Appendix 1 - Summary of VisUAL ARM Instruction Set and Other Useful Information

Summary	Opcode	Syntax
Move	MOV	MOV{S}{cond} dest, op1 {, SHIFT_op #expression}
Move Negated	MVN	MVN{S}{cond} dest, op1 {, SHIFT_op #expression}
Address Load	ADR	ADR{S}{cond} dest, expression
LDR Psuedo-Instruction	LDR	LDR{S}{cond} dest, =expression
Add	ADD	ADD{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Add with Carry	ADC	ADC{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Subtract	SUB	SUB{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Subtract with Carry	SBC	SBC{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Reverse Subtract	RSB	RSB{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Reverse Subtract with Carry	RSC	RSC{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Bitwise And	AND	AND{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Bitwise Exclusive Or	EOR	EOR{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Bitwise Clear	BIC	BIC{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Bitwise Or	ORR	ORR{S}{cond} dest, op1, op2 {, SHIFT_op #expression}
Logical Shift Left	LSL	LSL{S}{cond} dest, op1, op2
Logical Shift Right	LSR	LSR{S}{cond} dest, op1, op2
Arithmetic Shift Right	ASR	ASR{S}{cond} dest, op1, op2
Rotate Right	ROR	ROR{S}{cond} dest, op1, op2
Rotate Right and Extend	RRX	RRX{S}{cond} op1, op2
Compare	CMP	CMP{cond} op1, op2 {, SHIFT_op #expression}
Compare Negated	CMN	CMN{cond} op1, op2 {, SHIFT_op #expression}
Test Bit(s) Set	TST	TST{cond} op1, op2 {, SHIFT_op #expression}
Test Equals	TEQ	TEQ{cond} op1, op2 {, SHIFT_op #expression}
Load Register	LDR	LDR{B}{cond} dest, [source {, OFFSET}] Offset addressing LDR{B}{cond} dest, [source, OFFSET]! Pre-indexed addressing LDR{B}{cond} dest, [source], OFFSET Post-indexed addressing
Store Register	STR	STR{B}{cond} source, [dest {, OFFSET}] Offset addressing STR{B}{cond} source, [dest, OFFSET]! Pre-indexed addressing STR{B}{cond} source, [dest], OFFSET Post-indexed addressing
Load Multiple Registers	LDM[dir]	LDM[dir]{cond} source, {list of registers}
Store Multiple Registers	STM[dir]	STM[dir]{cond} dest, {list of registers}
Branch	B	B{cond} target
Branch with Link	BL	BL{cond} target
Declare Word(s) in Memory	DCD	name DCD value_1, value_2, ... value_N
Declare Constant	EQU	name equ expression
Declare Empty Word(s) in	FILL	{name} FILL N
Stop Emulation	END	END{cond}

Table 1. Condition code suffixes

Suffix	Flags	Meaning
EQ	Z = 1	Equal
NE	Z = 0	Not equal
CS or HS	C = 1	Higher or same, unsigned
CC or LO	C = 0	Lower, unsigned
MI	N = 1	Negative
PL	N = 0	Positive or zero
VS	V = 1	Overflow
VS	V = 1	Overflow
VC	V = 0	No overflow
HI	C = 1 and Z = 0	Higher, unsigned
LS	C = 0 or Z = 1	Lower or same, unsigned
GE	N = V	Greater than or equal, signed
LT	N != V	Less than, signed
GT	Z = 0 and N = V	Greater than, signed
LE	Z = 1 and N != V	Less than or equal, signed
AL	Can have any value	Always. This is the default when no suffix is specified.

Table 2. ASCII Table

MS / LS	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	'	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	
F	SI	US	/	?	O	_	o	DEL