ARMv7 Quick Reference

Arithmetic Instructions			
ADC{S}	rx, ry, op2	rx = ry + op2 + C	
ADD{S}	rx, ry, op2	rx = ry + op2	
ADDW	rx, ry, #i_{12}	rx = ry + $i^{\emptyset}$	T
ADR	rx, ±rel_{12}	rx = PC ± rel	
CMN	rx, op2	rx + op2	
CMP	rx, op2	rx − op2	
QADD	rx, ry, rz	rx = SATS(ry + rz, 32)	D
QDADD	rx, ry, rz	rx = SATS(ry + SATS(2×rz, 32), 32)	D
QDSUB	rx, ry, rz	rx = SATS(ry − SATS(2×rz, 32), 32)	D
QSUB	rx, ry, rz	rx = SATS(ry − rz, 32)	D
RSB{S}	rx, ry, op2	rx = op2 − ry	
RSC{S}	rx, ry, op2	rx = op2 − (ry + C)	A
SBC{S}	rx, ry, op2	rx = ry − (op2 + C)	
SDIV	rx, ry, rz	rx = ry ÷ rz	7
SSAT	rx, #j_5, ry{slr}	rx = SATS(ry ≪ ≫ sh, j)$^{\pm}$	6
SSAT16	rx, #j_4, ry	rx = SATS($ry^{\pm}_{H1}$, j)$^{\pm}$:SATS($ry^{\pm}_{H0}$, j)$^{\pm}$	6,D
SUB{S}	rx, ry, op2	rx = ry − op2	
SUBW	rx, ry, #i_{12}	rx = ry − $i^{\emptyset}$	T
UDIV	rx, ry, rz	rx = ry ÷ rz	7
USAD8	rx, ry, rz	rx = $\sum_{n=0}^{3}$(ABS($ry^{\emptyset}_{Bn}$)−$rz^{\emptyset}_{Bn}$)	6,D
USADA8	rx, ry, rz, rw	rx = rw + $\sum_{n=0}^{3}$(ABS($ry^{\emptyset}_{Bn}$)−$rz^{\emptyset}_{Bn}$)	6,D
USAT	rx, #j_5, ry{slr}	rx = SATU(ry ≪ ≫ sh, j)$^{\pm}$	6
USAT16	rx, #i_4, ry	rx = SATU($ry^{\pm}_{H1}$, i)$^{\pm}$:SATU($ry^{\pm}_{H0}$, i)$^{\pm}$	6,D

Operand 2		
#i_{32}	$i_8 \gg i_4$:0	A
#i_{32}	0_{24}:i_8, 0_8:$i_8$$0_8$:$i_8$, i_8:$0_8$$i_8$:$0_8$ or i_8:$i_8$$i_8$:$i_8$	T
#i_{32}	1:i_7 ≪ {1..24}	T
rz	rz	
rz, LSL #n	rz ≪ {1..31}	
rz, LSR #n	rz ≫ {1..32}	
rz, ASR #n	rz ≫ {1..32}	
rz, ROR #n	rz ⋙ {1..31}	
rz, RRX	C:$rz_{31:1}$; C = rz_0	
rz, LSL rw	rz ≪ rw	A
rz, LSR rw	rz ≫ rw	A
rz, ASR rw	rz ≫ rw	A
rz, ROR rw	rz ⋙ rw	A

Bitwise and Move Instructions			
AND{S}	rx, ry, op2	rx = ry & op2	
ASR{*S*}	rx, ry, #j_5	rx = ry ≫ j	
ASR{*S*}	rx, ry, Rs	rx = ry ≫ Rs	
BFC	rx, #p, #n	$rx_{p+n-1:p}$ = 0_n	6t
BFI	rx, ry, #p, #n	$rx_{p+n-1:p}$ = $ry_{n-1:0}$	6t
BIC{S}	rx, ry, op2	rx = ry & ∼op2	
CLZ	rx, ry	rx = CountLeadingZeros(ry)	
EOR{S}	rx, ry, op2	rx = ry ⊕ op2	
LSL{*S*}	rx, ry, #i_5	rx = ry ≪ i	
LSL{*S*}	rx, ry, Rs	rx = ry ≪ Rs	
LSR{*S*}	rx, ry, #j_5	rx = ry ≫ j	
LSR{*S*}	rx, ry, Rs	rx = ry ≫ Rs	
MOV{S}	rx, op2	rx = op2	
MOVT	rx, #i_{16}	$rx_{31:16}$ = i	6t
MOVW	rx, #i_{16}	rx = $i^{\emptyset}$	
MVN{S}	rx, op2	rx = ∼op2	
ORN{S}	rx, ry, op2	rx = ry | ∼op2	T
ORR{S}	rx, ry, op2	rx = ry | op2	
RBIT	rx, ry	rx = ReverseBits(ry)	6t
REV	rx, ry	rx = ry_{B0}:ry_{B1}:ry_{B2}:ry_{B3}	6
REV16	rx, ry	rx = ry_{B2}:ry_{B3}:ry_{B0}:ry_{B1}	6
REVSH	rx, ry	rx = $ry^{\pm}_{B0}$:ry_{B1}	6
ROR{*S*}	rx, ry, #i_5	rx = ry ⋙ i	
ROR{*S*}	rx, ry, Rs	rx = ry ⋙ Rs	
RRX{*S*}	rx, ry	rx = C:$ry_{31:1}$; C = ry_0	
SBFX	rx, ry, #p, #n	rx = $ry^{\pm}_{p+n-1:p}$	6t
TEQ	rx, op2	rx ⊕ op2	
TST	rx, op2	rx & op2	
UBFX	rx, ry, #p, #n	rx = $ry^{\emptyset}_{p+n-1:p}$	6t

Branch and Jump Instructions			
B	rel_{26}	PC = PC + $rel^{\pm}_{25:2}$:$0_{1:0}$	A
B	rel_{25}	PC = PC + $rel^{\pm}_{24:1}$:0	T
Bcc	rel_{21}	if(cc) PC = PC + $rel^{\pm}_{20:1}$:0	I
BKPT	#i_{16}	BreakPoint(i)	I
BL	rel_{26}	LR=$PC_{31:1}$:0; PC+=$rel^{\pm}_{25:2}$:$0_{1:0}$	A
BL	rel_{25}	LR=$PC_{31:1}$:1; PC+=$rel^{\pm}_{24:1}$:0	T
BLX	rel_{26}	LR=$PC_{31:1}$:0; Set=1; PC+=$rel^{\pm}_{25:1}$:0	A
BLX	rel_{25}	LR=$PC_{31:1}$:1; Set=0; PC+=$rel^{\pm}_{24:2}$:$0_{1:0}$	T
BLX	rx	LR=$PC_{31:1}$:0; Set=rx_0; PC=$rx_{31:1}$:0	A
BX	rx	Set = rx_0; PC = $rx_{31:1}$:0	A
TBB	[rx, ry]	PC = PC + 2 × $[rx + ry]^{\emptyset}_{8}$	T
TBH	[rx, ry, LSL #1]	PC = PC + 2 × $[rx + 2 \times ry]^{\emptyset}_{16}$	T

