Macro Assembler

An Assembler for
Arm Microcontrollers

User Guide & Reference Manual

Document: UM20006
Software Version: 2.10 Technology Preview
Revision: 0
Date: October 14, 2019

SEGGER

A product of SEGGER Microcontroller GmbH

www.segger.com

Disclaimer

Copyright notice

Trademarks

Contact address

SEGGER Microcontroller GmbH

Ecolab-Allee 5
D-40789 Monheim am Rhein

Germany

Tel.	+49 2173-99312-0
Fax.	+49 2173-99312-28
E-mail:	`support@segger.com`[*]
Internet:	*www.segger.com*

[*]By sending us an email your (personal) data will automatically be processed. For further information please refer to our privacy policy which is available at https://www.segger.com/legal/privacy-policy/.

Manual versions

This manual describes the current software version. If you find an error in the manual or a problem in the software, please report it to us and we will try to assist you as soon as possible.

Contact us for further information on topics or functions that are not yet documented.

Print date: October 14, 2019

Software	Revision	Date	By	Description
2.10	0	180514	PC	Initial release.
1.10	0	180427	PC	T32 + A32 internal release.
1.00	0	180410	PC	T32 internal release.

About this document

Assumptions

This document assumes that you already have a solid knowledge of the following:

- The software tools used for building your application (assembler, linker, C compiler).
- The C programming language.
- The target processor.
- DOS command line.

If you feel that your knowledge of C is not sufficient, we recommend *The C Programming Language* by Kernighan and Richie (ISBN 0--13--1103628), which describes the standard in C programming and, in newer editions, also covers the ANSI C standard.

How to use this manual

This manual describes all the functions and macros that the product offers. It assumes you have a working knowledge of the C language. Knowledge of assembly programming is not required.

Typographic conventions for syntax

This manual uses the following typographic conventions:

Style	Used for
Body	Body text.
`Parameter`	Parameters in API functions.
`Sample`	Sample code in program examples.
`Sample comment`	Comments in program examples.
`User Input`	Text entered at the keyboard by a user in a session transcript.
`Secret Input`	Text entered at the keyboard by a user, but not echoed (e.g. password entry), in a session transcript.
Reference	Reference to chapters, sections, tables and figures or other documents.
Emphasis	Very important sections.
SEGGER home page	A hyperlink to an external document or web site.

Table of contents

Chapter 1

About the assembler

1.1 Introduction

This section presents an overview of the SEGGER Assembler and its capabilities.

1.1.1 What is the SEGGER Assembler?

The SEGGER Assembler is a fast assembler that processes Cortex-M assembly language source files. It is designed to be very flexible, yet simple to use.

The assembler accepts GNU, Arm, and IAR syntax source files and will assemble them to Arm ELF object files that can be linked by a standard Cortex-M linker.

1.1.2 Assembler features

The SEGGER Assembler has the following features:

- Highly efficient and very fast to assemble.
- Supports GNU, Arm, and IAR syntax.

Chapter 2

Expressions

GNU	IAR	ARM
`()`, prefix `+`, `-`, `!`, `~`	`()`, prefix `+`, `-`, `!`, `~`, **LOW**, **HIGH**, **BYTE1**, **BYTE2**, **BYTE3**, **BYTE4**, **LWRD**, **HWRD**, **DATE**, **SFB**, **SFE**, **SIZEOF**	`()`, prefix `+`, `-`, `:NOT:`, `:LNOT:`
`*`, `/`, `%`, `<<`, `>>`	`*`, `/`, `%`	`*`, `/`, `%`, `:MOD:`
`|`, `&`, `^`, `!`	`+`, `-`	`<<`, `>>`, `:SHL:`, `:SHR:`, `:ROL:`, `:ROR:`
`+`, `-`, `=`, `<>`, `≠`, `<`, `>`, `≤`, `≥`	`<<`, `>>`	`+`, `-`, `&`, `^`, `:AND:`, `:OR:`, `:EOR:`
`&&`, `||`	`&&`, `&`	`=`, `<>`, `≠`, `/=`, `<`, `>`, `≤`, `≥`
	`||`, `|`, `^`, **XOR**	`&&`, `||`, `:LAND:`, `:LOR:`, `:LEOR:`
	`=`, `=`, `<>`, `≠`, `<`, `>`, `≤`, `≥`, **UGT**, **ULT**	

Chapter 3

GNU syntax

3.1 Source file options

3.1.1 .incbin

Include binary file.

