQDMLSL2 ARM64Op_sqdmlsl2_vector_Scalarwrites to high h
604	//	SQDMULL	//0x5E20D000,/* SQDMULL ARM64Op_sqdmull_vector_Scalarwrites to low half
605	//	SQDMULL2	//0x5E20D000,/* SQDMULL2 ARM64Op_sqdmull2_vector_Scalarwrites to high h
606	//	**AdvSIMD scalar two-reg r**	/* AdvSIMD scalar two-reg misc */
607	//	SUQADD	//0x5E203800,/* SUQADD ARM64Op_suqadd_Scalar */
608	//	SQABS	//0x5E207800,/* SQABS ARM64Op_sqabs_Scalar */
609	//	CMGT	//0x5E208800,/* CMGT ARM64Op_cmgt_zero_Scalar */
610	//	CMEQ	//0x5E209800,/* CMEQ ARM64Op_cmeq_zero_Scalar */
611	//	CMLT	//0x5E20A800,/* CMLT ARM64Op_cmlt_zero_Scalar */
612	//	ABS	//0x5E20B800,/* ABS ARM64Op_abs_Scalar */
613	//	SQXTN	//0x5E214800,/* SQXTN ARM64Op_sqxtn_Scalarwrites to low half of the dest.
614	//	SQXTN2	//0x5E214800,/* SQXTN2 ARM64Op_sqxtn2_Scalarwrites to high half of the de
615	//	FCVTNS	//0x5E21A800,/* FCVTNS ARM64Op_fcvtns_vector_Scalar */
616	//	FCVTMS	//0x5E21B800,/* FCVTMS ARM64Op_fcvtms_vector_Scalar */
617	//	FCVTAS	//0x5E21C800,/* FCVTAS ARM64Op_fcvtas_vector_Scalar */
618	//	SCVTF	//0x5E21D800,/* SCVTF ARM64Op_scvtf_vector_integer_Scalar */
619	//	FCMGT	//0x5EA0C800,/* FCMGT ARM64Op_fcmgt_zero_Scalar */
620	//	FCMEQ	//0x5EA0D800,/* FCMEQ ARM64Op_fcmeq_zero_Scalar */

1	in_use	Opcode	//BINARY Opcode	Opcodecomments
621	//	FCMLT	//0x5EA0E800,/* FCMLT	ARM64Op_fcmlt_zero_Scalar */
622	//	FCVTPS	//0x5EA1A800,/* FCVTPS	ARM64Op_fcvtps_vector_Scalar */
623	//	FCVTZS	//0x5EA1B800,/* FCVTZS	ARM64Op_fcvtzs_vector_integer_Scalar */
624	//	FRECPE	//0x5EA1D800,/* FRECPE	ARM64Op_frecpe_Scalar */
625	//	FRECPX	//0x5EA1F800,/* FRECPX	ARM64Op_frecpx */
626	//	USQADD	//0x7E203800,/* USQADD	ARM64Op_usqadd_Scalar */
627	//	SQNEG	//0x7E207800,/* SQNEG	ARM64Op_sqneg_Scalar */
628	//	CMGE	//0x7E208800,/* CMGE	ARM64Op_cmge_zero_Scalar */
629	//	CMLE	//0x7E209800,/* CMLE	ARM64Op_cmle_zero_Scalar */
630	//	NEG	//0x7E20B800,/* NEG	ARM64Op_neg_vector_Scalar */
631	//	SQXTUN	//0x7E212800,/* SQXTUN	ARM64Op_sqxtun_Scalarwrites to low half of the des
632	//	SQXTUN2	//0x7E212800,/* SQXTUN2	ARM64Op_sqxtun2_Scalarwrites to high half of the (
633	//	UQXTN	//0x7E214800,/* UQXTN	ARM64Op_uqxtn_Scalarwrites to low half of the dest.
634	//	UQXTN2	//0x7E214800,/* UQXTN2	ARM64Op_uqxtn2_Scalarwrites to high half of the de
635	//	FCVTXN	//0x7E216800,/* FCVTXN	ARM64Op_fcvtxn_Scalarwrites to low half of the dest
636	//	FCVTXN2	//0x7E216800,/* FCVTXN2	ARM64Op_fcvtxn2_Scalarwrites to high half of the d
637	//	FCVTNU	//0x7E21A800,/* FCVTNU	ARM64Op_fcvtnu_vector_Scalar */
638	//	FCVTMU	//0x7E21B800,/* FCVTMU	ARM64Op_fcvtmu_vector_Scalar */
639	//	FCVTAU	//0x7E21C800,/* FCVTAU	ARM64Op_fcvtau_vector_Scalar */
640	//	UCVTF	//0x7E21D800,/* UCVTF	ARM64Op_ucvtf_vector_integer_Scalar */
641	//	FCMGE	//0x7EA0C800,/* FCMGE	ARM64Op_fcmge_zero_Scalar */
642	//	FCMLE	//0x7EA0D800,/* FCMLE	ARM64Op_fcmle_zero_Scalar */
643	//	FCVTPU	//0x7EA1A800,/* FCVTPU	ARM64Op_fcvtpu_vector_Scalar */
644	//	FCVTZU	//0x7EA1B800,/* FCVTZU	ARM64Op_fcvtzu_vector_integer_Scalar */
645	//	FRSQRTE	//0x7EA1D800,/* FRSQRTE	ARM64Op_frsqrte_Scalar */
646	//	**AdvSIMD scalar pairwise**	/* AdvSIMD scalar pairwise */	
647	//	ADDP	//0x5E31B800,/* ADDP	ARM64Op_addp_scalar */
648	//	FMAXNMP	//0x7E30C800,/* FMAXNMP	ARM64Op_fmaxnmp_scalar */
649	//	FADDP	//0x7E30D800,/* FADDP	ARM64Op_faddp_scalar */
650	//	FMAXP	//0x7E30F800,/* FMAXP	ARM64Op_fmaxp_scalar */
651	//	FMINNMP	//0x7EB0C800,/* FMINNMP	ARM64Op_fminnmp_scalar */
652	//	FMINP	//0x7EB0F800,/* FMINP	ARM64Op_fminp_scalar */
653	//	**AdvSIMD scalar copy**	/* AdvSIMD scalar copy */	
654	//	DUP	//0x5E000400,/* DUP	ARM64Op_dup_element_Scalar */

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
655	//	**AdvSIMD scalar x indexe**	/* AdvSIMD scalar x indexed element */		
656	//	SQDMLAL	//0x5F003000,/* SQDMLAL	ARM64Op_sqdmlal_by_element_Scalar */	
657	//	SQDMLAL2	//0x5F003000,/* SQDMLAL2	ARM64Op_sqdmlal2_by_element_Scalar */	
658	//	SQDMLSL	//0x5F007000,/* SQDMLSL	ARM64Op_sqdmlsl_by_element_Scalar */	
659	//	SQDMLSL2	//0x5F007000,/* SQDMLSL2	ARM64Op_sqdmlsl2_by_element_Scalar */	
660	//	SQDMULL	//0x5F00B000,/* SQDMULL	ARM64Op_sqdmull_by_element_Scalar */	
661	//	SQDMULL2	//0x5F00B000,/* SQDMULL2	ARM64Op_sqdmull2_by_element_Scalar */	
662	//	SQDMULH	//0x5F00C000,/* SQDMULH	ARM64Op_sqdmulh_by_element_Scalar */	
663	//	SQRDMULH	//0x5F00D000,/* SQRDMULH	ARM64Op_sqrdmulh_by_element_Scalar */	
664	//	FMLA	//0x5F801000,/* FMLA	ARM64Op_fmla_by_element_Scalar */	
665	//	FMLS	//0x5F805000,/* FMLS	ARM64Op_fmls_by_element_Scalar */	
666	//	FMUL	//0x5F809000,/* FMUL	ARM64Op_fmul_by_element_Scalar */	
667	//	FMULX	//0x7F809000,/* FMULX	ARM64Op_fmulx_by_element_Scalar */	
668	//	**AdvSIMD scalar shift by i**	/* AdvSIMD scalar shift by immediate */		
669	//	SSHR	//0x5F000400,/* SSHR	ARM64Op_sshr_Scalarimmh != 0000 */	
670	//	SSRA	//0x5F001400,/* SSRA	ARM64Op_ssra_Scalarimmh != 0000 */	
671	//	SRSHR	//0x5F002400,/* SRSHR	ARM64Op_srshr_Scalarimmh != 0000 */	