Load and Store Instructions			
LDMDA	rx{!}, rlist	rlist = [rx−4×cnt+4]; if(!) rx−=4×cnt	A
LDMDB	rx{!}, rlist	rlist = [rx − 4×cnt]; if(!) rx−=4×cnt	
LDM*IA*	rx{!}, rlist	rlist = [rx]; if(!) rx += 4×cnt	
LDMIB	rx{!}, rlist	rlist = [rx + 4]; if(!) rx += 4×cnt	A
LDR{T}	rx, [addr]	rx = [addr]	
LDRB{T}	rx, [addr]	rx = $[addr]^{\emptyset}_{8}$	
LDRD	rx, ry, [addr]	ry:rx = [addr]	
LDRH{T}	rx, [addr]	rx = $[addr]^{\emptyset}_{16}$	
LDRSB{T}	rx, [addr]	rx = $[addr]^{\pm}_{8}$	
LDRSH{T}	rx, [addr]	rx = $[addr]^{\pm}_{16}$	
POP	rlist	rlist = [SP]; SP += 4×cnt	
PUSH	rlist	SP −= 4×cnt; [SP] = rlist	
STMDA	rx{!}, rlist	[rx−4×cnt+4] = rlist; if(!) rx−=4×cnt	A
STMDB	rx{!}, rlist	[rx − 4×cnt] = rlist; if(!) rx−=4×cnt	
STM*IA*	rx{!}, rlist	[rx] = rlist; if(!) rx += 4×cnt	
STMIB	rx{!}, rlist	[rx+4] = rlist; if(!) rx += 4×cnt	A
STR{T}	rx, [addr]	[addr] = rx	
STRB{T}	rx, [addr]	$[addr]_8$ = rx_{B0}	
STRD	rx, ry, [addr]	[addr] = ry:rx	
STRH{T}	rx, [addr]	$[addr]_{16}$ = rx_{H0}	

ARM LDR/STR Addressing Modes		
non-T	[rz{, #±i_8}]{!}	addr = rz ± i; if(!) rz = addr
xxR{,B}	[rz{, #±i_{12}}]{!}	addr = rz ± i; if(!) rz = addr
any	[rz]{, #±i_8}	addr = rz; rz ±= i
xxR{,B}{T}	[rz], #±i_{12}	addr = rz; rz ±= i
non-T	[rz, ±rw]{!}	addr = rz ± rw; if(!) rz = addr
xxR{,B}	[rz, ±rw{AS}]{!}	addr = rz ± AS(rw); if(!) rz = addr
any	[rz], ±rw	addr = rz; rz ±= rw
xxR{,B}{T}	[rz], ±rw{AS}	addr = rz; rz ±= AS(rw)
LD non-T	±rel_8	addr = PC ± rel
LDR{,B}	±rel_{12}	addr = PC ± rel

Thumb2 LDR/STR Addressing Modes		
any	[rz{, #i_8}]	addr = rz + i
xxR{,B,H,SB,SH}	[rz, #i_{12}]	addr = rz + i
xxR{,B,H,SB,SH}	[rz, #±i_8]{!}	addr = rz ± i; if(!) rz = addr
xxR{,B,H,SB,SH}	[rz], #±i_8	addr = rz; rz ±= i
xxR{,B,H,SB,SH}	[rz,rw{,LSL #i_2}]	addr = rz + rw ≪ i
LDR{,B,H,SB,SH}	±rel_{12}	addr = PC ± rel
xxRD	[rz{, #±i_{10}}]{!}	addr=rz±$i_{9:2}$:$0_{1:0}$; if(!) rz=addr
xxRD	[rz], #±i_{10}	addr = rz; rz ±= $i^{\pm}_{9:2}$:$0_{1:0}$
LDRD	±rel_{10}	addr = PC ± $rel_{9:2}$:$0_{1:0}$

Multiplication Instructions			
MLA	rx, ry, rz, rw	$rx = rw + ry \times rz$	
MLA{S}	rx, ry, rz, rw	$rx = rw + ry \times rz$	A
MLS	rx, ry, rz, rw	$rx = rw - ry \times rz$	6t
MUL	rx, ry, rz	$rx = ry \times rz$	
MUL{S}	rx, ry, rz	$rx = ry \times rz$	A
SMLAxy	rx, ry, rz, rw	$rx = rw + ry^{\pm}_{Hx} \bar{\times} rz^{\pm}_{Hy}$	D
SMLaD	rx, ry, rz, rw	$rx = rw + ry^{\pm}_{H0} \bar{\times} rz^{\pm}_{H0} \pm ry^{\pm}_{H1} \bar{\times} rz^{\pm}_{H1}$	6,D
SMLaDX	rx, ry, rz, rw	$rx = rw + ry^{\pm}_{H0} \bar{\times} rz^{\pm}_{H1} \pm ry^{\pm}_{H1} \bar{\times} rz^{\pm}_{H0}$	D
SMLaLD	rx, ry, rz, rw	$ry:rx \mathrel{+}= rz^{\pm}_{H0} \bar{\times} rw^{\pm}_{H0} \pm rz^{\pm}_{H1} \bar{\times} rw^{\pm}_{H1}$	6,D
SMLaLDX	rx, ry, rz, rw	$ry:rx \mathrel{+}= rz^{\pm}_{H0} \bar{\times} rw^{\pm}_{H1} \pm rz^{\pm}_{H1} \bar{\times} rw^{\pm}_{H0}$	D
SMLAL	rx, ry, rz, rw	$ry:rx \mathrel{+}= rz \bar{\times} rw$	
SMLAL{S}	rx, ry, rz, rw	$ry:rx \mathrel{+}= rz \bar{\times} rw$	A
SMLALxy	rx, ry, rz, rw	$ry:rx \mathrel{+}= rz^{\pm}_{Hx} \bar{\times} rw^{\pm}_{Hy}$	D
SMLAWy	rx, ry, rz, rw	$rx = rw + ry \bar{\times} rz^{\pm}_{Hy}$	D
SMMLa	rx, ry, rz, rw	$rx = rw \pm (ry \bar{\times} rz)_{63:32}$	6,D
SMMLaR	rx, ry, rz, rw	$rx = rw \pm (ry \bar{\times} rz + 0x80000000)_{63:32}$	D
SMMUL	rx, ry, rz	$rx = (ry \bar{\times} rz)_{63:32}$	6,D
SMMULR	rx, ry, rz	$rx = (ry \bar{\times} rz + 0x80000000)_{63:32}$	D
SMUaD	rx, ry, rz	$rx = ry^{\pm}_{H0} \bar{\times} rz^{\pm}_{H0} \pm ry^{\pm}_{H1} \bar{\times} rz^{\pm}_{H1}$	6,D
SMUaDX	rx, ry, rz	$rx = ry^{\pm}_{H0} \bar{\times} rz^{\pm}_{H1} \pm ry^{\pm}_{H1} \bar{\times} rz^{\pm}_{H0}$	D
SMULxy	rx, ry, rz	$rx = ry^{\pm}_{Hx} \bar{\times} rz^{\pm}_{Hy}$	D
SMULL	rx, ry, rz, rw	$ry:rx = rz \bar{\times} rw$	
SMULL{S}	rx, ry, rz, rw	$ry:rx = rz \bar{\times} rw$	A
SMULWy	rx, ry, rz	$rx = (ry \bar{\times} rz^{\pm}_{Hy})_{47:16}$	D
UMAAL	rx, ry, rz, rw	$ry:rx = ry + rx + rz \times rw$	D
UMLAL	rx, ry, rz, rw	$ry:rx \mathrel{+}= rz \times rw$	
UMULL	rx, ry, rz, rw	$ry:rx = rz \times rw$	