Syntax

`.incbin` “*filename*”

Description

Searches for the binary file *filename* using include paths. Inserts the binary file into the current section as data with alignment 1.

3.2 Section options

3.2.1 .data

Switch to data section

Syntax

```
.data
```

Description

Switches to the `.data` section.

3.2.2 .bss

Switch to bss section

Syntax

```
.bss
```

Description

Switches to the `.bss` section.

3.2.3 .text

Switch to text section

Syntax

```
.text
```

Description

Switches to the `.text` section.

3.3 Symbol options

3.3.1 .local

Set local symbol binding.

Syntax

`.local` *name*, *name*…

Description

Defines all symbols in the symbol list to have local binding so that they are not externally visible.

Chapter 4

Arm syntax

Chapter 5

IAR syntax

Chapter 6

Instructions

6.1 Integer instructions

6.1.1 ADC

Thumb syntax (16-bit, v4T)

```
ADCS    Rd, Rn                      [1]
```

Thumb syntax (32-bit, v6T2)

```
ADC     Rd, Rn, #const              [2]
ADC     Rd, Rn, Rm, shift           [3]
ADCS    Rd, Rn, #const              [2]
ADCS    Rd, Rn, Rm, shift           [3]
```

Arm syntax (v4T)

```
ADC     Rd, Rn, #const              [4]
ADC     Rd, Rn, Rm, shift           [5]
ADCS    Rd, Rn, #const              [4]
ADCS    Rd, Rn, Rm, shift           [5]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**; ordering *Rd*, *Rn*, *Rd* is also permitted
[2] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.2 ADD

Thumb syntax (16-bit, v4T)

ADD	*Rd*, *Rn*	[1]
ADD	**SP**, **SP**, **#***0…508*	[2]
ADD	*Rd*, **SP**, **#***0…1020*	[3]
ADD	*Rd*, **PC**, **#***0…1020*	[3]
ADDS	*Rd*, *Rn*, *Rm*	[4]
ADDS	*Rd*, **#***0…255*	[5]
ADDS	*Rd*, *Rn*, **#***0…7*	[6]

Thumb syntax (32-bit, v6T2)

ADD	*Rd*, *Rn*, **#***0…4095*	
ADD	*Rd*, *Rn*, **#***const*	[7]
ADD	*Rd*, *Rn*, *Rm*, *shift*	[8]
ADDS	*Rd*, *Rn*, **#***const*	[9]
ADDS	*Rd*, *Rn*, *Rm*, *shift*	[8]

Arm syntax (v4T)

ADD	*Rd*, *Rn*, **#***const*	[10]
ADD	*Rd*, *Rn*, *Rm*, *shift*	[11]
ADD	*Rd*, **SP**, *Rm*, *shift*	[12]
ADDS	*Rd*, *Rn*, **#***const*	[10]
ADDS	*Rd*, *Rn*, *Rm*, *shift*	[11]
ADDS	*Rd*, **SP**, *Rm*, *shift*	[12]

Notes

[1] ordering *Rd*, *Rn*, *Rd* is also permitted
[2] offset a multiple of 4
[3] *Rd* must be **R0**…**R7**; offset a multiple of 4
[4] *Rd*, *Rn*, *Rm* must be **R0**…**R7**
[5] *Rd* must be **R0**…**R7**
[6] *Rd*, *Rn* must be **R0**…**R7**
[7] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*XXXXXX, *$xx* **LSL** *0…24*
[8] *Rm* ≠ **PC**; *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[9] *Rd* ≠ **PC**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*XXXXXX, *$xx* **LSL** *0…24*
[10] *const* is *$xx* **ROR** 2n
[11] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**
[12] *Rd*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.3 ADDW

Thumb syntax (16-bit, v6T2)

ADDW *Rd*, **PC**, **#***0…1020* [1]

Thumb syntax (32-bit, v6T2)

ADDW *Rd*, *Rn*, **#***0…4095*

Notes

[1] *Rd* must be **R0**…**R7**; offset a multiple of 4

6.1.4 AND

Thumb syntax (16-bit, v4T)