672	//	SRSRA	//0x5F003400,/* SRSRA	ARM64Op_srsra_Scalarimmh != 0000 */	
673	//	SHL	//0x5F005400,/* SHL	ARM64Op_shl_Scalarimmh != 0000 */	
674	//	SQSHL	//0x5F007400,/* SQSHL	ARM64Op_sqshl_immediate_Scalarimmh != 0000 */	
675	//	SQSHRN	//0x5F009400,/* SQSHRN	ARM64Op_sqshrn_Scalarimmh != 0000 */	
676	//	SQSHRN2	//0x5F009400,/* SQSHRN2	ARM64Op_sqshrn2_Scalarimmh != 0000 */	
677	//	SQRSHRN	//0x5F009C00,/* SQRSHRN	ARM64Op_sqrshrn_Scalarimmh != 0000 */	
678	//	SQRSHRN2	//0x5F009C00,/* SQRSHRN2	ARM64Op_sqrshrn2_Scalarimmh != 0000 */	
679	//	SCVTF	//0x5F00E400,/* SCVTF	ARM64Op_scvtf_vector_fixed_point_Scalarimmh != 0(	
680	//	FCVTZS	//0x5F00FC00,/* FCVTZS	ARM64Op_fcvtzs_vector_fixed_point_Scalarimmh !=	
681	//	USHR	//0x7F000400,/* USHR	ARM64Op_ushr_Scalarimmh != 0000 */	
682	//	USRA	//0x7F001400,/* USRA	ARM64Op_usra_Scalarimmh != 0000 */	
683	//	URSHR	//0x7F002400,/* URSHR	ARM64Op_urshr_Scalarimmh != 0000 */	
684	//	URSRA	//0x7F003400,/* URSRA	ARM64Op_ursra_Scalarimmh != 0000 */	
685	//	SRI	//0x7F004400,/* SRI	ARM64Op_sri_Scalarimmh != 0000 */	
686	//	SLI	//0x7F005400,/* SLI	ARM64Op_sli_Scalarimmh != 0000 */	
687	//	SQSHLU	//0x7F006400,/* SQSHLU	ARM64Op_sqshlu_Scalarimmh != 0000 */	
688	//	UQSHL	//0x7F007400,/* UQSHL	ARM64Op_uqshl_immediate_Scalarimmh != 0000 */	

1	in_use	Opcode	//BINARY Opcode Opcodecomments
689	//	SQSHRUN	//0x7F008400,/* SQSHRUN ARM64Op_sqshrun_Scalarimmh != 0000 */
690	//	SQSHRUN2	//0x7F008400,/* SQSHRUN2 ARM64Op_sqshrun2_Scalarimmh != 0000 */
691	//	SQRSHRUN	//0x7F008C00,/* SQRSHRUN ARM64Op_sqrshrun_Scalarimmh != 0000 */
692	//	SQRSHRUN2	//0x7F008C00,/* SQRSHRUN2 ARM64Op_sqrshrun2_Scalarimmh != 0000 */
693	//	UQSHRN	//0x7F009400,/* UQSHRN ARM64Op_uqshrn_Scalarimmh != 0000 */
694	//	UQRSHRN	//0x7F009C00,/* UQRSHRN ARM64Op_uqrshrn_Scalarimmh != 0000 */
695	//	UQRSHRN2	//0x7F009C00,/* UQRSHRN2 ARM64Op_uqrshrn2_Scalarimmh != 0000 */
696	//	UCVTF	//0x7F00E400,/* UCVTF ARM64Op_ucvtf_vector_fixed_point_Scalarimmh != 0
697	//	FCVTZU	//0x7F00FC00,/* FCVTZU ARM64Op_fcvtzu_vector_fixed_point_Scalarimmh !=
698	//	**Crypto three-reg SHA**	/* Crypto three-reg SHA */
699	//	SHA1C	//0x5E000000,/* SHA1C ARM64Op_sha1c */
700	//	SHA1P	//0x5E001000,/* SHA1P ARM64Op_sha1p */
701	//	SHA1M	//0x5E002000,/* SHA1M ARM64Op_sha1m */
702	//	SHA1SU0	//0x5E003000,/* SHA1SU0 ARM64Op_sha1su0 */
703	//	SHA256H	//0x5E004000,/* SHA256H ARM64Op_sha256h */
704	//	SHA256H2	//0x5E005000,/* SHA256H2 ARM64Op_sha256h2 */
705	//	SHA256SU1	//0x5E006000,/* SHA256SU1 ARM64Op_sha256su1 */
706	//	**Crypto two-reg SHA**	/* Crypto two-reg SHA */
707	//	SHA1H	//0x5E280800,/* SHA1H ARM64Op_sha1h */
708	//	SHA1SU1	//0x5E281800,/* SHA1SU1 ARM64Op_sha1su1 */
709	//	SHA256SU0	//0x5E282800,/* SHA256SU0 ARM64Op_sha256su0 */
710	//	**Crypto AES**	/* Crypto AES */
711	//	AESE	//0x4E284800,/* AESE ARM64Op_aese */
712	//	AESD	//0x4E285800,/* AESD ARM64Op_aesd */
713	//	AESMC	//0x4E286800,/* AESMC ARM64Op_aesmc */
714	//	AESIMC	//0x4E287800,/* AESIMC ARM64Op_aesimc */
715	//	**AdvSIMD three same**	/* AdvSIMD three same */
716	//	SHADD	//0x0E200400,/* SHADD ARM64Op_shadd */
717	//	SQADD	//0x0E200C00,/* SQADD ARM64Op_sqadd_Vector */
718	//	SRHADD	//0x0E201400,/* SRHADD ARM64Op_srhadd */
719	//	SHSUB	//0x0E202400,/* SHSUB ARM64Op_shsub */
720	//	SQSUB	//0x0E202C00,/* SQSUB ARM64Op_sqsub_Vector */
721	//	CMGT	//0x0E203400,/* CMGT ARM64Op_cmgt_register_Vector */
722	//	CMGE	//0x0E203C00,/* CMGE ARM64Op_cmge_register_Vector */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
723	//	SSHL Vector	//0x0E204400,/* SSHL VectoARM64Op_sshl vector */
724	//	SQSHL	//0x0E204C00,/* SQSHL ARM64Op_sqshl_register_Vector */
725	//	SRSHL	//0x0E205400,/* SRSHL ARM64Op_srshl_Vector */
726	//	SQRSHL	//0x0E205C00,/* SQRSHL ARM64Op_sqrshl_Vector */
727	//	SMAX	//0x0E206400,/* SMAX ARM64Op_smax */
728	//	SMIN	//0x0E206C00,/* SMIN ARM64Op_smin */
729	//	SABD	//0x0E207400,/* SABD ARM64Op_sabd */
730	//	SABA	//0x0E207C00,/* SABA ARM64Op_saba */
731	//	ADD	//0x0E208400,/* ADD ARM64Op_add_vector_Vector */
732	//	CMTST	//0x0E208C00,/* CMTST ARM64Op_cmtst_Vector */
733	//	MLA	//0x0E209400,/* MLA ARM64Op_mla_vector */
734	//	MUL	//0x0E209C00,/* MUL ARM64Op_mul_vector */
735	//	SMAXP	//0x0E20A400,/* SMAXP ARM64Op_smaxp */
736	//	SMINP	//0x0E20AC00,/* SMINP ARM64Op_sminp */
737	//	SQDMULH	//0x0E20B400,/* SQDMULH ARM64Op_sqdmulh_vector_Vector */
738	//	ADDP	//0x0E20BC00,/* ADDP ARM64Op_addp_vector */
739	//	FMAXNM	//0x0E20C400,/* FMAXNM ARM64Op_fmaxnm_vector */
740	//	FMLA	//0x0E20CC00,/* FMLA ARM64Op_fmla_vector */
741	//	FADD	//0x0E20D400,/* FADD ARM64Op_fadd_vector */
742	//	FMULX	//0x0E20DC00,/* FMULX ARM64Op_fmulx_Vector */
743	//	FCMEQ	//0x0E20E400,/* FCMEQ ARM64Op_fcmeq_register_Vector */
744	//	FMAX	//0x0E20F400,/* FMAX ARM64Op_fmax_vector */
745	//	FRECPS	//0x0E20FC00,/* FRECPS ARM64Op_frecps_Vector */
746	//	AND	//0x0E201C00,/* AND ARM64Op_and_vector */
747	//	BIC	//0x0E601C00,/* BIC ARM64Op_bic_vector_register */