Parallel Instructions			
pADD16	rx, ry, rz	$\text{for(n=0..1) } rx_{Hn} = p(ry_{Hn} + rz_{Hn})$	6,D
pADD8	rx, ry, rz	$\text{for(n=0..3) } rx_{Bn} = p(ry_{Bn} + rz_{Bn})$	6,D
pASX	rx, ry, rz	$rx = p(ry_{H1} + rz_{H0}):p(ry_{H0} - rz_{H1})$	6,D
pSAX	rx, ry, rz	$rx = p(ry_{H1} - rz_{H0}):p(ry_{H0} + rz_{H1})$	6,D
pSUB16	rx, ry, rz	$\text{for(n=0..1) } rx_{Hn} = p(ry_{Hn} - rz_{Hn})$	6,D
pSUB8	rx, ry, rz	$\text{for(n=0..3) } rx_{Bn} = p(ry_{Bn} - rz_{Bn})$	6,D
SEL	rx, ry, rz	$\text{for(n=0..3) } rx_{Bn} = (\text{GEn ? ry : rz})_{Bn}$	6,D

Parallel Instruction Prefixes	
Q	Signed operation, Results are saturated
S	Signed operation, Results are truncated
SH	Signed operation, Results are right shifted by one
U	Unsigned operation, Results are truncated
UH	Unsigned operation, Results are right shifted by one
UQ	Unsigned operation, Results are saturated

Packing and Unpacking Instructions			
PKHBT	rx, ry, rz{sl}	$rx = (rz \ll sh)_{H1}:ry_{H0}$	6,D
PKHTB	rx, ry, rz{sr}	$rx = ry_{H1}:(rz \gg sh)_{H0}$	6,D
SXTAB	rx, ry, rz{rb}	$rx = ry + (rz \ggg sh)^{\pm}_{B0}$	6,D
SXTAB16	rx, ry, rz{rb}	$\text{for(n=0..1) } rx_{Hn}=ry_{Hn}+(rz \ggg sh)^{\pm}_{B2n}$	6,D
SXTAH	rx, ry, rz{rb}	$rx = ry + (rz \ggg sh)^{\pm}_{H0}$	6,D
SXTB	rx, ry{rb}	$rx = (ry \ggg sh)^{\pm}_{B0}$	6
SXTB16	rx, ry{rb}	$\text{for(n=0..1) } rx_{Hn} = (ry \ggg sh)^{\pm}_{B2n}$	6,D
SXTH	rx, ry{rb}	$rx = (ry \ggg sh)^{\pm}_{H0}$	6
UXTAB	rx, ry, rz{rb}	$rx = ry + (rz \ggg sh)^{\emptyset}_{B0}$	6,D
UXTAB16	rx, ry, rz{rb}	$\text{for(n=0..1) } rx_{Hn}=ry_{Hn}+(rz \ggg sh)^{\emptyset}_{B2n}$	6,D
UXTAH	rx, ry, rz{rb}	$rx = ry + (rz \ggg sh)^{\emptyset}_{H0}$	6,D
UXTB	rx, ry{rb}	$rx = (ry \ggg sh)^{\emptyset}_{B0}$	6
UXTB16	rx, ry{rb}	$\text{for(n=0..1) } rx_{Hn} = (ry \ggg sh)^{\emptyset}_{B2n}$	6,D
UXTH	rx, ry{rb}	$rx = (ry \ggg sh)^{\emptyset}_{H0}$	6

Exclusive Load and Store Instructions			
CLREX		ClearExclusiveLocal()	I,6k
LDREX	rx, [ry]	rx = [ry]; SetExclusiveMonitor	6k
LDREX	rx, [ry, #i_{10}]	$rx = [ry+i^{\emptyset}_{9:2}:0_{1:0}]$; SetExclusiveMonitor	T,6k
LDREXB	rx, [ry]	$rx = [ry]^{\emptyset}_{8}$; SetExclusiveMonitor	6k
LDREXD	rx, ry, [rz]	ry:rx = [rz]; SetExclusiveMonitor	6k
LDREXH	rx, [ry]	$rx = [ry]^{\emptyset}_{16}$; SetExclusiveMonitor	6k
STREX	rx,ry,[rz]	if(Pass) [rz] = ry; rx = Pass ? 1 : 0	6k
STREX	rx,ry,[rz,#i_{10}]	$\text{if(Pass) } [rz+i^{\emptyset}_{9:2}:0_{1:0}]=ry$; rx=Pass?1:0	T,6k
STREXB	rx,ry,[rz]	$\text{if(Pass) } [rz]_8 = ry_{B0}$; rx = Pass?1:0	6k
STREXD	rx,ry,rz,[rw]	if(Pass) [rw] = rz:ry; rx = Pass?1:0	6k
STREXH	rx,ry,[rz]	$\text{if(Pass) } [rz]_{16} = ry_{H0}$; rx = Pass?1:0	6k

System Instructions			
CPSI{D,E}	{aif}{, #mode}	{a}{i}{f} = (E ? 1: 0); MODE = mode	6
CPS	#mode	MODE = mode	6
ERET		PC = LR; CPSR = SPSR	7
HVC	#i_{16}	CallHypervisor(i)	7
MRS	rx, xPSR	rx = {CPSR,SPSR}	
MRS	rx, Rbanked	rx = Rbanked	7
MSR	xPSR, rx	{CPSR,SPSR} = rx	
MSR	Rbanked, rx	Rbanked = rx	7
MSR	xPSR_{cxsf}, i	$\{CPSR,SPSR\}_{f;s;x;c} = i_{f;s;x;c}$	A
MSR	xPSR_{cxsf}, rx	$\{CPSR,SPSR\}_{f;s;x;c} = rx_{f;s;x;c}$	
RFEdi	rx{!}	LDMdi rx{!}, {PC, CPSR}	
SMC	#i_4	CallSecureMonitor()	6k
SRSdi	SP{!}, #mode	STMdi SP_mode{!}, {LR, SPSR}	6

Special Instructions			
DBG	#i_4	DebugHint(i)	7
DMB	option	DataMemoryBarrier(option)	I,7
DSB	option	DataSynchronizationBarrier(option)	I,7
ISB	SY	InstructionSynchronizationBarrier(SY)	I,7
NOP			6k
PLD{W}	[addr]	PreloadData(addr)	
PLI	[addr]	PreloadInstr(addr)	7
SETEND	{BE/LE}	EndianState = {BE/LE}	I,6
SEV		SendEvent()	6k
SVC	#i_{24}	CallSupervisor()	A
UDF	#i_{16}	UndefinedException()	
WFE		WaitForEvent()	6k
WFI		WaitForInterrupt()	6k
YIELD		HintYield()	6k