```
ANDS     Rd, Rn                        [1]
```

Thumb syntax (32-bit, v6T2)

```
AND      Rd, Rn, #const                [2]
AND      Rd, Rn, Rm, shift             [3]
ANDS     Rd, Rn, #const                [2]
ANDS     Rd, Rn, Rm, shift             [3]
```

Arm syntax (v4T)

```
AND      Rd, Rn, #const                [4]
AND      Rd, Rn, Rm, shift             [5]
ANDS     Rd, Rn, #const                [4]
ANDS     Rd, Rn, Rm, shift             [5]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**; ordering *Rd*, *Rn*, *Rd* is also permitted
[2] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.5 ASR

Thumb syntax (16-bit, v4T)

ASRS	*Rd*, *Rn*	[1]
ASRS	*Rd*, *Rn*, **#***1…32*	[1]

Thumb syntax (32-bit, v6T2)

ASR	*Rd*, *Rn*, *Rm*	[2]
ASR	*Rd*, *Rn*, **#***1…32*	[3]
ASRS	*Rd*, *Rn*, *Rm*	[2]
ASRS	*Rd*, *Rn*, **#***1…32*	[3]

Arm syntax (v6)

ASR	*Rd*, *Rn*, *Rm*	[2]
ASR	*Rd*, *Rn*, **#***1…32*	[3]
ASRS	*Rd*, *Rn*, *Rm*	[2]
ASRS	*Rd*, *Rn*, **#***1…32*	[3]

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**
[3] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.6 B

Thumb syntax (16-bit, v4T)

```
B       Label
```

Thumb syntax (32-bit, v6T2)

```
B       Label
```

Arm syntax (v4T)

```
B       Label
```

6.1.7 BCC

Thumb syntax (16-bit, v4T)

BCC *Label*

Thumb syntax (32-bit, v6T2)

BCC *Label*

6.1.8 BCS

Thumb syntax (16-bit, v4T)

BCS *Label*

Thumb syntax (32-bit, v6T2)

BCS *Label*

6.1.9 BEQ

Thumb syntax (16-bit, v4T)

```
BEQ      Label
```

Thumb syntax (32-bit, v6T2)

```
BEQ      Label
```

6.1.10 BFC

Thumb syntax (32-bit, v6T2)

```
BFC     Rd, #lsb, #width                  [1]
```

Arm syntax (v6T2)

```
BFC     Rd, #lsb, #width                  [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.11 BFI

Thumb syntax (32-bit, v6T2)

```
BFI     Rd, Rn, #lsb, #width          [1]
```

Arm syntax (v6T2)

```
BFI     Rd, Rn, #lsb, #width          [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.12 BGE

Thumb syntax (16-bit, v4T)

```
BGE       Label
```

Thumb syntax (32-bit, v6T2)

```
BGE       Label
```

6.1.13 BGT

Thumb syntax (16-bit, v4T)

```
BGT     Label
```

Thumb syntax (32-bit, v6T2)

```
BGT     Label
```

6.1.14 BHI

Thumb syntax (16-bit, v4T)

```
BHI     Label
```

Thumb syntax (32-bit, v6T2)

```
BHI     Label
```

6.1.15 BHS

Thumb syntax (16-bit, v4T)

```
BHS        Label
```

Thumb syntax (32-bit, v6T2)

```
BHS        Label
```

6.1.16 BIC

Thumb syntax (16-bit, v4T)

BICS	*Rd*, *Rn*	[1]

Thumb syntax (32-bit, v6T2)

BIC	*Rd*, *Rn*, **#***const*	[2]
BIC	*Rd*, *Rn*, *Rm*, *shift*	[3]
BICS	*Rd*, *Rn*, **#***const*	[2]
BICS	*Rd*, *Rn*, *Rm*, *shift*	[3]

Arm syntax (v4T)

BIC	*Rd*, *Rn*, **#***const*	[4]
BIC	*Rd*, *Rn*, *Rm*, *shift*	[5]
BICS	*Rd*, *Rn*, **#***const*	[4]
BICS	*Rd*, *Rn*, *Rm*, *shift*	[5]

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.17 BKPT

Thumb syntax (16-bit, v5T)

BKPT **#***0…255*

Arm syntax (v5T)