748	//	FMINNM	//0x0EA0C400,/* FMINNM ARM64Op_fminnm_vector */
749	//	FMLS	//0x0EA0CC00,/* FMLS ARM64Op_fmls_vector */
750	//	FSUB	//0x0EA0D400,/* FSUB ARM64Op_fsub_vector */
751	//	FMIN	//0x0EA0F400,/* FMIN ARM64Op_fmin_vector */
752	//	FRSQRTS	//0x0EA0FC00,/* FRSQRTS ARM64Op_frsqrts_Vector */
753	//	ORR	//0x0EA01C00,/* ORR ARM64Op_orr_vector_register */
754	//	ORN	//0x0EE01C00,/* ORN ARM64Op_orn_vector */
755	//	UHADD	//0x2E200400,/* UHADD ARM64Op_uhadd */
756	//	UQADD	//0x2E200C00,/* UQADD ARM64Op_uqadd_Vector */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
757	//	URHADD	//0x2E201400,/* URHADD ARM64Op_urhadd */
758	//	UHSUB	//0x2E202400,/* UHSUB ARM64Op_uhsub */
759	//		//0x2E202C00,/* ARM64Op_ _Vector */
760	//	CMHI	//0x2E203400,/* CMHI ARM64Op_cmhi_register_Vector */
761	//	CMHS	//0x2E203C00,/* CMHS ARM64Op_cmhs_register_Vector */
762	//	USHL	//0x2E204400,/* USHL ARM64Op_ushl_Vector */
763	//	UQSHL	//0x2E204C00,/* UQSHL ARM64Op_uqshl_register_Vector */
764	//	URSHL	//0x2E205400,/* URSHL ARM64Op_urshl_Vector */
765	//	UQRSHL	//0x2E205C00,/* UQRSHL ARM64Op_uqrshl_Vector */
766	//	UMAX	//0x2E206400,/* UMAX ARM64Op_umax */
767	//	UMIN	//0x2E206C00,/* UMIN ARM64Op_umin */
768	//	UABD	//0x2E207400,/* UABD ARM64Op_uabd */
769	//	UABA	//0x2E207C00,/* UABA ARM64Op_uaba */
770	//	SUB	//0x2E208400,/* SUB ARM64Op_sub_vector_Vector */
771	//	CMEQ	//0x2E208C00,/* CMEQ ARM64Op_cmeq_register_Vector */
772	//	MLS	//0x2E209400,/* MLS ARM64Op_mls_vector */
773	//	PMUL	//0x2E209C00,/* PMUL ARM64Op_pmul */
774	//	UMAXP	//0x2E20A400,/* UMAXP ARM64Op_umaxp */
775	//	UMINP	//0x2E20AC00,/* UMINP ARM64Op_uminp */
776	//	SQRDMULH	//0x2E20B400,/* SQRDMULH ARM64Op_sqrdmulh_vector_Vector */
777	//	FMAXNMP	//0x2E20B400,/* FMAXNMP ARM64Op_fmaxnmp_vector */
778	//	FADDP	//0x2E20D400,/* FADDP ARM64Op_faddp_vector */
779	//	FMUL	//0x2E20DC00,/* FMUL ARM64Op_fmul_vector */
780	//	FCMGE	//0x2E20E400,/* FCMGE ARM64Op_fcmge_register_Vector */
781	//	FACGE	//0x2E20EC00,/* FACGE ARM64Op_facge_Vector */
782	//	FMAXP	//0x2E20F400,/* FMAXP ARM64Op_fmaxp_vector */
783	//	FDIV	//0x2E20FC00,/* FDIV ARM64Op_fdiv_vector */
784	//	EOR	//0x2E201C00,/* EOR ARM64Op_eor_vector */
785	//	BSL	//0x2E601C00,/* BSL ARM64Op_bsl */
786	//	FMINNMP	//0x2EA0C400,/* FMINNMP ARM64Op_fminnmp_vector */
787	//	FABD	//0x2EA0D400,/* FABD ARM64Op_fabd_Vector */
788	//	FCMGT	//0x2EA0E400,/* FCMGT ARM64Op_fcmgt_register_Vector */
789	//	FACGT	//0x2EA0EC00,/* FACGT ARM64Op_facgt_Vector */
790	//	FMINP	//0x2EA0F400,/* FMINP ARM64Op_fminp_vector */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
791	//	BIT	//0x2EA01C00,/* BIT ARM64Op_bit */
792	//	BIF	//0x2EE01C00,/* BIF ARM64Op_bif */
793	//	**AdvSIMD three different**	/* AdvSIMD three different */
794	//	SADDL	//0x0E200000,/* SADDL ARM64Op_saddlwrites to low half of the dest. register
795	//	SADDL2	//0x4E200000,/* SADDL2 ARM64Op_saddl2writes to high half of the dest. regis
796	//	SADDW	//0x0E201000,/* SADDW ARM64Op_saddwwrites to low half of the dest. regist
797	//	SADDW2	//0x4E201000,/* SADDW2 ARM64Op_saddw2writes to high half of the dest. reg
798	//	SSUBL	//0x0E202000,/* SSUBL ARM64Op_ssublwrites to low half of the dest. register
799	//	SSUBL2	//0x4E202000,/* SSUBL2 ARM64Op_ssubl2writes to high half of the dest. regist
800	//	SSUBW	//0x0E203000,/* SSUBW ARM64Op_ssubwwrites to low half of the dest. registe
801	//	SSUBW2	//0x4E203000,/* SSUBW2 ARM64Op_ssubw2writes to high half of the dest. reg
802	//	ADDHN	//0x0E204000,/* ADDHN ARM64Op_addhnwrites to low half of the dest. registe
803	//	ADDHN2	//0x4E204000,/* ADDHN2 ARM64Op_addhn2writes to high half of the dest. regi
804	//	SABAL	//0x0E205000,/* SABAL ARM64Op_sabalwrites to low half of the dest. register
805	//	SABAL2	//0x4E205000,/* SABAL2 ARM64Op_sabal2writes to high half of the dest. regist
806	//	SUBHN	//0x0E206000,/* SUBHN ARM64Op_subhnwrites to low half of the dest. registe
807	//	SUBHN2	//0x4E206000,/* SUBHN2 ARM64Op_subhn2writes to high half of the dest. regi
808	//	SABDL	//0x0E207000,/* SABDL ARM64Op_sabdlwrites to low half of the dest. register
809	//	SABDL2	//0x4E207000,/* SABDL2 ARM64Op_sabdl2writes to high half of the dest. regis
810	//	SMLAL	//0x0E208000,/* SMLAL ARM64Op_smlal_vectorwrites to low half of the dest. r
811	//	SMLAL2	//0x4E208000,/* SMLAL2 ARM64Op_smlal2_vectorwrites to high half of the des
812	//	SQDMLAL	//0x0E209000,/* SQDMLAL ARM64Op_sqdmlal_vector_Vectorwrites to low half
813	//	SQDMLAL2	//0x4E209000,/* SQDMLAL2 ARM64Op_sqdmlal2_vector_Vectorwrites to high h
814	//	SMLSL	//0x0E20A000,/* SMLSL ARM64Op_smlsl_vectorwrites to low half of the dest. r
815	//	SMLSL2	//0x4E20A000,/* SMLSL2 ARM64Op_smlsl2_vectorwrites to high half of the des
816	//	SQDMLSL	//0x0E20B000,/* SQDMLSL ARM64Op_sqdmlsl_vector_Vectorwrites to low half
817	//	SQDMLSL2	//0x4E20B000,/* SQDMLSL2 ARM64Op_sqdmlsl2_vector_Vectorwrites to high h
818	//	SMULL	//0x0E20C000,/* SMULL ARM64Op_smull_vectorwrites to low half of the dest.