Keys	
{S}	Optional suffix, if present update flags
{t}	Conditional for additional instructions (T or E)
{T}	LDR/STR instruction uses user privileges.
a	A or S to add or subtract operand.
x, y	Selects bottom (B) or top (T) half of register(s)
cc	Condition code (can suffix most ARM instructions)
di	DA, DB, IA or IB for decrease/increase before/after.
i, j	Immediate operand, range 0..max / 1..max+1
rx, ry, rz, rw	General register
Rbanked	Banked register
rlist	Comma separated list of registers within { }.
op2	Immediate or shifted register
xPSR	APSR, CPSR or SPSR
SAT{S,U}(x,b)	Saturated signed/unsigned b bit value
B{0,1,2,3}	Selected byte (bits 7:0, 15:8, 23:16 or 31:24)
H{0,1}	Selected half word (bits 15:0 or 31:16)
{rb}	Optional rotate (ROR 8, ROR 16 or ROR 24)
{slr}	Optional shift (LSL #{1..31} or ASR #{1..32})
{sl}	Optional left shift (LSL #{1..31})
{sr}	Optional right shift (ASR #{1..32})
{AS}	ARM shift or rotate (LSL/ROR #{1..31}, LSR/ASR #{1..32} or RRX)
value$^{\pm}$, value$^{\emptyset}$	Value is sign/zero extended
$\bar{\times}$ $\bar{\div}$ $\bar{\gg}$	Operation is signed

General Registers		
R0-R3		Arguments and return values (useable by Thumb16)
R4-R7		General purpose (must be preserved, useable by Thumb16)
R8-R11		General purpose registers (must be preserved)
R12	IP	Intra-procedure-call scratch register
R13	SP	Stack pointer
R14	LR	Return address
R15	PC	Program counter

Condition Codes		
EQ	Equal	Z
NE	Not equal	!Z
CS/HS	Carry set, Unsigned higher or same	C
CC/LO	Carry clear, Unsigned lower	!C
MI	Minus, Negative	N
PL	Plus, Positive or zero	!N
VS	Overflow	V
VC	No overflow	!V
HI	Unsigned higher	C & !Z
LS	Unsigned lower or same	!C | Z
GE	Signed greater than or equal	N = V
LT	Signed less than	N ≠ V
GT	Signed greater than	!Z & N = V
LE	Signed less than or equal	Z | N ≠ V
AL	Always (default)	1

DMB and DSB Options	
SY	Full system, Read and write
(SY)ST	Full system, Write only
ISH	Inner shareable, Read and write
ISHST	Inner shareable, Write only
NSH	Non-shareable, Read and write
NSHST	Non-shareable, Write only
OSH	Outer shareable, Read and write
OSHST	Outer sharable, Write only

Notes for Instruction Set	
6,6k,6t,7	Introduced in ARMv6, ARMv6k, ARMv6T2, or ARMv7
A	Only available in ARM mode
D	Not available on ARM-M without DSP extension
H	Thumb16 instruction can use high registers
I	Can't be conditional
S	Thumb16 instruction must have S suffix unless in IT block
T	Only available in Thumb mode

Thumb16 Bitwise and Move Instructions			
AND{S}	rx, ry	rx = rx & ry	S
ASR{S}	rx, ry, #j_5	rx = ry $\gg$ j	S
ASR{S}	rx, ry	rx = rx $\gg$ ry	S
BIC{S}	rx, ry	rx = rx & ~ry	S
EOR{S}	rx, ry	rx = rx $\oplus$ ry	S
LSL{S}	rx, ry, #i_5	rx = ry $\ll$ i	S
LSL{S}	rx, ry	rx = rx $\ll$ ry	S
LSR{S}	rx, ry, #j_5	rx = ry $\gg$ j	S
LSR{S}	rx, ry	rx = rx $\gg$ ry	S
MOV	rx, ry	rx = ry	H
MOVS	rx, ry	rx = ry	
MOV{S}	rx, #i_8	rx = $i^{\emptyset}$	S
MVN{S}	rx, ry	rx = ~ry	S
ORR{S}	rx, ry	rx = rx | ry	S
REV	rx, ry	rx = $ry_{7:0}$:$ry_{15:8}$:$ry_{23:16}$:$ry_{31:24}$	6
REV16	rx, ry	rx = $ry_{23:16}$:$ry_{31:24}$:$ry_{7:0}$:$ry_{15:8}$	6
REVSH	rx, ry	rx = $ry^{\pm}_{7:0}$:$ry_{15:8}$	6
ROR{S}	rx, ry	rx = rx $\ggg$ ry	S
SXTB	rx, ry	rx = $ry^{\pm}_{7:0}$	
SXTH	rx, ry	rx = $ry^{\pm}_{15:0}$	
TST	rx, ry	rx & ry	
UXTB	rx, ry	rx = $ry^{\emptyset}_{7:0}$	6
UXTH	rx, ry	rx = $ry^{\emptyset}_{15:0}$	6

Thumb16 Branch and Special Instructions			
B	rel_{12}	PC = PC + $rel^{\pm}_{11:1}$:0	
Bcc	rel_9	if(cc) PC = PC + $rel^{\pm}_{8:1}$:0	I
BKPT	#i_8	BreakPoint(i)	I
BL	rel_{23}	LR=$PC_{31:1}$:1; PC+=$rel^{\pm}_{22:1}$:0	
BLX	rel_{23}	LR=$PC_{31:1}$:1; Set=0; PC+=$rel^{\pm}_{22:2}$:$0_{1:0}$	
BLX	rx	LR=$PC_{31:1}$:1; Set=rx_0; PC=$rx_{31:1}$:0	
BX	rx	Set=rx_0; PC = $rx_{31:1}$:0	
CBNZ	rx, rel_7	if(rx ≠ 0) PC += $rel^{\emptyset}_{6:1}$:0	I,6t
CBZ	rx, rel_7	if(rx = 0) PC += $rel^{\emptyset}_{6:1}$:0	I,6t
CPSI{D,E}	{aif}	{a}{i}{f} = (E ? 1 : 0)	6
IT{t{t{t}}}cc		if(cc) NextInstruction	I,6t
NOP			6k
SETEND	{BE/LE}	EndianState = {BE/LE}	I,6
SEV		SendEvent()	7
SVC	#i_8	CallSupervisor()	
UDF	#i_8	UndefinedException()	
WFE		WaitForEvent()	7
WFI		WaitForInterrupt()	7
YIELD		HintYield()	7