BKPT *Rd*, **#***0…65535*

6.1.18 BL

Thumb syntax (32-bit, v4T)

```
BL          Label
```

Arm syntax (v4T)

```
BL          Label
```

6.1.19 BLE

Thumb syntax (16-bit, v4T)

```
BLE       Label
```

Thumb syntax (32-bit, v6T2)

```
BLE       Label
```

6.1.20 BLO

Thumb syntax (16-bit, v4T)

```
BLO         Label
```

Thumb syntax (32-bit, v6T2)

```
BLO         Label
```

6.1.21 BLS

Thumb syntax (16-bit, v4T)

BLS *Label*

Thumb syntax (32-bit, v6T2)

BLS *Label*

6.1.22 BLT

Thumb syntax (16-bit, v4T)

```
BLT       Label
```

Thumb syntax (32-bit, v6T2)

```
BLT       Label
```

6.1.23 BLX

Thumb syntax (16-bit, v5T)

BLX	*Rd*	[1]

Thumb syntax (32-bit, v4T)

BLX	*Label*

Arm syntax (v4T)

BLX	*Rd*	[2]
BLX	*Label*	

Notes

[1] *Rd* ≠ **PC**; *Rd* ≠ **SP**
[2] *Rd* ≠ **PC**

6.1.24 BMI

Thumb syntax (16-bit, v4T)

```
BMI     Label
```

Thumb syntax (32-bit, v6T2)

```
BMI     Label
```

6.1.25 BNE

Thumb syntax (16-bit, v4T)

```
BNE     Label
```

Thumb syntax (32-bit, v6T2)

```
BNE     Label
```

6.1.26 BPL

Thumb syntax (16-bit, v4T)

```
BPL      Label
```

Thumb syntax (32-bit, v6T2)

```
BPL      Label
```

6.1.27 BVC

Thumb syntax (16-bit, v4T)

BVC *Label*

Thumb syntax (32-bit, v6T2)

BVC *Label*

6.1.28 BVS

Thumb syntax (16-bit, v4T)

BVS *Label*

Thumb syntax (32-bit, v6T2)

BVS *Label*

6.1.29 BX

Thumb syntax (16-bit, v4T)

BX *Rd*

Arm syntax (v6)

BX *Rd* [1]

Notes

[1] *Rd* ≠ **SP**

6.1.30 BXJ

Thumb syntax (32-bit, v6J)

```
BXJ    Rd
```

Arm syntax (v6J)

```
BXJ    Rd
```

6.1.31 CBNZ

Thumb syntax (16-bit, v6T2)

```
CBNZ    Rd, Label
```

6.1.32 CBZ

Thumb syntax (16-bit, v6T2)

```
CBZ     Rd, Label
```

6.1.33 CDP

Thumb syntax (32-bit, v6T2)

CDP	*** Unknown ***

Arm syntax (v6T2)

CDP	*** Unknown ***

6.1.34 CDP2

Thumb syntax (32-bit, v6T2)

`CDP2` *** Unknown ***

Arm syntax (v6T2)

`CDP2` *** Unknown ***

6.1.35 CLREX

Thumb syntax (32-bit, v6T2)

```
CLREX
```

Arm syntax (v6K)

```
CLREX
```

6.1.36 CLZ

Thumb syntax (32-bit, v6T2)

```
CLZ     Rd, Rn                          [1]
```

Arm syntax (v6)

```
CLZ     Rd, Rn                          [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.37 CMN

Thumb syntax (16-bit, v4T)

```
CMN     Rn, Rm                          [1]
```

Thumb syntax (32-bit, v6T2)

```
CMN     Rn, #const                      [2]
CMN     Rn, Rm, shift                   [3]
```

Arm syntax (v4T)

```
CMN     Rn, #const                      [4]
CMN     Rn, Rm, shift                   [5]
```

Notes

[1] *Rn*, *Rm* must be **R0**…**R7**
[2] *Rn* ≠ **PC**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0…24*
[3] *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[4] *Rn* ≠ **PC**; *const* is *$xx* **ROR** 2n
[5] *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.38 CMP

Thumb syntax (16-bit, v4T)

```
CMP     Rn, Rm                     [1]
CMP     Rn, #0…255                 [2]
```

Thumb syntax (32-bit, v6T2)