819	//	SMULL2	//0x4E20C000,/* SMULL2 ARM64Op_smull2_vectorwrites to high half of the des
820	//	SQDMULL	//0x0E20D000,/* SQDMULL ARM64Op_sqdmull_vector_Vectorwrites to low half
821	//	SQDMULL2	//0x4E20D000,/* SQDMULL2 ARM64Op_sqdmull2_vector_Vectorwrites to high h
822	//	PMULL	//0x0E20E000,/* PMULL ARM64Op_pmullwrites to low half of the dest. register
823	//	PMULL2	//0x4E20E000,/* PMULL2 ARM64Op_pmull2writes to high half of the dest. regis
824	//	UADDL	//0x2E200000,/* UADDL ARM64Op_uaddlwrites to low half of the dest. register

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
825	//	UADDL2	//0x6E200000,/*	UADDL2	ARM64Op_uaddl2writes to high half of the dest. regis
826	//	UADDW	//0x2E201000,/*	UADDW	ARM64Op_uaddwwrites to low half of the dest. regist
827	//	UADDW2	//0x6E201000,/*	UADDW2	ARM64Op_uaddw2writes to high half of the dest. reg
828	//	USUBL	//0x2E202000,/*	USUBL	ARM64Op_usublwrites to low half of the dest. register
829	//	USUBL2	//0x6E202000,/*	USUBL2	ARM64Op_usubl2writes to high half of the dest. regis
830	//	USUBW	//0x2E203000,/*	USUBW	ARM64Op_usubwwrites to low half of the dest. regist
831	//	USUBW2	//0x6E203000,/*	USUBW2	ARM64Op_usubw2writes to high half of the dest. reg
832	//	RADDHN	//0x2E204000,/*	RADDHN	ARM64Op_raddhnwrites to low half of the dest. regis
833	//	RADDHN2	//0x6E204000,/*	RADDHN2	ARM64Op_raddhn2writes to high half of the dest. re
834	//	UABAL	//0x2E205000,/*	UABAL	ARM64Op_uabalwrites to low half of the dest. register
835	//	UABAL2	//0x6E205000,/*	UABAL2	ARM64Op_uabal2writes to high half of the dest. regis
836	//	RSUBHN	//0x2E206000,/*	RSUBHN	ARM64Op_rsubhnwrites to low half of the dest. regis
837	//	RSUBHN2	//0x6E206000,/*	RSUBHN2	ARM64Op_rsubhn2writes to high half of the dest. re
838	//	UABDL	//0x2E207000,/*	UABDL	ARM64Op_uabdlwrites to low half of the dest. register
839	//	UABDL2	//0x6E207000,/*	UABDL2	ARM64Op_uabdl2writes to high half of the dest. regis
840	//	UMLAL	//0x2E208000,/*	UMLAL	ARM64Op_umlal_vectorwrites to low half of the dest.
841	//	UMLAL2	//0x6E208000,/*	UMLAL2	ARM64Op_umlal2_vectorwrites to high half of the des
842	//	UMLSL	//0x2E20A000,/*	UMLSL	ARM64Op_umlsl_vectorwrites to low half of the dest.
843	//	UMLSL2	//0x6E20A000,/*	UMLSL2	ARM64Op_umlsl2_vectorwrites to high half of the des
844	//	UMULL	//0x2E20C000,/*	UMULL	ARM64Op_umull_vectorwrites to low half of the dest.
845	//	UMULL2	//0x6E20C000,/*	UMULL2	ARM64Op_umull2_vectorwrites to high half of the de
846	//	**AdvSIMD two-reg misc**	/* AdvSIMD two-reg misc */		
847	//	REV64	//0x0E200800,/*	REV64	ARM64Op_rev64 */
848	//	REV16	//0x0E201800,/*	REV16	ARM64Op_rev16_vector */
849	//	SADDLP	//0x0E202800,/*	SADDLP	ARM64Op_saddlp */
850	//	SUQADD	//0x0E203800,/*	SUQADD	ARM64Op_suqadd_Vector */
851	//	CLS	//0x0E204800,/*	CLS	ARM64Op_cls_vector */
852	//	CNT	//0x0E205800,/*	CNT	ARM64Op_cnt */
853	//	SADALP	//0x0E206800,/*	SADALP	ARM64Op_sadalp */
854	//	SQABS	//0x0E207800,/*	SQABS	ARM64Op_sqabs_Vector */
855	//	CMGT	//0x0E208800,/*	CMGT	ARM64Op_cmgt_zero_Vector */
856	//	CMEQ	//0x0E209800,/*	CMEQ	ARM64Op_cmeq_zero_Vector */
857	//	CMLT	//0x0E20A800,/*	CMLT	ARM64Op_cmlt_zero_Vector */
858	//	ABS	//0x0E20B800,/*	ABS	ARM64Op_abs_Vector */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
859	//	XTN	//0x0E212800,/* XTN ARM64Op_xtn */
860	//	XTN2	//0x0E212800,/* XTN2 ARM64Op_xtn2 */
861	//	SQXTN	//0x0E214800,/* SQXTN ARM64Op_sqxtn_Vector */
862	//	SQXTN2	//0x0E214800,/* SQXTN2 ARM64Op_sqxtn2_Vector */
863	//	FCVTN	//0x0E216800,/* FCVTN ARM64Op_fcvtn */
864	//	FCVTN2	//0x0E216800,/* FCVTN2 ARM64Op_fcvtn2 */
865	//	FCVTL	//0x0E217800,/* FCVTL ARM64Op_fcvtl */
866	//	FCVTL2	//0x0E217800,/* FCVTL2 ARM64Op_fcvtl2 */
867	//	FRINTN	//0x0E218800,/* FRINTN ARM64Op_frintn_vector */
868	//	FRINTM	//0x0E219800,/* FRINTM ARM64Op_frintm_vector */
869	//	FCVTNS	//0x0E21A800,/* FCVTNS ARM64Op_fcvtns_vector_Vector */
870	//	FCVTMS	//0x0E21B800,/* FCVTMS ARM64Op_fcvtms_vector_Vector */
871	//	FCVTAS	//0x0E21C800,/* FCVTAS ARM64Op_fcvtas_vector_Vector */
872	//	SCVTF	//0x0E21D800,/* SCVTF ARM64Op_scvtf_vector_integer_Vector */
873	//	FCMGT	//0x0EA0C800,/* FCMGT ARM64Op_fcmgt_zero_Vector */
874	//	FCMEQ	//0x0EA0D800,/* FCMEQ ARM64Op_fcmeq_zero_Vector */
875	//	FCMLT	//0x0EA0E800,/* FCMLT ARM64Op_fcmlt_zero_Vector */
876	//	FABS	//0x0EA0F800,/* FABS ARM64Op_fabs_vector */
877	//	FRINTP	//0x0EA18800,/* FRINTP ARM64Op_frintp_vector */
878	//	FRINTZ	//0x0EA19800,/* FRINTZ ARM64Op_frintz_vector */
879	//	FCVTPS	//0x0EA1A800,/* FCVTPS ARM64Op_fcvtps_vector_Vector */
880	//	FCVTZS	//0x0EA1B800,/* FCVTZS ARM64Op_fcvtzs_vector_integer_Vector */
881	//	URECPE	//0x0EA1C800,/* URECPE ARM64Op_urecpe */
882	//	FRECPE	//0x0EA1D800,/* FRECPE ARM64Op_frecpe_Vector */
883	//	REV32	//0x2E200800,/* REV32 ARM64Op_rev32_vector */
884	//	UADDLP	//0x2E202800,/* UADDLP ARM64Op_uaddlp */
885	//	USQADD	//0x2E203800,/* USQADD ARM64Op_usqadd_Vector */
886	//	CLZ	//0x2E204800,/* CLZ ARM64Op_clz_vector */
887	//	UADALP	//0x2E206800,/* UADALP ARM64Op_uadalp */
888	//	SQNEG	//0x2E207800,/* SQNEG ARM64Op_sqneg_Vector */
889	//	CMGE	//0x2E208800,/* CMGE ARM64Op_cmge_zero_Vector */
890	//	CMLE	//0x2E209800,/* CMLE ARM64Op_cmle_zero_Vector */
891	//	NEG	//0x2E20B800,/* NEG ARM64Op_neg_vector_Vector */
892	//	SQXTUN	//0x2E212800,/* SQXTUN ARM64Op_sqxtun_Vector */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
893	//	SQXTUN2	//0x2E212800,/* SQXTUN2 ARM64Op_sqxtun2_Vector */