Thumb16 Arithmetic Instructions			
ADC{S}	rx, ry	rx = rx + ry + C	S
ADD{S}	rx, ry, #i_3	rx = ry + $i^{\emptyset}$	S
ADD{S}	rx, #i_8	rx = rx + $i^{\emptyset}$	S
ADD{S}	rx, ry, rz	rx = ry + rz	S
ADD	rx, ry	rx = rx + ry	H
ADD	rx, SP, #i_8	rx = SP + $i^{\emptyset}$	
ADD	SP, #i_9	SP = SP + $i^{\emptyset}_{8:2}$:$0_{1:0}$	
ADR	rx, rel_{10}	rx = PC + $rel^{\emptyset}_{9:2}$:$0_{1:0}$	
CMN	rx, ry	rx + ry	
CMP	rx, #i_8	rx − $i^{\emptyset}$	
CMP	rx, ry	rx − ry	H
MUL{S}	rx, ry	rx = rx × ry	S
RSB{S}	rx, ry, #0	rx = 0 − ry	S
SBC{S}	rx, ry	rx = rx − (ry + C)	S
SUB{S}	rx, ry, #i_3	rx = ry − $i^{\emptyset}$	S
SUB{S}	rx, #i_8	rx = rx − $i^{\emptyset}$	S
SUB{S}	rx, ry, rz	rx = ry − rz	S
SUB	SP, #i_9	SP = SP − $i^{\emptyset}_{8:2}$:$0_{1:0}$	

Thumb16 Load and Store Instructions			
LDM*IA*	rx{!}, rlist	rlist = [rx]; if(!) rx += 4×cnt	
LDMIA	SP!, rlist	rlist = [SP]; SP += 4×cnt	
LDR	rx, [ry{, #i_7}]	rx = [ry + $i^{\emptyset}_{6:2}$:$0_{1:0}$]	
LDR	rx, [SP{, #i_{10}}]	rx = [SP + $i^{\emptyset}_{9:2}$:$0_{1:0}$]	
LDR	rx, rel_{10}	rx = [PC + $rel^{\emptyset}_{9:2}$:$0_{1:0}$]	
LDR	rx, [ry, rz]	rx = [ry + rz]	
LDRB	rx, [ry{, #i_5}]	rx = [ry + $i^{\emptyset}$]$^{\emptyset}_8$	
LDRB	rx, [ry, rz]	rx = [ry + rz]$^{\emptyset}_8$	
LDRH	rx, [ry{, #i_6}]	rx = [ry + $i^{\emptyset}_{5:1}$:0]$^{\emptyset}_{16}$	
LDRH	rx, [ry, rz]	rx = [ry + rz]$^{\emptyset}_{16}$	
LDRSB	rx, [ry, rz]	rx = [ry + rz]$^{\pm}_8$	
LDRSH	rx, [ry, rz]	rx = [ry + rz]$^{\pm}_{16}$	
POP	rlist	rlist = [SP]; SP += 4×cnt	
PUSH	rlist	SP −= 4×cnt; [SP] = rlist	
STM*IA*	rx!, rlist	[rx] = rlist; rx += 4×cnt	
STMDB	SP!, rlist	SP −= 4×cnt; [SP] = rlist	
STR	rx, [ry{, #i_7}]	[ry + $i^{\emptyset}_{6:2}$:$0_{1:0}$] = rx	
STR	rx, [SP{, #i_{10}}]	[SP + $i^{\emptyset}_{9:2}$:$0_{1:0}$] = rx	
STR	rx, [ry, rz]	[ry + rz] = rx	
STRB	rx, [ry{, #i_5}]	[ry + $i^{\emptyset}$]$_8$ = $rx_{7:0}$	
STRB	rx, [ry, rz]	[ry + rz]$_8$ = $rx_{7:0}$	
STRH	rx, [ry{, #i_6}]	[ry + $i^{\emptyset}_{5:1}$:0]$_{16}$ = $rx_{15:0}$	
STRH	rx, [ry, rz]	[ry + rz]$_{16}$ = $rx_{15:0}$	

ARMv7-A & ARMv7-R System

Current Program Status Register (CPSR)			
M	0x0000001f	Processor Operating Mode	
T	0x00000020	Instruction set (JT: 00=ARM, 01=Thumb)	
F	0x00000040	FIQ exception masked	
I	0x00000080	IRQ exception masked	
A	0x00000100	Asynchronous abort masked	6
E	0x00000200	Big-endian operation	6
IT	0x0600fc00	IT state bits	6t
GE{3..0}	0x000f0000	SIMD Greater than or equal to	6
J	0x01000000	Instr set (JT: 10=Jazelle, 11=ThumbEE)	6
Q	0x08000000	Cumulative saturation bit	
V	0x10000000	Overflow condition flag	
C	0x20000000	Carry condition flag	
Z	0x40000000	Zero condition flag	
N	0x80000000	Negative condition flag	

Processor Operating Modes			
usr	0x10	User	
fiq	0x11	FIQ	
irq	0x12	IRQ	
svc	0x13	Supervisor	
mon	0x16	Monitor (Secure only)	S
abt	0x17	Abort	
hyp	0x1a	Hypervisor (Non-secure only)	V
und	0x1b	Undefined	
sys	0x1f	System	

Vectors	
0x00	Reset
0x04	Undefined instruction
0x08	Supervisor Call / Secure Monitor Call / Hypervisor Call
0x0c	Prefetch abort
0x10	Data abort
0x14	Hyp trap
0x18	IRQ interrupt
0x1c	FIQ interrupt

Notes for System Registers and Tables	
6,6k,6t,7	Introduced in ARMv6, ARMv6k, ARMv6T2, or ARMv7
A	Only present on ARM-A
B	Banked between secure and non-secure usage
R	Only present on ARM-R
S	Only present with security extensions (Implies 6k,A)
V	Only present with virtualization extensions (Implies 7,A)

System Control Register (SCTLR)			
M	0x00000001	MMU enabled	B
A	0x00000002	Alignment check enabled	B
C	0x00000004	Data and unified caches enabled	B
CP15BEN	0x00000020	CP15 barrier enable	7,B
SW	0x00000400	Enable SWP and SWPB instructions	6,B
Z	0x00000800	Program flow prediction enabled	B
I	0x00001000	Instruction cache enabled	B
V	0x00002000	High exception vectors	B
RR	0x00004000	Round Robin select (Non-Secure RO)	
HA	0x00020000	Hardware access flag enable	B,S
BR	0x00020000	Background region enable	7,R
WXN	0x00080000	Write force to XN	V
DZ	0x00080000	Divide by zero causes undefined instruction	7,R
UWXN	0x00100000	Unprivileged write forced to XN for PL1	V
FI	0x00200000	Fast Interrupts (Non-Secure RO)	6
VE	0x01000000	Interrupt Vectors Enable	6,B
EE	0x02000000	Exception Endianess	6,B
NMFI	0x08000000	Non-maskable FIQ support (RO)	6
TRE	0x10000000	TEX remap functionality enabled	B,S
AFE	0x20000000	Access flag enable	B,S
TE	0x40000000	Thumb exception enable	6t,B
IE	0x80000000	Big-endian byte order in instructions	7,R

Coprocessor Access Control Register (CPACR)		
CP{0..13}	$3 \ll (2 \times \{0..13\})$	CP{0..13} access (00=denied, 01=privileged mode only, 11=privileged or user mode)
TRCDIS	0x10000000	Disable CP14 access to trace registers
D32DIS	0x40000000	Disable use of D16-D31 registers
ASEDIS	0x80000000	Disable advanced SIMD functionality