```
CMP     Rn, #const                 [3]
CMP     Rn, Rm, shift              [4]
```

Arm syntax (v4T)

```
CMP     Rn, #const                 [5]
CMP     Rn, Rm, shift              [6]
```

Notes

[1] *Rn* ≠ **PC**
[2] *Rn* must be **R0**…**R7**
[3] *Rn* ≠ **PC**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0…24*
[4] *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[5] *Rn* ≠ **PC**; *const* is *$xx* **ROR** 2n
[6] *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.39 CPS

Thumb syntax (32-bit, v)

```
CPS      #0...31                    [1]
```

Arm syntax (v)

```
CPS      #0...31                    [1]
```

Notes

[1] not permitted in an IT block

6.1.40 CPSID

Thumb syntax (16-bit, v6)

```
CPSID   flags                          [1]
```

Thumb syntax (32-bit, v)

```
CPSID   flags                          [1]
CPSID   flags, #0…31                   [1]
```

Arm syntax (v6)

```
CPSID   flags                          [2]
CPSID   flags, #0…31                   [2]
```

Notes

[1] *flags* is any combination of 'A', 'I', and 'F'; not permitted in an IT block
[2] *flags* is any combination of 'A', 'I', and 'F'

6.1.41 CPSIE

Thumb syntax (16-bit, v6)

```
CPSIE   flags                          [1]
```

Thumb syntax (32-bit, v)

```
CPSIE   flags                          [1]
CPSIE   flags, #0…31                   [1]
```

Arm syntax (v6)

```
CPSIE   flags                          [2]
CPSIE   flags, #0…31                   [2]
```

Notes

[1] *flags* is any combination of 'A', 'I', and 'F'; not permitted in an IT block
[2] *flags* is any combination of 'A', 'I', and 'F'

6.1.42 CPY

Thumb syntax (16-bit, v6)

```
CPY     Rd, Rn
```

Arm syntax (v4T)

```
CPY     Rd, Rn
```

6.1.43 CRC32B

Thumb syntax (32-bit, v)

CRC32B *Rd*, *Rn*, *Rm*

Arm syntax (v)

CRC32B *Rd*, *Rn*, *Rm*

6.1.44 CRC32CB

Thumb syntax (32-bit, v)

CRC32CB *Rd*, *Rn*, *Rm*

Arm syntax (v)

CRC32CB *Rd*, *Rn*, *Rm*

6.1.45 CRC32CH

Thumb syntax (32-bit, v)

CRC32CH *Rd*, *Rn*, *Rm*

Arm syntax (v)

CRC32CH *Rd*, *Rn*, *Rm*

6.1.46 CRC32CW

Thumb syntax (32-bit, v)

`CRC32CW` *`Rd, Rn, Rm`*

Arm syntax (v)

`CRC32CW` *`Rd, Rn, Rm`*

6.1.47 CRC32H

Thumb syntax (32-bit, v)

CRC32H *Rd*, *Rn*, *Rm*

Arm syntax (v)

CRC32H *Rd*, *Rn*, *Rm*

6.1.48 CRC32W

Thumb syntax (32-bit, v)

`CRC32W` *`Rd, Rn, Rm`*

Arm syntax (v)

`CRC32W` *`Rd, Rn, Rm`*

6.1.49 DBG

Thumb syntax (32-bit, v6T2)

DBG **#**_0…15_

Arm syntax (v6)

DBG **#**_0…15_

6.1.50 DCPS1

Thumb syntax (32-bit, v)

```
DCPS1
```

6.1.51 DCPS2

Thumb syntax (32-bit, v)

```
DCPS2
```

6.1.52 DCPS3

Thumb syntax (32-bit, v)

```
DCPS3
```

6.1.53 DMB

Thumb syntax (32-bit, v6-M)

```
DMB      #0…15
```

Arm syntax (v)

```
DMB      #0…15
```

6.1.54 DSB

Thumb syntax (32-bit, v6-M)

```
DSB        #0...15
```

Arm syntax (v)