894	//	SHLL	//0x2E213800,/* SHLL ARM64Op_shll */
895	//	SHLL2	//0x2E213800,/* SHLL2 ARM64Op_shll2 */
896	//	UQXTN	//0x2E214800,/* UQXTN ARM64Op_uqxtn_Vector */
897	//	UQXTN2	//0x2E214800,/* UQXTN2 ARM64Op_uqxtn2_Vector */
898	//	FCVTXN	//0x2E216800,/* FCVTXN ARM64Op_fcvtxn_Vector */
899	//	FCVTXN2	//0x2E216800,/* FCVTXN2 ARM64Op_fcvtxn2_Vector */
900	//	FRINTA	//0x2E218800,/* FRINTA ARM64Op_frinta_vector */
901	//	FRINTX	//0x2E219800,/* FRINTX ARM64Op_frintx_vector */
902	//	FCVTNU	//0x2E21A800,/* FCVTNU ARM64Op_fcvtnu_vector_Vector */
903	//	FCVTMU	//0x2E21B800,/* FCVTMU ARM64Op_fcvtmu_vector_Vector */
904	//	FCVTAU	//0x2E21C800,/* FCVTAU ARM64Op_fcvtau_vector_Vector */
905	//	UCVTF	//0x2E21D800,/* UCVTF ARM64Op_ucvtf_vector_integer_Vector */
906	//	NOT	//0x2E205800,/* NOT ARM64Op_not */
907	//	RBIT	//0x2E605800,/* RBIT ARM64Op_rbit_vector */
908	//	FCMGE	//0x2EA0C800,/* FCMGE ARM64Op_fcmge_zero_Vector */
909	//	FCMLE	//0x2EA0D800,/* FCMLE ARM64Op_fcmle_zero_Vector */
910	//	FNEG	//0x2EA0F800,/* FNEG ARM64Op_fneg_vector */
911	//	FRINTI	//0x2EA19800,/* FRINTI ARM64Op_frinti_vector */
912	//	FCVTPU	//0x2EA1A800,/* FCVTPU ARM64Op_fcvtpu_vector_Vector */
913	//	FCVTZU	//0x2EA1B800,/* FCVTZU ARM64Op_fcvtzu_vector_integer_Vector */
914	//	URSQRTE	//0x2EA1C800,/* URSQRTE ARM64Op_ursqrte */
915	//	FRSQRTE	//0x2EA1D800,/* FRSQRTE ARM64Op_frsqrte_Vector */
916	//	FSQRT	//0x2EA1F800,/* FSQRT ARM64Op_fsqrt_vector */
917	//	**AdvSIMD across lanes**	/* AdvSIMD across lanes */
918	//	SADDLV	//0x0E303800,/* SADDLV ARM64Op_saddlv */
919	//	SMAXV	//0x0E30A800,/* SMAXV ARM64Op_smaxv */
920	//	SMINV	//0x0E31A800,/* SMINV ARM64Op_sminv */
921	//	ADDV	//0x0E31B800,/* ADDV ARM64Op_addv */
922	//	UADDLV	//0x2E303800,/* UADDLV ARM64Op_uaddlv */
923	//	UMAXV	//0x2E30A800,/* UMAXV ARM64Op_umaxv */
924	//	UMINV	//0x2E31A800,/* UMINV ARM64Op_uminv */
925	//	FMAXNMV	//0x2E30C800,/* FMAXNMV ARM64Op_fmaxnmv */
926	//	FMAXV	//0x2E30F800,/* FMAXV ARM64Op_fmaxv */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
927	//	FMINNMV	//0x2EB0C800,/* FMINNMV ARM64Op_fminnmv */
928	//	FMINV	//0x2EB0F800,/* FMINV ARM64Op_fminv */
929	//	**AdvSIMD copy**	/* AdvSIMD copy */
930	//	DUP	//0x0E000400,/* DUP ARM64Op_dup_element_Vector */
931	//	DUP	//0x0E000C00,/* DUP ARM64Op_dup_general */
932	//	SMOV	//0x0E002C00,/* SMOV ARM64Op_smov_32_bit */
933	//	UMOV	//0x0E003C00,/* UMOV ARM64Op_umov_32_bit */
934	//	INS	//0x4E001C00,/* INS ARM64Op_ins_general */
935	//	SMOV	//0x4E002C00,/* SMOV ARM64Op_smov_64_bit */
936	//	UMOV	//0x4E003C00,/* UMOV ARM64Op_umov_64_bit */
937	//	INS	//0x6E000400,/* INS ARM64Op_ins_element */
938	//	**AdvSIMD vector x indexe**	/* AdvSIMD vector x indexed element */
939	//	SMLAL	//0x0F002000,/* SMLAL ARM64Op_smlal_by_element */
940	//	SMLAL2	//0x0F002000,/* SMLAL2 ARM64Op_smlal2_by_element */
941	//	SQDMLAL	//0x0F003000,/* SQDMLAL ARM64Op_sqdmlal_by_element_Vector */
942	//	SQDMLAL2	//0x0F003000,/* SQDMLAL2 ARM64Op_sqdmlal2_by_element_Vector */
943	//	SMLSL	//0x0F006000,/* SMLSL ARM64Op_smlsl_by_element */
944	//	SMLSL2	//0x0F006000,/* SMLSL2 ARM64Op_smlsl2_by_element */
945	//	SQDMLSL	//0x0F007000,/* SQDMLSL ARM64Op_sqdmlsl_by_element_Vector */
946	//	SQDMLSL2	//0x0F007000,/* SQDMLSL2 ARM64Op_sqdmlsl2_by_element_Vector */
947	//	MUL	//0x0F008000,/* MUL ARM64Op_mul_by_element */
948	//	SMULL	//0x0F00A000,/* SMULL ARM64Op_smull_by_element */
949	//	SMULL2	//0x0F00A000,/* SMULL2 ARM64Op_smull2_by_element */
950	//	SQDMULL	//0x0F00B000,/* SQDMULL ARM64Op_sqdmull_by_element_Vector */
951	//	SQDMULL2	//0x0F00B000,/* SQDMULL2 ARM64Op_sqdmull2_by_element_Vector */
952	//	SQDMULH	//0x0F00C000,/* SQDMULH ARM64Op_sqdmulh_by_element_Vector */
953	//	SQRDMULH	//0x0F00D000,/* SQRDMULH ARM64Op_sqrdmulh_by_element_Vector */
954	//	FMLA	//0x0F801000,/* FMLA ARM64Op_fmla_by_element_Vector */
955	//	FMLS	//0x0F805000,/* FMLS ARM64Op_fmls_by_element_Vector */
956	//	FMUL	//0x0F809000,/* FMUL ARM64Op_fmul_by_element_Vector */
957	//	MLA	//0x2F000000,/* MLA ARM64Op_mla_by_element */
958	//	UMLAL	//0x2F002000,/* UMLAL ARM64Op_umlal_by_element */
959	//	UMLAL2	//0x2F002000,/* UMLAL2 ARM64Op_umlal2_by_element */
960	//	MLS	//0x2F004000,/* MLS ARM64Op_mls_by_element */

1	in_use	Opcode	//BINARY Opcode	Opcodecomments
961	//	UMLSL	//0x2F006000,/* UMLSL	ARM64Op_umlsl_by_element */
962	//	UMLSL2	//0x2F006000,/* UMLSL2	ARM64Op_umlsl2_by_element */
963	//	UMULL	//0x2F00A000,/* UMULL	ARM64Op_umull_by_element */
964	//	UMULL2	//0x2F00A000,/* UMULL2	ARM64Op_umull2_by_element */
965	//	FMULX	//0x2F809000,/* FMULX	ARM64Op_fmulx_by_element_Vector */
966	//	**AdvSIMD modified immec**	/* AdvSIMD modified immediate */	
967	//	MOVI	//0x0F000400,/* MOVI	ARM64Op_movi_32_bit_shifted_immediate */
968	//	ORR	//0x0F001400,/* ORR	ARM64Op_orr_vector_immediate_32_bit */
969	//	MOVI	//0x0F008400,/* MOVI	ARM64Op_movi_16_bit_shifted_immediate */
970	//	ORR	//0x0F009400,/* ORR	ARM64Op_orr_vector_immediate_16_bit */
971	//	MOVI	//0x0F00C400,/* MOVI	ARM64Op_movi_32_bit_shifting_ones */
972	//	MOVI	//0x0F00E400,/* MOVI	ARM64Op_movi_8_bit */
973	//	FMOV	//0x0F00F400,/* FMOV	ARM64Op_fmov_vector_immediate_Single_precision
974	//	MVNI	//0x2F000400,/* MVNI	ARM64Op_mvni_32_bit_shifted_immediate */
975	//	BIC	//0x2F001400,/* BIC	ARM64Op_bic_vector_immediate_32_bit */
976	//	MVNI	//0x2F008400,/* MVNI	ARM64Op_mvni_16_bit_shifted_immediate */
977	//	BIC	//0x2F009400,/* BIC	ARM64Op_bic_vector_immediate_16_bit */