CP15 System Control Registers			
SCTLR	c1,0,c0,0	System Control Register	
ACTLR	c1,0,c0,1	Auxiliary Control Register	6,B
CPACR	c1,0,c0,2	Coprocessor Access Control Register	6
SCR	c1,0,c1,0	Secure Configuration (Secure only)	S
SDER	c1,0,c1,1	Secure Debug Enable (Secure only)	S
NSACR	c1,0,c1,2	Non-Secure Access Control (Non-Secure RO)	S

CP15 Security Extension Registers (ARM-A Only)			
VBAR	c12,0,c0,0	Vector Base Register	B
MVBAR	c12,0,c0,1	Monitor Vector Base Address (Secure only)	
ISR	c12,0,c1,0	Interrupt Status Register (RO)	

Secure Configuration Register (SCR)			
NS	0x001	System state is non-secure unless in Monitor mode	
IRQ	0x002	IRQs taken to Monitor mode	
FIQ	0x004	FIQs taken to Monitor mode	
EA	0x008	External aborts taken to Monitor mode	
FW	0x010	CPSR.F writeable in non-secure state	
AW	0x020	CPSR.A writeable in non-secure state	
nET	0x040	Disable early termination	
SCD	0x080	Secure monitor call disable	V
HCE	0x100	Hyp Call enable	V
SIF	0x200	Secure instruction fetch	V

Non-Secure Access Control Register (NSACR)		
CP{0..13}	$1 \ll \{0..13\}$	CP{0..13} can be accessed in non-secure state
NSD32DIS	0x00004000	CPACR.D32DIS is fixed 1 in non-secure state
NSASEDIS	0x00008000	CPACR.ASEDIS is fixed 1 in non-secure state
RFR	0x00080000	Reserve FIQ mode for non-secure
NSTRCDIS	0x00100000	Disable non-secure access to CP14 trace regs

CP15 Memory System Fault Registers			
DFSR	c5,0,c0,0	Data Fault Status Register	B
IFSR	c5,0,c0,1	Instruction Fault Status Register	6,B
ADFSR	c5,0,c1,0	Auxiliary DFSR	7,B
AIFSR	c5,0,c1,1	Auxiliary IFSR	7,B
DFAR	c6,0,c0,0	Data Fault Address Register	B
IFAR	c6,0,c0,2	Instruction Fault Address Register	6,B
DRBAR	c6,0,c1,0	Data Region Base Address Register	R
IRBAR	c6,0,c1,1	Instruction Region Base Address Register	R
DRSR	c6,0,c1,2	Data Region Size and Enable Register	R
IRSR	c6,0,c1,3	Instruction Region Size and Enable Register	R
DRACR	c6,0,c1,4	Data Region Access Control Register	R
IRACR	c6,0,c1,5	Instruction Region Access Control Register	R
RGNR	c6,0,c2,0	MPU Region Number Register	R

CP15 Generic Timer Registers			
CNTFRQ	c14,0,c0,0	Counter Frequency Reg (Non-Secure RO)	7
CNTKCTL	c14,0,c1,0	Timer PL1 Control Register	7
CNTP_TVAL	c14,0,c2,0	PL1 Physical TimerValue Register	7,B
CNTP_CTL	c14,0,c2,1	PL1 Physical Timer Control Register	7,B
CNTV_TVAL	c14,0,c3,0	Virtual TimerValue Register	7
CNTV_CTL	c14,0,c3,1	Virtual TimerControl Register	7
CNTPCT	c14,0	Physical Count Register (RO)	7
CNTVCT	c14,1	Virtual Count Register (RO)	7
CNTP_CVAL	c14,2	PL1 Physical Timer CompareValue Register	7,B
CNTV_CVAL	c14,3	Virtual Timer CompareValue Register	7

CP15 ID Registers (Read-Only)			
MIDR	c0,0,c0,0	Main ID Register	
CTR	c0,0,c0,1	Cache Type Register	
TCMTR	c0,0,c0,2	TCM Type Register	
TLBTR	c0,0,c0,3	TLB Type Register	A
MPUIR	c0,0,c0,4	MPU Type Register	R
MPIDR	c0,0,c0,5	Multiprocessor Affinity Register	
REVIDR	c0,0,c0,6	Revision ID	
ID_PFR{0..1}	c0,0,c1,{0..1}	Processor Feature Registers	6
ID_DFR0	c0,0,c1,2	Debug Feature Register 0	6
ID_AFR0	c0,0,c1,3	Auxiliary Feature Register 0	6
ID_MMFR{0..3}	c0,0,c1,{4..7}	Memory Model Feature Regs	6
ID_ISAR{0..5}	c0,0,c2,{0..5}	Instruction Set Attribute Regs	6
CCSIDR	c0,1,c0,0	Cache Size ID Register	7
CLIDR	c0,1,c0,1	Cache Level ID Register	7
AIDR	c0,1,c0,7	Auxiliary ID Register	7
CSSELR	c0,2,c0,0	Cache Size Selection Register (RW)	7,B

CP15 Cache Maintenance Registers (Write Only)			
CP15WFI	c7,0,c0,4	Wait for interrupt operation	
ICIALLUIS	c7,0,c1,0	Inv all instr caches to PoU Inner Sharable	7
BPIALLIS	c7,0,c1,6	Inv all branche predictors Inner Sharable	7
PAR	c7,0,c4,0	Physical Address Register (RW)	7,A,B
ICIALLU	c7,0,c5,0	Invalidate all instruction caches to PoU	
ICIMVAU	c7,0,c5,1	Inv instruction caches by MVA to PoU	
CP15ISB	c7,0,c5,4	Instruction Sync Barrier operation	7
BPIALL	c7,0,c5,6	Invalidate all branch predictors	
BPIMVA	c7,0,c5,7	Invalidate MVA from branch predictors	
DCIMVAC	c7,0,c6,1	Inv data cache line my MVA to PoC	
DCISW	c7,0,c6,2	Invalidate data cache line by set/way	
ATS1CPR	c7,0,c8,0	PL1 read translation (Current state)	7,A
ATS1CPW	c7,0,c8,1	PL1 write translation (Current state)	7,A
ATS1CUR	c7,0,c8,2	Unpriv read translation (Current state)	7,A
ATS1CUW	c7,0,c8,3	Unpriv write translation (Current state)	7,A
ATS12NSOPR	c7,0,c8,4	PL1 read translation (NS state)	7,S
ATS12NSOPW	c7,0,c8,5	PL1 write translation (NS state)	7,S
ATS12NSOUR	c7,0,c8,6	Unprivileged read translation (NS state)	7,S
ATS12NSOUW	c7,0,c8,7	Unprivileged write translation (NS state)	7,S
DCCMVAC	c7,0,c10,1	Clean data cache line my MVA to PoC	
DCCSW	c7,0,c10,2	Clean data cache line by set/way	
CP15DSB	c7,0,c10,4	Data Synchronization Barrier operation	7
CP15DMB	c7,0,c10,5	Data Memory Barrier operation	7
DCCMVAU	c7,0,c11,1	Clean data cache line by MVA to PoU	
DCCIMVAC	c7,0,c14,1	Clean and inv data c-line by MVA to PoC	
DCCISW	c7,0,c14,2	Clean and inv data c-line by set/way	
PAR	c7,0	Physical Address Register (RW)	7,A,B