```
DSB        #0...15
```

6.1.55 EOR

Thumb syntax (16-bit, v4T)

EORS	*Rd*, *Rn*	[1]

Thumb syntax (32-bit, v6T2)

EOR	*Rd*, *Rn*, #*const*	[2]
EOR	*Rd*, *Rn*, *Rm*, *shift*	[3]
EORS	*Rd*, *Rn*, #*const*	[2]
EORS	*Rd*, *Rn*, *Rm*, *shift*	[3]

Arm syntax (v4T)

EOR	*Rd*, *Rn*, #*const*	[4]
EOR	*Rd*, *Rn*, *Rm*, *shift*	[5]
EORS	*Rd*, *Rn*, #*const*	[4]
EORS	*Rd*, *Rn*, *Rm*, *shift*	[5]

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**; ordering *Rd*, *Rn*, *Rd* is also permitted
[2] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.56 ERET

Thumb syntax (32-bit, v)

```
ERET
```

Arm syntax (v)

```
ERET
```

6.1.57 ESB

Thumb syntax (32-bit, v)

```
ESB
```

Arm syntax (v)

```
ESB
```

6.1.58 HLT

Thumb syntax (16-bit, v)

```
HLT       #0…63
```

Arm syntax (v)

```
HLT       Rd, #0…65535
```

6.1.59 HVC

Thumb syntax (32-bit, v)

```
HVC      #0…65535
```

Arm syntax (v)

```
HVC      Rd, #0…65535
```

6.1.60 ISB

Thumb syntax (32-bit, v6-M)

```
ISB      #0…15
```

Arm syntax (v)

```
ISB      #0…15
```

6.1.61 ITEQ

Thumb syntax (16-bit, v6T2)

```
ITEQ
```

6.1.62 LDA

Thumb syntax (32-bit, v)

LDA *Rd*, [*Rn*] [1]

Arm syntax (v)

LDA *Rd*, [*Rn*] [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.63 LDAB

Thumb syntax (32-bit, v)

LDAB	*Rd*, [*Rn*]	[1]

Arm syntax (v)

LDAB	*Rd*, [*Rn*]	[1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.64 LDAEX

Thumb syntax (32-bit, v)

LDAEX *Rd*, **[***Rn***]** [1]

Arm syntax (v)

LDAEX *Rd*, **[***Rn***]** [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.65 LDAEXB

Thumb syntax (32-bit, v)

LDAEXB *Rd*, [*Rn*] [1]

Arm syntax (v)

LDAEXB *Rd*, [*Rn*] [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.66 LDAEXD

Thumb syntax (32-bit, v)

LDAEXD *Rd1*, *Rd*2, [*Rn*] [1]

Arm syntax (v)

LDAEXD *Rd1*, *Rd*2, [*Rn*] [2]

Notes

[1] *Rd1*, *Rd*2, *Rn* ≠ **PC**; *Rd1* ≠ *Rd*2
[2] *Rd1*, *Rd*2, *Rn* ≠ **PC**; *Rd1* ≠ *Rd*2; *Rd1* must be an even-numbered register; *Rd*2 must be the following odd-numbered register

6.1.67 LDAEXH

Thumb syntax (32-bit, v)

LDAEXH *Rd*, [*Rn*] [1]

Arm syntax (v)

LDAEXH *Rd*, [*Rn*] [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.68 LDAH

Thumb syntax (32-bit, v)

LDAH *Rd*, [*Rn*] [1]

Arm syntax (v)

LDAH *Rd*, [*Rn*] [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.69 LDM

Thumb syntax (16-bit, v4T)

```
LDM     SP!, {Rn, Rm…}
LDM     Rd, {Rn, Rm…}                       [1]
LDM     Rd, {Rn, Rm…}                       [2]
LDM     Rd!, {Rn, Rm…}                      [1]
```

Thumb syntax (32-bit, v6T2)

```
LDM     Rd, {Rn, Rm…}                       [2]
LDM     Rd!, {Rn, Rm…}                      [2]
```

Arm syntax (v4T)

```
LDM     Rd, {Rn, Rm…}                       [2]
LDM     Rd!, {Rn, Rm…}                      [2]
```

Notes

[1] *Rd* must be **R0**…**R7**; *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**

6.1.70 LDMDA

Thumb syntax (16-bit, v4T)

LDMDA *Rd*, {*Rn*, *Rm*…}

Thumb syntax (32-bit, v6T2)

LDMDA *Rd*, {*Rn*, *Rm*…}
LDMDA *Rd*!, {*Rn*, *Rm*…} [1]

Arm syntax (v4T)

LDMDA *Rd*, {*Rn*, *Rm*…} [1]
LDMDA *Rd*!, {*Rn*, *Rm*…} [1]

Notes

[1] *Rd* ≠ **PC**

6.1.71 LDMDB

Thumb syntax (32-bit, v6T2)

