CE1921 ARMV4 BASIC INSTRUCTION SET ARCHITECTURE QUICK REFERENCE

INSTRUCTION	MODE	EXAMPLE	SYNTAX		REGISTER TRANSFER BEHAVIOR	TYPE	COND	OP	CMD	I	S
ADD	register	ADD R3, R4, R5	ADD	Rd, Rn, Rm	Rd ← Rn + Rm	D	E	0	4	0	0
ADD	immediate	ADD R3, R4, #8	ADD	Rd, Rn, Imm	Rd ← Rn + Ext. Imm.	D	E	0	4	1	0
AND	register	AND R8, R9, R10	AND	Rd, Rn, Rm	Rd ← Rn ● Rm	D	E	0	0	0	0
AND	immediate	AND R8, R9, #0xBE9	AND	Rd, Rn, Imm	Rd ← Rn ● Ext. Imm.	D	E	0	0	1	0
CMP	register	CMP R4, R5	CMP	Rn, Rm	Rn – Rm; STATUS ← CVNZ	D	E	0	10	0	1
CMP	immediate	CMP R8, #0x91	CMP	Rn, Imm	Rn – Ext. Imm; STATUS ← CVNZ	D	E	0	10	1	1
EOR	register	EOR R0, R10, R14	EOR	Rd, Rn, Rm	Rd ← Rn ⊕ Rm	D	E	0	1	0	0
EOR	immediate	EOR R0, R10, #0x1A	EOR	Rd, Rn, Imm	Rd ← Rn ⊕ Ext. Imm.	D	E	0	1	1	0
MOV	register	MOV R4, R5	MOV	Rd, Rm	Rd ← Rm	D	E	0	13	0	0
MOV	immediate	MOV R4, #38	MOV	Rd, Imm	Rd ← Ext. Imm.	D	E	0	13	1	0
MLA	register	MLA R4, R5, R6, R7	MLA	Rd, Rn, Rm, Ra	Rd ← (Rn x Rm) + Ra (low 32 bits)	D	E	0	1	0	0
MUL	register	MUL R9, R4, R5	MUL	Rd, Rn, Rm	Rd ← Rn x Rm (low 32 bits)	D	E	0	0	0	0
MVN	register	MVN R0, R1	MVN	Rd, Rm	Rd ← not Rm	D	E	0	15	0	0
MVN	immediate	MVN R0, #0xF4	MVN	Rd, Imm	Rd ← not Ext. Imm.	D	E	0	15	1	0
ORR	register	ORR R8, R9, R10	ORR	Rd, Rn, Rm	Rd ← Rn | Rm	D	E	0	12	0	0
ORR	immediate	ORR R1, R11, #15	ORR	Rd, Rn, Imm	Rd ← Rn | Ext. Imm.	D	E	0	12	1	0
SUB	register	SUB R3, R4, R5	SUB	Rd, Rn, Rm	Rd ← Rn – Rm	D	E	0	2	0	0
SUB	immediate	SUB R3, R4, #8	SUB	Rd, Rn, Imm	Rd ← Rn – Ext. Imm.	D	E	0	2	1	0
LDR	register	LDR R4, [R5]	LDR	Rd, [Rn]	Rd ← MEM[Rn + Ext. Imm]	M	E	1	PUBWL	1	
LDR	immediate	LDR R4, [R5, #8]	LDR	Rd, [Rn, Imm]	Rd ← MEM[Rn + Ext. Imm]	M	E	1	PUBWL	1	
STR	register	STR R1, [R6]	STR	Rd, [Rn]	MEM[Rn + Ext. Imm] ← Rd	M	E	1	PUBWL	0	
STR	immediate	STR R1,[R6,#0x20]	STR	Rd, [Rn, Imm]	MEM[Rn + Ext. Imm] ← Rd	M	E	1	PUBWL	0	
B	unconditional	B LOOP	B	labeled line	PC ← BranchAddr	B	E	2	LXXXX	1	
BL	unconditional	BL GETDATA	BL	labeled line	LR ← PC + 4; PC ←BranchAddr	B	E	2	LXXXX	1	
BEQ	conditional	BEQ LOOP	BEQ	labeled line	PC ← BranchAddr if Z=1 else PC+4	B	0	2	LXXXX	1	
BNE	conditional	BNE LOOP	BNE	labeled line	PC ← BranchAddr if Z=0 else PC+4	B	1	2	LXXXX	1	