CP15 Memory Protection and Control Registers (ARM-A only)			
TTBR0	c2,0,c0,0	Translation Table Base 0	B
TTBR1	c2,0,c0,1	Translation Table Base 1	6,B
TTBCR	c2,0,c0,2	Translation Table Base Control	6,B
TTBR0	c2,0	Translation Table Base 0 (LPAE only)	7,B
TTBR1	c2,1	Translation Table Base 1 (LPAE only)	7,B
DACR	c3,0,c0,0	Domain Access Control Register	B

CP15 TLB Maintenance Operation Regs (Write Only, ARM-A Only)			
TLBIALLIS	c8,0,c3,0	Invalidate entire TLB IS	7
TLBIMVAIS	c8,0,c3,1	Invalidate unified TLB by MVA and ASID IS	7
TLBIASIDIS	c8,0,c3,2	Invalidate unified TLB by ASID match IS	7
TLBIMVAAIS	c8,0,c3,3	Inv unified TLB entry by MVA all ASID IS	7
ITLIALL	c8,0,c5,0	Invalidate instruction TLB	
ITLIMVA	c8,0,c5,1	Inv instr TLB entry by MVA all ASID IS	
ITLIASID	c8,0,c5,2	Invalidate instruction TLB by ASID match	6
DTLBIALL	c8,0,c6,0	Invalidate data TLB	
DTLBIMVA	c8,0,c6,1	Invalidate data TLB entry by MVA and ASID	
DTLBIASID	c8,0,c6,2	Invalidate data TLB by ASID match	6
TLBIALL	c8,0,c7,0	Invalidate unified TLB	
TLBIMVA	c8,0,c7,1	Inv unified TLB entry by MVA and ASID	
TLBIASID	c8,0,c7,2	Invalidate unified TLB by ASID match	6
TLBIMVAA	c8,0,c7,3	Inval unified TLB entries by MVA all ASID	6

CP15 Performance Monitor Registers (ARM-R Only)		
PMCR	c9,0,c12,0	PM Control Register
PMCNTENSET	c9,0,c12,1	PM Count Enable Set Register
PMCNTENCLR	c9,0,c12,2	PM Count Enable Clear Register
PMOVSR	c9,0,c12,3	PM Overflow Flag Status Register
PMSWINC	c9,0,c12,4	PM Software Increment Register
PMSELR	c9,0,c12,5	PM Event Counter Selection Register
PMCEID0	c9,0,c12,6	PM Common Event Identification Register 0
PMCEID1	c9,0,c12,7	PM Common Event Identification Register 1
PMCCNTR	c9,0,c13,0	PM Cycle Count Register
PMXEVTYPER	c9,0,c13,1	PM Event Type Select Register
PMXEVCNTR	c9,0,c13,2	PM Event Count Register
PMUSERENR	c9,0,c14,0	PM User Enable Register
PMINTENSET	c9,0,c14,1	PM Interrupt Enable Set Register
PMINTENCLR	c9,0,c14,2	PM Interrupt Enable Clear Register

CP15 Memory Mapping Registers (ARM-A Only)			
PRRR	c10,0,c2,0	Primary Region Remap Register	6,B
NMRR	c10,0,c2,1	Normal Memory Remap Register	6,B
AMAIR0	c10,0,c3,0	Aux Memory Attribute Indirection Reg 0	7
AMAIR1	c10,0,c3,1	Aux Memory Attribute Indirection Reg 1	7

CP15 Process, Context, and Thread ID Registers			
FCSEIDR	c13,0,c0,0	FSCE PID Register	A,B
CONTEXIDR	c13,0,c0,1	Context ID Register	6,B
TPIDRURW	c13,0,c0,2	User Read/Write Thread ID	6,B
TPIDRURO	c13,0,c0,3	User Read-only Thread ID	6,B
TPIDRPRW	c13,0,c0,4	PL1 only Thread ID	6,B

CP15 Virtualization Extension Registers (ARM-A Only)		
VPIDR	c0,4,c0,0	Virtualization Processor ID Register
VMPIDR	c0,4,c0,5	Virtualization Multiproc ID Register
HSCTLR	c1,4,c0,0	Hyp System Control Register
HACTLR	c1,4,c0,1	Hyp Auxiliary Control Register
HCR	c1,4,c1,0	Hyp Configuration Register
HDCR	c1,4,c1,1	Hyp Debug Configuration Register
HCPTR	c1,4,c1,2	Hyp Coprocessor Trap Register
HSTR	c1,4,c1,3	Hyp System Trap Register
HACR	c1,4,c1,7	Hyp Auxiliary Configuration Register
HTCR	c2,4,c0,2	Hyp Translation Control Register
VTCR	c2,4,c1,2	Virtualization Translation Control Reg
HTTBR	c2,4	Hyp Translation Table Base Reg
VTTBR	c2,6	Virt Translation Table Base Reg
HADFSR	c5,4,c1,0	Hyp Auxiliary DFSR
HAIFSR	c5,4,c1,1	Hyp Auxiliary IFSR
HSR	c5,4,c2,0	Hyp Syndrome Register
HDFAR	c6,4,c0,0	Hyp Data Fault Address Register
HIFAR	c6,4,c0,2	Hyp Instruction Fault Address Register
HPFAR	c6,4,c0,4	Hyp IPA Fault Address Register
ATS1HR	c7,4,c8,0	Addr Tran Stage 1 Hyp mode Read (WO)
ATS1HW	c7,4,c8,1	Addr Tran Stage 1 Hyp mode Write (WO)
TLBIALLHIS	c8,4,c3,0	Inv entry hyp unif TLB IS (WO)
TLBIMVAHIS	c8,4,c3,1	Inv hyp unif TLB entry by MVA IS (WO)
TLBIALLNSNHIS	c8,4,c3,4	Inv non-sec/hyp uni TLB IS (WO)
TLBIALLH	c8,4,c7,0	Inv hyp unified (WO)
TLBIMVAH	c8,4,c7,1	Inv hyp unif TLB by MVA (WO)
TLBIALLNSNH	c8,4,c7,4	Inv non-sec/hyp unif TLB (WO)
HMAIR0	c10,4,c2,0	Hyp Mem Attribute Indirection Reg 0
HMAIR1	c10,4,c2,1	Hyp Mem Attribute Indirection Reg 1
HAMAIR0	c10,4,c3,0	Hyp Aux Mem Attr Indirection Reg 0
HAMAIR1	c10,4,c3,1	Hyp Aux Mem Attr Indirection Reg 1
HVBAR	c12,4,c0,0	Hyp Vector Base Address Register
HTPIDR	c13,4,c0,2	Hyp Read/Write Thread ID
CNTHCTL	c14,4,c1,0	Timer PL2 Control Register
CNTHP_TVAL	c14,4,c2,0	PL2 Physical TimerValue Register
CNTHP_CTL	c14,4,c2,1	PL2 Physical Timer Control Register
CNTVOFF	c14,4	Virtual Offset Register
CNTHP_CVAL	c14,6	PL2 Physical Timer CompareValue Register