```
LDMDB   Rd, {Rn, Rm...}                   [1]
LDMDB   Rd!, {Rn, Rm...}                  [1]
```

Arm syntax (v4T)

```
LDMDB   Rd, {Rn, Rm...}                   [1]
LDMDB   Rd!, {Rn, Rm...}                  [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.72 LDMEA

Thumb syntax (32-bit, v6T2)

LDMEA	*Rd*, {*Rn*, *Rm*...}	[1]
LDMEA	*Rd*!, {*Rn*, *Rm*...}	[1]

Arm syntax (v4T)

LDMEA	*Rd*, {*Rn*, *Rm*...}	[1]
LDMEA	*Rd*!, {*Rn*, *Rm*...}	[1]

Notes

[1] *Rd* ≠ **PC**

6.1.73 LDMED

Thumb syntax (16-bit, v4T)

```
LDMED   Rd, {Rn, Rm...}                    [1]
```

Thumb syntax (32-bit, v6T2)

```
LDMED   Rd, {Rn, Rm...}
LDMED   Rd!, {Rn, Rm...}                   [1]
```

Arm syntax (v4T)

```
LDMED   Rd, {Rn, Rm...}                    [1]
LDMED   Rd!, {Rn, Rm...}                   [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.74 LDMFA

Thumb syntax (16-bit, v4T)

```
LDMFA   Rd, {Rn, Rm...}
```

Thumb syntax (32-bit, v6T2)

```
LDMFA   Rd, {Rn, Rm...}
LDMFA   Rd!, {Rn, Rm...}                    [1]
```

Arm syntax (v4T)

```
LDMFA   Rd, {Rn, Rm...}                     [1]
LDMFA   Rd!, {Rn, Rm...}                    [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.75 LDMFD

Thumb syntax (16-bit, v4T)

```
LDMFD   SP!, {Rn, Rm…}
LDMFD   Rd, {Rn, Rm…}                    [1]
LDMFD   Rd, {Rn, Rm…}                    [2]
LDMFD   Rd!, {Rn, Rm…}                   [1]
```

Thumb syntax (32-bit, v6T2)

```
LDMFD   Rd, {Rn, Rm…}                    [2]
LDMFD   Rd!, {Rn, Rm…}                   [2]
```

Arm syntax (v4T)

```
LDMFD   Rd, {Rn, Rm…}                    [2]
LDMFD   Rd!, {Rn, Rm…}                   [2]
```

Notes

[1] *Rd* must be **R0**…**R7**; *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**

6.1.76 LDMIA

Thumb syntax (16-bit, v4T)

```
LDMIA   SP!, {Rn, Rm…}
LDMIA   Rd, {Rn, Rm…}                [1]
LDMIA   Rd, {Rn, Rm…}                [2]
LDMIA   Rd!, {Rn, Rm…}               [1]
```

Thumb syntax (32-bit, v6T2)

```
LDMIA   Rd, {Rn, Rm…}                [2]
LDMIA   Rd!, {Rn, Rm…}               [2]
```

Arm syntax (v4T)

```
LDMIA   Rd, {Rn, Rm…}                [2]
LDMIA   Rd!, {Rn, Rm…}               [2]
```

Notes

[1] *Rd* must be **R0**…**R7**; *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**

6.1.77 LDMIB

Thumb syntax (16-bit, v4T)

```
LDMIB   Rd, {Rn, Rm…}                       [1]
```

Thumb syntax (32-bit, v6T2)

```
LDMIB   Rd, {Rn, Rm…}
LDMIB   Rd!, {Rn, Rm…}                      [1]
```

Arm syntax (v4T)

```
LDMIB   Rd, {Rn, Rm…}                       [1]
LDMIB   Rd!, {Rn, Rm…}                      [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.78 LDR

Thumb syntax (16-bit, v4T)