978	//	MVNI	//0x2F00C400,/* MVNI	ARM64Op_mvni_32_bit_shifting_ones */
979	//	MOVI	//0x2F00E400,/* MOVI	ARM64Op_movi_64_bit_scalar */
980	//	MOVI	//0x6F00E400,/* MOVI	ARM64Op_movi_64_bit_vector */
981	//	FMOV	//0x6F00F400,/* FMOV	ARM64Op_fmov_vector_immediate_Double_precision
982	//	**AdvSIMD shift by immedi**	/* AdvSIMD shift by immediate */	
983	//	SSHR	//0x0F000400,/* SSHR	ARM64Op_sshr_Vector */
984	//	SSRA	//0x0F001400,/* SSRA	ARM64Op_ssra_Vector */
985	//	SRSHR	//0x0F002400,/* SRSHR	ARM64Op_srshr_Vector */
986	//	SRSRA	//0x0F003400,/* SRSRA	ARM64Op_srsra_Vector */
987	//	SHL	//0x0F005400,/* SHL	ARM64Op_shl_Vector */
988	//	SQSHL	//0x0F007400,/* SQSHL	ARM64Op_sqshl_immediate_Vector */
989	//	SHRN	//0x0F008400,/* SHRN	ARM64Op_shrn */
990	//	SHRN2	//0x0F008400,/* SHRN2	ARM64Op_shrn2 */
991	//	RSHRN	//0x0F008C00,/* RSHRN	ARM64Op_rshrn */
992	//	RSHRN2	//0x0F008C00,/* RSHRN2	ARM64Op_rshrn2 */
993	//	SQSHRN	//0x0F009400,/* SQSHRN	ARM64Op_sqshrn_Vector */
994	//	SQSHRN2	//0x0F009400,/* SQSHRN2	ARM64Op_sqshrn2_Vector */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
995	//	SQRSHRN	//0x0F009C00,/* SQRSHRN ARM64Op_sqrshrn_Vector */
996	//	SQRSHRN2	//0x0F009C00,/* SQRSHRN2 ARM64Op_sqrshrn2_Vector */
997	//	SSHLL	//0x0F00A400,/* SSHLL ARM64Op_sshll */
998	//	SSHLL2	//0x0F00A400,/* SSHLL2 ARM64Op_sshll2 */
999	//	SCVTF	//0x0F00E400,/* SCVTF ARM64Op_scvtf_vector_fixed_point_Vector */
1000	//	FCVTZS	//0x0F00FC00,/* FCVTZS ARM64Op_fcvtzs_vector_fixed_point_Vector */
1001	//	USHR	//0x2F000400,/* USHR ARM64Op_ushr_Vector */
1002	//	USRA	//0x2F001400,/* USRA ARM64Op_usra_Vector */
1003	//	URSHR	//0x2F002400,/* URSHR ARM64Op_urshr_Vector */
1004	//	URSRA	//0x2F003400,/* URSRA ARM64Op_ursra_Vector */
1005	//	SRI	//0x2F004400,/* SRI ARM64Op_sri_Vector */
1006	//	SLI	//0x2F005400,/* SLI ARM64Op_sli_Vector */
1007	//	SQSHLU	//0x2F006400,/* SQSHLU ARM64Op_sqshlu_Vector */
1008	//	UQSHL	//0x2F007400,/* UQSHL ARM64Op_uqshl_immediate_Vector */
1009	//	SQSHRUN	//0x2F008400,/* SQSHRUN ARM64Op_sqshrun_Vector */
1010	//	SQSHRUN2	//0x2F008400,/* SQSHRUN2 ARM64Op_sqshrun2_Vector */
1011	//	SQRSHRUN	//0x2F008C00,/* SQRSHRUN ARM64Op_sqrshrun_Vector */
1012	//	SQRSHRUN2	//0x2F008C00,/* SQRSHRUN2 ARM64Op_sqrshrun2_Vector */
1013	//	UQSHRN	//0x2F009400,/* UQSHRN ARM64Op_uqshrn_Vector */
1014	//	UQRSHRN	//0x2F009C00,/* UQRSHRN ARM64Op_uqrshrn_Vector */
1015	//	UQRSHRN2	//0x2F009C00,/* UQRSHRN2 ARM64Op_uqrshrn2_Vector */
1016	//	USHLL	//0x2F00A400,/* USHLL ARM64Op_ushll */
1017	//	USHLL2	//0x2F00A400,/* USHLL2 ARM64Op_ushll2 */
1018	//	UCVTF	//0x2F00E400,/* UCVTF ARM64Op_ucvtf_vector_fixed_point_Vector */
1019	//	FCVTZU	//0x2F00FC00,/* FCVTZU ARM64Op_fcvtzu_vector_fixed_point_Vector */
1020	//	**AdvSIMD TBL/TBX**	/* AdvSIMD TBL/TBX */
1021	//	TBL	//0x0E000000,/* TBL ARM64Op_tbl_Single_register_table */
1022	//	TBX	//0x0E001000,/* TBX ARM64Op_tbx_Single_register_table */
1023	//	TBL	//0x0E002000,/* TBL ARM64Op_tbl_Two_register_table */
1024	//	TBX	//0x0E003000,/* TBX ARM64Op_tbx_Two_register_table */
1025	//	TBL	//0x0E004000,/* TBL ARM64Op_tbl_Three_register_table */
1026	//	TBX	//0x0E005000,/* TBX ARM64Op_tbx_Three_register_table */
1027	//	TBL	//0x0E006000,/* TBL ARM64Op_tbl_Four_register_table */
1028	//	TBX	//0x0E007000,/* TBX ARM64Op_tbx_Four_register_table */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
1029	//	**AdvSIMD ZIP/UZP/TRN**	/* AdvSIMD ZIP/UZP/TRN */
1030	//	UZP1	//0x0E001800,/* UZP1 ARM64Op_uzp1 */
1031	//	TRN1	//0x0E002800,/* TRN1 ARM64Op_trn1 */
1032	//	ZIP1	//0x0E003800,/* ZIP1 ARM64Op_zip1 */
1033	//	UZP2	//0x0E005800,/* UZP2 ARM64Op_uzp2 */
1034	//	TRN2	//0x0E006800,/* TRN2 ARM64Op_trn2 */
1035	//	ZIP2	//0x0E007800,/* ZIP2 ARM64Op_zip2 */
1036	//	**AdvSIMD EXT**	/* AdvSIMD EXT */
1037	//	EXT	//0x2E000000,/* EXT ARM64Op_ext */
1038	//	**Loads and stores**	/* Loads and stores */
1039	//	**AdvSIMD load/store multi**	/* AdvSIMD load/store multiple structures */
1040	//	ST4	//0x0C000000,/* ST4 ARM64Op_st4_multiple_structures_No_offset */
1041	//	ST1	//0x0C002000,/* ST1 ARM64Op_st1_multiple_structures_Four_registers */
1042	//	ST3	//0x0C004000,/* ST3 ARM64Op_st3_multiple_structures_No_offset */
1043	//	ST1	//0x0C006000,/* ST1 ARM64Op_st1_multiple_structures_Three_registers */
1044	//	ST1	//0x0C007000,/* ST1 ARM64Op_st1_multiple_structures_One_register */
1045	//	ST2	//0x0C008000,/* ST2 ARM64Op_st2_multiple_structures_No_offset */
1046	//	ST1	//0x0C00A000,/* ST1 ARM64Op_st1_multiple_structures_Two_registers */
1047	//	LD4	//0x0C400000,/* LD4 ARM64Op_ld4_multiple_structures_No_offset */
1048	//	LD1	//0x0C402000,/* LD1 ARM64Op_ld1_multiple_structures_Four_registers */
1049	//	LD3	//0x0C404000,/* LD3 ARM64Op_ld3_multiple_structures_No_offset */
1050	//	LD1	//0x0C406000,/* LD1 ARM64Op_ld1_multiple_structures_Three_registers */
1051	//	LD1	//0x0C407000,/* LD1 ARM64Op_ld1_multiple_structures_One_register */
1052	//	LD2	//0x0C408000,/* LD2 ARM64Op_ld2_multiple_structures_No_offset */
1053	//	LD1	//0x0C40A000,/* LD1 ARM64Op_ld1_multiple_structures_Two_registers */
1054	//	**AdvSIMD load/store multi**	/* AdvSIMD load/store multiple structures (post-indexed) */
1055	//	ST4	//0x0C800000,/* ST4 ARM64Op_st4_multiple_structures_Register_offsetRm !
1056	//	ST1	//0x0C802000,/* ST1 ARM64Op_st1_multiple_structures_Four_registers_regi
1057	//	ST3	//0x0C804000,/* ST3 ARM64Op_st3_multiple_structures_Register_offsetRm !