- **TYPE**: ARM instructions can be categorized as data processing, memory, or branch. The **opcode (OP)** field of the machine instruction encodes the type.
- Branches use sign extension to form **BranchAddr = PC + 8 + (SignExtImm << 2)**. Arithmetic and load/store instructions only use positive constants and thus zero extend.
- **PUBWL** are bits of information encoded into the command field for memory instructions LDR and STR. **See the tables that follows on the next page.**
- **L** is the only bit that is encoded into the command field for branch instructions. The remaining bits are part of the 24-bit immediate. L determines if it is a normal branch or a branch-and-link subroutine call. **See the table that follows on the next page.**

Dr. Russ Meier, Milwaukee School of Engineering, Last Update: 2020

CE1921 ARMV4 BASIC INSTRUCTION SET ARCHITECTURE QUICK REFERENCE

LOAD/STORE FUNCTION FIELD BITS

B	L	INSTRUCTION
0	0	STR
0	1	LDR
1	0	STRB
1	1	LDRB

P	W	INDEX MODE
0	0	Post-Index
0	1	Not Supported
1	0	Offset
1	1	Pre-Index

U	OFFSET ARITHMETIC
0	Subtract offset from base
1	Add offset to base

SHIFT ENCODING

INSTRUCTION	SH TYPE	BEHAVIOR
LSL	0	Logical Shift Left
LSR	1	Logical Shift Right
ASR	2	Arithmetic Shift Right
ROR	3	Rotate Right

REGISTER FILE – EACH REGISTER IS 32-BITS WIDE

	R0	R1-R3	R4-R11	R12	R13	R14	R15
USE AS	Func. Argument 1	Func. Arguments 2-4	Saved Variables	Temporary	SP	LR	PC
USE AS	Func. Return Value						

BRANCH FUNCTION FIELD BITS

L	INSTRUCTION
0	B, BEQ, BNE
1	BL

CONDITIONAL EXECUTION SUFFIX: COND FIELD ENCODING WRITTEN IN DECIMAL

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
unused	AL	LE	GT	LT	GE	LS	HI	VC	VS	PL	MI	CC/LO	CS/HS	NE	EQ

MULTIPLY CMD FIELD WRITTEN IN DECIMAL

7	6	5	4	3	2	1	0
SMLAL	SMULL	UMLAL	UMULL	unused	unused	MLA	MUL

CURRENT PROGRAM STATUS REGISTER (CPSR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N	Z	C	V																					I	F	T	MODE				

CPSR T CONTROL BIT

0	1
ARM	THUMB

CPSR I CONTROL BIT

0	1
interrupts on	interrupts off

CPSR F CONTROL BIT

0	1
fast interrupts on	fast interrupts off

CPSR MODE CONTROL FIELD WRITTEN IN BINARY

10000	10001	10010	10011	10111
user	fast interrupt	interrupt	supervisor (OS)	abort

INSTRUCTION BINARY NUMBER ENCODINGS

DATA PROCESSING																															
MODE	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Register	COND				OPCODE		I	CMD				S	RN				RD				SHAMT					SHTYPE		0	RM			
Shifted Register	COND				OPCODE		I	CMD				S	RN				RD				RS				0	SHTYPE		1	RM			
Immediate	COND				OPCODE		I	CMD				S	RN				RD				ROTATE				IMMEDIATE							
LOAD-STORE																																
MODE	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Register	COND				OPCODE		$\bar{I}$	P	U	B	W	L	RN				RD				SHAMT					SHTYPE		1	RM			
Immediate	COND				OPCODE		$\bar{I}$	P	U	B	W	L	RN				RD				IMMEDIATE											
BRANCH																																
MODE	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Immediate	COND				OPCODE		1	L	IMMEDIATE																							
MULTIPLY																																
MODE	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Register	COND				OPCODE		0	0	CMD			S	RD				RA				RM				1	0	0	1	RN			

Dr. Russ Meier, Milwaukee School of Engineering, Last Update: 2020