ARMv7-M System

Special Registers	
{I}{E}{A}PSR	Program Status Registers
XPSR	Alias for IEAPSR
MSP	Main Stack Pointer
PSP	Process Stack Pointer
PRIMASK	Exceptions Mask Register
BASEPRI	Base Priority Register
BASEPRI_MAX	Alias for BASEPRI that ignores writes of lower value
FAULTMASK	Raise exception priority to HardFloat
CONTROL	Special-Purpose Control Register

Program Status Register (xPSR)		
	0x000001ff	Exception number (RO)
IT	0x0600fc00	IT state bits
GE{3..0}	0x000f0000	SIMD Greater than or equal to (DSP extension only)
Q	0x08000000	Cumulative saturation bit
V	0x10000000	Overflow condition flag
C	0x20000000	Carry condition flag
Z	0x40000000	Zero condition flag
N	0x80000000	Negative condition flag

Vector Table	
0	Main SP register value at reset
1	Reset
2	NMI
3	HardFault
4	MemManage
5	BusFault
6	UsageFault
11	SVCall
12	DebugMonitor
14	PendSV
15	SysTick
16+{n}	External interrupt {n}

Address Map	
0x00000000-0x1fffffff	On-chip ROM or flash memory
0x20000000-0x3fffffff	On-chip SRAM
0x40000000-0x5fffffff	On-chip Peripherals
0x60000000-0x7fffffff	RAM with write-back cache
0x80000000-0x9fffffff	RAM with write-through cache
0xa0000000-0xbfffffff	Shared device space
0xc0000000-0xdfffffff	Non-shared device space
0xe0000000-0xffffffff	System segment

Interrupt Control and State Register (ICSR)		
VECTACTIVE	0x000001ff	Current executing exception (RO)
RETTOBASE	0x00000800	No active exceptions (except by IPSR) (RO)
VECTPENDING	0x001ff000	Highest pending and enabled exception (RO)
ISRPENDING	0x00400000	External interrupt is pending (RO)
ISRPREEMPT	0x00800000	Will service exception on debug exit (RO)
PENDSTCLR	0x02000000	Clear pending SysTick exception
PENDSTSET	0x04000000	Make SysTick exception pending
PENDSVCLR	0x08000000	Clear pending PendSV exception
PENDSVSET	0x10000000	Make PendSV exception pending
NMIPENDSET	0x80000000	Make NMI exception active

SysTick Control and Status Register (SYST_CSR)		
ENABLE	0x00000001	Counter is operating
TICKINT	0x00000002	SysTick exception on counter zero
CLKSOURCE	0x00000004	SysTick uses processor clock
COUNTFLAG	0x00010000	Timer has reached zero since last read (RO)

System Control Registers		
ICTR	0xe000e004	Interrupt Controller Type Register
ACTLR	0xe000e008	Auxiliary Control Register
ICSR	0xe000ed04	Interrupt Control and State Register
VTOR	0xe000ed08	Vector Table Offset Register
AIRCR	0xe000ed0c	App Interrupt and Reset Ctrl Reg
SCR	0xe000ed10	System Control Register
CCR	0xe000ed14	Configuration and Control Register
SHPR{1..3}	0xe000ed{18..20}	System Handler Priority Registers
SHCSR	0xe000ed24	System Handler Control and State Reg
CFSR	0xe000ed28	Configurable Fault Status Register
HFSR	0xe000ed2c	HardFault Status Register
DFSR	0xe000ed30	Debug Fault Status Register
MMFAR	0xe000ed34	MemManage Fault Address Registers
BFAR	0xe000ed38	BusFault Address Register
AFAR	0xe000ed3c	Auxiliary Fault Status Register
CPACR	0xe000ed88	Coprocessor Access Control Register

CPUID Registers (Read Only)		
CPUID	0xe000ed00	CPUID Base Register
ID_PFR{0..1}	0xe000ed4{0..4}	Processor Feature Registers
ID_DFR0	0xe000ed48	Debug Feature Register
ID_AFR0	0xe000ed4c	Auxiliary Feature Register
ID_MMFR{0..3}	0xe000ed5{0..c}	Memory Model Feature Registers
ID_ISAR{0..4}	0xe000ed{60..70}	Instruction Set Attribute Regs
ID_CLIDR	0xe000ed78	Cache Level ID Register
ID_CTR	0xe000ed7c	Cache Type Register
ID_CCSIDR	0xe000ed80	Cache Size ID Register
ID_CSSELR	0xe000ed84	Cache Size Selection Register

System Timer Registers		
SYST_CSR	0xe000e010	SysTick Control and Status Register
SYST_RVR	0xe000e014	SysTick Reload Value Register
SYST_CVR	0xe000e018	SysTick Current Value Register
SYST_CALIB	0xe000e01c	SysTick Calibration Value Register

External Interrupt Controller Registers		
NVIC_ISER{0..15}	0xe000e1{00..3c}	Interrupt Set-Enable Registers
NVIC_ICER{0..15}	0xe000e1{80..bc}	Interrupt Clear-Enable Registers
NVIC_ISPR{0..15}	0xe000e2{00..3c}	Interrupt Set-Pending Registers
NVIC_ICPR{0..15}	0xe000e2{80..bc}	Interrupt Clear-Pending Registers
NVIC_IABR{0..15}	0xe000e3{00..3c}	Interrupt Active Bit Registers
NVIC_IPR{0..123}	0xe000e{400..5ec}	Interrupt Priority Registers

Memory Protection Unit Registers		
MPU_TYPE	0xe000ed90	MPU Type Register (RO)
MPU_CTRL	0xe000ed94	MPU Control Register
MPU_RNR	0xe000ed98	MPU Region Number Register
MPU_RBAR	0xe000ed9c	MPU Region Base Address Register
MPU_RASR	0xe000eda0	MPU Region Attribute and Size Register

SW Trigger Interrupt Registers		
STIR	0xe000ef00	Software Triggered Interrupt Register (WO)
FPCCR	0xe000ef34	Floating Point Context Control Register
FPCAR	0xe000ef38	Floating Point Context Address Register
FPDSCR	0xe000ef3c	Floating Point Default Status Control Reg
MVFR{0..2}	0xe000ef4{0..8}	Medial and FP Feature Registers (RO)

Cache and Branch Predictior Maintenance (Write-Only)		
ICIALLU	0xe000ef50	I-cache invalidate all to PoU
ICIMVAU	0xe000ef58	I-cache invalidate by MVA to PoU
DCIMVAC	0xe000ef5c	D-cache invalidate by MVA to PoC
DCISW	0xe000ef60	D-cache invalidate by set-way
DCCMVAU	0xe000ef64	D-cache clean by MVA to PoU
DCCMVAC	0xe000ef68	D-cache clean by MVA to PoC
DCCSW	0xe000ef6c	D-cache clean by set-way
DCCIMVAC	0xe000ef70	D-cache clean and invalidate by MVA to PoC
DCCISW	0xe000ef74	D-cache clean and invalidate by set-way
BPIALL	0xe000ef78	Branch predictor invalidate all

Microcontroller-specific ID Registers		
PID{4..7}	0xe000efd{0..c}	Peripheral Identification Registers
PID{0..3}	0xe000efe{0..c}	Peripheral Identification Registers
CID{0..3}	0xe000eff{0..c}	Component Identification Registers