1058	//	ST1	//0x0C806000,/* ST1 ARM64Op_st1_multiple_structures_Three_registers_reg
1059	//	ST1	//0x0C807000,/* ST1 ARM64Op_st1_multiple_structures_One_register_regist
1060	//	ST2	//0x0C808000,/* ST2 ARM64Op_st2_multiple_structures_Register_offsetRm !
1061	//	ST1	//0x0C80A000,/* ST1 ARM64Op_st1_multiple_structures_Two_registers_regi
1062	//	ST4	//0x0C9F0000,/* ST4 ARM64Op_st4_multiple_structures_Immediate_offset */

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
1063	//	ST1	//0x0C9F2000,/* ST1		ARM64Op_st1_multiple_structures_Four_registers_imm
1064	//	ST3	//0x0C9F4000,/* ST3		ARM64Op_st3_multiple_structures_Immediate_offset */
1065	//	ST1	//0x0C9F6000,/* ST1		ARM64Op_st1_multiple_structures_Three_registers_im
1066	//	ST1	//0x0C9F7000,/* ST1		ARM64Op_st1_multiple_structures_One_register_imme
1067	//	ST2	//0x0C9F8000,/* ST2		ARM64Op_st2_multiple_structures_Immediate_offset */
1068	//	ST1	//0x0C9FA000,/* ST1		ARM64Op_st1_multiple_structures_Two_registers_imm
1069	//	LD4	//0x0CC00000,/* LD4		ARM64Op_ld4_multiple_structures_Register_offsetRm
1070	//	LD1	//0x0CC02000,/* LD1		ARM64Op_ld1_multiple_structures_Four_registers_regi
1071	//	LD3	//0x0CC04000,/* LD3		ARM64Op_ld3_multiple_structures_Register_offsetRm
1072	//	LD1	//0x0CC06000,/* LD1		ARM64Op_ld1_multiple_structures_Three_registers_re
1073	//	LD1	//0x0CC07000,/* LD1		ARM64Op_ld1_multiple_structures_One_register_regis
1074	//	LD2	//0x0CC08000,/* LD2		ARM64Op_ld2_multiple_structures_Register_offsetRm
1075	//	LD1	//0x0CC0A000,/* LD1		ARM64Op_ld1_multiple_structures_Two_registers_regi
1076	//	LD4	//0x0CDF0000,/* LD4		ARM64Op_ld4_multiple_structures_Immediate_offset */
1077	//	LD1	//0x0CDF2000,/* LD1		ARM64Op_ld1_multiple_structures_Four_registers_imn
1078	//	LD3	//0x0CDF4000,/* LD3		ARM64Op_ld3_multiple_structures_Immediate_offset */
1079	//	LD1	//0x0CDF6000,/* LD1		ARM64Op_ld1_multiple_structures_Three_registers_im
1080	//	LD1	//0x0CDF7000,/* LD1		ARM64Op_ld1_multiple_structures_One_register_imme
1081	//	LD2	//0x0CDF8000,/* LD2		ARM64Op_ld2_multiple_structures_Immediate_offset */
1082	//	LD1	//0x0CDFA000,/* LD1		ARM64Op_ld1_multiple_structures_Two_registers_imn
1083	//	**AdvSIMD load/store singl**	/* AdvSIMD load/store single structure */		
1084	//	ST1	//0x0D000000,/* ST1		ARM64Op_st1_single_structure_8_bit */
1085	//	ST3	//0x0D002000,/* ST3		ARM64Op_st3_single_structure_8_bit */
1086	//	ST1	//0x0D004000,/* ST1		ARM64Op_st1_single_structure_16_bit */
1087	//	ST3	//0x0D006000,/* ST3		ARM64Op_st3_single_structure_16_bit */
1088	//	ST1	//0x0D008000,/* ST1		ARM64Op_st1_single_structure_32_bit */
1089	//	ST1	//0x0D008400,/* ST1		ARM64Op_st1_single_structure_64_bit */
1090	//	ST3	//0x0D00A000,/* ST3		ARM64Op_st3_single_structure_32_bit */
1091	//	ST3	//0x0D00A400,/* ST3		ARM64Op_st3_single_structure_64_bit */
1092	//	ST2	//0x0D200000,/* ST2		ARM64Op_st2_single_structure_8_bit */
1093	//	ST4	//0x0D202000,/* ST4		ARM64Op_st4_single_structure_8_bit */
1094	//	ST2	//0x0D204000,/* ST2		ARM64Op_st2_single_structure_16_bit */
1095	//	ST4	//0x0D206000,/* ST4		ARM64Op_st4_single_structure_16_bit */
1096	//	ST2	//0x0D208000,/* ST2		ARM64Op_st2_single_structure_32_bit */

1	in_use	Opcode	//BINARY Opcode Opcodecomments
1097	//	ST2	//0x0D208400,/* ST2 ARM64Op_st2_single_structure_64_bit */
1098	//	ST4	//0x0D20A000,/* ST4 ARM64Op_st4_single_structure_32_bit */
1099	//	ST4	//0x0D20A400,/* ST4 ARM64Op_st4_single_structure_64_bit */
1100	//	LD1	//0x0D400000,/* LD1 ARM64Op_ld1_single_structure_8_bit */
1101	//	LD3	//0x0D402000,/* LD3 ARM64Op_ld3_single_structure_8_bit */
1102	//	LD1	//0x0D404000,/* LD1 ARM64Op_ld1_single_structure_16_bit */
1103	//	LD3	//0x0D406000,/* LD3 ARM64Op_ld3_single_structure_16_bit */
1104	//	LD1	//0x0D408000,/* LD1 ARM64Op_ld1_single_structure_32_bit */
1105	//	LD1	//0x0D408400,/* LD1 ARM64Op_ld1_single_structure_64_bit */
1106	//	LD3	//0x0D40A000,/* LD3 ARM64Op_ld3_single_structure_32_bit */
1107	//	LD3	//0x0D40A400,/* LD3 ARM64Op_ld3_single_structure_64_bit */
1108	//	LD1R	//0x0D40C000,/* LD1R ARM64Op_ld1r_No_offset */
1109	//	LD3R	//0x0D40E000,/* LD3R ARM64Op_ld3r_No_offset */
1110	//	LD2	//0x0D600000,/* LD2 ARM64Op_ld2_single_structure_8_bit */
1111	//	LD4	//0x0D602000,/* LD4 ARM64Op_ld4_single_structure_8_bit */
1112	//	LD2	//0x0D604000,/* LD2 ARM64Op_ld2_single_structure_16_bit */
1113	//	LD4	//0x0D606000,/* LD4 ARM64Op_ld4_single_structure_16_bit */
1114	//	LD2	//0x0D608000,/* LD2 ARM64Op_ld2_single_structure_32_bit */
1115	//	LD2	//0x0D608400,/* LD2 ARM64Op_ld2_single_structure_64_bit */
1116	//	LD4	//0x0D60A000,/* LD4 ARM64Op_ld4_single_structure_32_bit */
1117	//	LD4	//0x0D60A400,/* LD4 ARM64Op_ld4_single_structure_64_bit */
1118	//	LD2R	//0x0D60C000,/* LD2R ARM64Op_ld2r_No_offset */
1119	//	LD4R	//0x0D60E000,/* LD4R ARM64Op_ld4r_No_offset */
1120	//	**AdvSIMD load/store singl**	/* AdvSIMD load/store single structure (post-indexed) */
1121	//	ST1	//0x0D800000,/* ST1 ARM64Op_st1_single_structure_8_bit_register_offsetRr
1122	//	ST3	//0x0D802000,/* ST3 ARM64Op_st3_single_structure_8_bit_register_offsetRr
1123	//	ST1	//0x0D804000,/* ST1 ARM64Op_st1_single_structure_16_bit_register_offsetF
1124	//	ST3	//0x0D806000,/* ST3 ARM64Op_st3_single_structure_16_bit_register_offsetF
1125	//	ST1	//0x0D808000,/* ST1 ARM64Op_st1_single_structure_32_bit_register_offsetF
1126	//	ST1	//0x0D808400,/* ST1 ARM64Op_st1_single_structure_64_bit_register_offsetF
1127	//	ST3	//0x0D80A000,/* ST3 ARM64Op_st3_single_structure_32_bit_register_offsetF
1128	//	ST3	//0x0D80A400,/* ST3 ARM64Op_st3_single_structure_64_bit_register_offsetF
1129	//	ST1	//0x0D9F0000,/* ST1 ARM64Op_st1_single_structure_8_bit_immediate_offse
1130	//	ST3	//0x0D9F2000,/* ST3 ARM64Op_st3_single_structure_8_bit_immediate_offse

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
1131	//	ST1	//0x0D9F4000,/* ST1	ARM64Op_st1_single_structure_16_bit_immediate_offs	
1132	//	ST3	//0x0D9F6000,/* ST3	ARM64Op_st3_single_structure_16_bit_immediate_offs	
1133	//	ST1	//0x0D9F8000,/* ST1	ARM64Op_st1_single_structure_32_bit_immediate_offs	
1134	//	ST1	//0x0D9F8400,/* ST1	ARM64Op_st1_single_structure_64_bit_immediate_offs	
1135	//	ST3	//0x0D9FA000,/* ST3	ARM64Op_st3_single_structure_32_bit_immediate_offs	
1136	//	ST3	//0x0D9FA400,/* ST3	ARM64Op_st3_single_structure_64_bit_immediate_offs	
1137	//	ST2	//0x0DA00000,/* ST2	ARM64Op_st2_single_structure_8_bit_register_offsetRı	
1138	//	ST4	//0x0DA02000,/* ST4	ARM64Op_st4_single_structure_8_bit_register_offsetRı	
1139	//	ST2	//0x0DA04000,/* ST2	ARM64Op_st2_single_structure_16_bit_register_offsetF	
1140	//	ST4	//0x0DA06000,/* ST4	ARM64Op_st4_single_structure_16_bit_register_offsetF	
1141	//	ST2	//0x0DA08000,/* ST2	ARM64Op_st2_single_structure_32_bit_register_offsetF	
1142	//	ST2	//0x0DA08400,/* ST2	ARM64Op_st2_single_structure_64_bit_register_offsetF	
1143	//	ST4	//0x0DA0A000,/* ST4	ARM64Op_st4_single_structure_32_bit_register_offsetF	
1144	//	ST4	//0x0DA0A400,/* ST4	ARM64Op_st4_single_structure_64_bit_register_offsetF	
1145	//	ST2	//0x0DBF0000,/* ST2	ARM64Op_st2_single_structure_8_bit_immediate_offse	
1146	//	ST4	//0x0DBF2000,/* ST4	ARM64Op_st4_single_structure_8_bit_immediate_offse	
1147	//	ST2	//0x0DBF4000,/* ST2	ARM64Op_st2_single_structure_16_bit_immediate_offs	
1148	//	ST4	//0x0DBF6000,/* ST4	ARM64Op_st4_single_structure_16_bit_immediate_offs	
1149	//	ST2	//0x0DBF8000,/* ST2	ARM64Op_st2_single_structure_32_bit_immediate_offs	
1150	//	ST2	//0x0DBF8400,/* ST2	ARM64Op_st2_single_structure_64_bit_immediate_offs	
1151	//	ST4	//0x0DBFA000,/* ST4	ARM64Op_st4_single_structure_32_bit_immediate_offs	
1152	//	ST4	//0x0DBFA400,/* ST4	ARM64Op_st4_single_structure_64_bit_immediate_offs	
1153	//	LD1	//0x0DC00000,/* LD1	ARM64Op_ld1_single_structure_8_bit_register_offsetRı	
1154	//	LD3	//0x0DC02000,/* LD3	ARM64Op_ld3_single_structure_8_bit_register_offsetRı	
1155	//	LD1	//0x0DC04000,/* LD1	ARM64Op_ld1_single_structure_16_bit_register_offsetF	
1156	//	LD3	//0x0DC06000,/* LD3	ARM64Op_ld3_single_structure_16_bit_register_offsetF	
1157	//	LD1	//0x0DC08000,/* LD1	ARM64Op_ld1_single_structure_32_bit_register_offsetF	
1158	//	LD1	//0x0DC08400,/* LD1	ARM64Op_ld1_single_structure_64_bit_register_offsetF	
1159	//	LD3	//0x0DC0A000,/* LD3	ARM64Op_ld3_single_structure_32_bit_register_offsetI	
1160	//	LD3	//0x0DC0A400,/* LD3	ARM64Op_ld3_single_structure_64_bit_register_offsetI	
1161	//	LD1R	//0x0DC0C000,/* LD1R	ARM64Op_ld1r_Register_offsetRm != 11111 */	
1162	//	LD3R	//0x0DC0E000,/* LD3R	ARM64Op_ld3r_Register_offsetRm != 11111 */	
1163	//	LD1	//0x0DDF0000,/* LD1	ARM64Op_ld1_single_structure_8_bit_immediate_offse	
1164	//	LD3	//0x0DDF2000,/* LD3	ARM64Op_ld3_single_structure_8_bit_immediate_offse	

1	in_use	Opcode	//BINARY	Opcode	Opcodecomments
1165	//	LD1	//0x0DDF4000,/* LD1	ARM64Op_ld1_single_structure_16_bit_immediate_offs	
1166	//	LD3	//0x0DDF6000,/* LD3	ARM64Op_ld3_single_structure_16_bit_immediate_offs	
1167	//	LD1	//0x0DDF8000,/* LD1	ARM64Op_ld1_single_structure_32_bit_immediate_offs	
1168	//	LD1	//0x0DDF8400,/* LD1	ARM64Op_ld1_single_structure_64_bit_immediate_offs	
1169	//	LD3	//0x0DDFA000,/* LD3	ARM64Op_ld3_single_structure_32_bit_immediate_offs	
1170	//	LD3	//0x0DDFA400,/* LD3	ARM64Op_ld3_single_structure_64_bit_immediate_offs	
1171	//	LD1R	//0x0DDFC000,/* LD1R	ARM64Op_ld1r_Immediate_offset */	
1172	//	LD3R	//0x0DDFE000,/* LD3R	ARM64Op_ld3r_Immediate_offset */	
1173	//	LD2	//0x0DE00000,/* LD2	ARM64Op_ld2_single_structure_8_bit_register_offsetRı	
1174	//	LD4	//0x0DE02000,/* LD4	ARM64Op_ld4_single_structure_8_bit_register_offsetRı	
1175	//	LD2	//0x0DE04000,/* LD2	ARM64Op_ld2_single_structure_16_bit_register_offsetF	
1176	//	LD4	//0x0DE06000,/* LD4	ARM64Op_ld4_single_structure_16_bit_register_offsetF	
1177	//	LD2	//0x0DE08000,/* LD2	ARM64Op_ld2_single_structure_32_bit_register_offsetF	
1178	//	LD2	//0x0DE08400,/* LD2	ARM64Op_ld2_single_structure_64_bit_register_offsetF	
1179	//	LD4	//0x0DE0A000,/* LD4	ARM64Op_ld4_single_structure_32_bit_register_offsetF	
1180	//	LD4	//0x0DE0A400,/* LD4	ARM64Op_ld4_single_structure_64_bit_register_offsetF	
1181	//	LD2R	//0x0DE0C000,/* LD2R	ARM64Op_ld2r_Register_offsetRm != 11111 */	
1182	//	LD4R	//0x0DE0E000,/* LD4R	ARM64Op_ld4r_Register_offsetRm != 11111 */	
1183	//	LD2	//0x0DFF0000,/* LD2	ARM64Op_ld2_single_structure_8_bit_immediate_offse	
1184	//	LD4	//0x0DFF2000,/* LD4	ARM64Op_ld4_single_structure_8_bit_immediate_offse	
1185	//	LD2	//0x0DFF4000,/* LD2	ARM64Op_ld2_single_structure_16_bit_immediate_offs	
1186	//	LD4	//0x0DFF6000,/* LD4	ARM64Op_ld4_single_structure_16_bit_immediate_offs	
1187	//	LD2	//0x0DFF8000,/* LD2	ARM64Op_ld2_single_structure_32_bit_immediate_offs	
1188	//	LD2	//0x0DFF8400,/* LD2	ARM64Op_ld2_single_structure_64_bit_immediate_offs	
1189	//	LD4	//0x0DFFA000,/* LD4	ARM64Op_ld4_single_structure_32_bit_immediate_offs	
1190	//	LD4	//0x0DFFA400,/* LD4	ARM64Op_ld4_single_structure_64_bit_immediate_offs	
1191	//	LD2R	//0x0DFFC000,/* LD2R	ARM64Op_ld2r_Immediate_offset */	
1192	//	LD4R	//0x0DFFE000,/* LD4R	ARM64Op_ld4r_Immediate_offset */	