```
LDR     Rd, [Rn, #0…124]              [1]
LDR     Rd, [SP, #0…1020]             [2]
LDR     Rd, [PC, #0…1020]             [2]
LDR     Rd, [Rn, Rm]                  [3]
LDR     Rd, #Label
```

Thumb syntax (32-bit, v6T2)

```
LDR     Rd, [Rn, #-255…4095]
LDR     Rd, [PC, #-4095…4095]
LDR     Rd, [Rn, #-255…255]!          [4]
LDR     Rd, [Rn], #-255…255           [4]
LDR     Rd, [Rn, Rm]                  [5]
LDR     Rd, [Rn, Rm, LSL #1…3]        [6]
LDR     Rd, #Label
```

Arm syntax (v4T)

```
LDR     Rd, [Rn, #-4095…4095]
LDR     Rd, [Rn, #-4095…4095]!        [4]
LDR     Rd, [Rn], #-4095…4095         [4]
LDR     Rd, [Rn], ±Rm, shift          [7]
LDR     Rd, [Rn, ±Rm]                 [8]
LDR     Rd, [Rn, ±Rm]!                [7]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**; offset a multiple of 4
[2] offset a multiple of 4
[3] *Rd*, *Rn*, *Rm* must be **R0**…**R7**
[4] *Rn* ≠ **PC**; *Rd* ≠ *Rn*
[5] *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**
[6] *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; **LSL #***0* is also permitted
[7] *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *Rd* ≠ *Rn*
[8] *Rm* ≠ **PC**; *Rm* ≠ **SP**

6.1.79 LDRB

Thumb syntax (16-bit, v4T)

```
LDRB    Rd, [Rn, #0…31]                 [1]
LDRB    Rd, [Rn, Rm]                    [2]
```

Thumb syntax (32-bit, v6T2)

```
LDRB    Rd, [Rn, #-255…4095]            [3]
LDRB    Rd, [PC, #-4095…4095]           [3]
LDRB    Rd, [Rn, #-255…255]!            [4]
LDRB    Rd, [Rn], #-255…255             [4]
LDRB    Rd, [Rn, Rm, LSL #1…3]          [5]
```

Arm syntax (v4T)

```
LDRB    Rd, [Rn, #-4095…4095]           [3]
LDRB    Rd, [Rn, #-4095…4095]!          [4]
LDRB    Rd, [Rn], #-4095…4095           [4]
LDRB    Rd, [Rn], ±Rm, shift            [6]
LDRB    Rd, [Rn, ±Rm]                   [7]
LDRB    Rd, [Rn, ±Rm]!                  [6]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**; *Rd* ≠ **PC**; *Rd* ≠ **SP**
[2] *Rd*, *Rn*, *Rm* must be **R0**…**R7**
[3] *Rd* ≠ **PC**; *Rd* ≠ **SP**
[4] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; **LSL #***0* is also permitted
[6] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*
[7] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**

6.1.80 LDRBT

Thumb syntax (32-bit, v6T2)

LDRBT *Rd*, [*Rn*, #*0…255*] [1]

Arm syntax (v4T)

LDRBT *Rd*, [*Rn*], #*-4095…4095* [2]
LDRBT *Rd*, [*Rn*], ±*Rm*, *shift* [3]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**
[2] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*

6.1.81 LDRD

Thumb syntax (32-bit, v6T2)

```
LDRD    Rd1, Rd2, [Rn, #-1020…1020]       [1]
LDRD    Rd1, Rd2, [Rn, #-1020…1020]!      [2]
LDRD    Rd1, Rd2, [Rn], #-1020…1020       [2]
```

Arm syntax (v)

```
LDRD    Rd1, Rd2, [Rn, #-255…255]         [3]
LDRD    Rd1, Rd2, [Rn, #-255…255]!        [4]
LDRD    Rd1, Rd2, [Rn], #-255…255         [4]
LDRD    Rd1, Rd2, [Rn, ±Rm]               [5]
LDRD    Rd1, Rd2, [Rn, ±Rm]!              [6]
LDRD    Rd1, Rd2, [Rn], ±Rm               [6]
```

Notes

[1] *Rd1*, *Rd2* ≠ **PC**; *Rd1*, *Rd2* ≠ **SP**; *Rd1* ≠ *Rd2*; offset a multiple of 4
[2] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd1*, *Rd2* ≠ **SP**; *Rd1* ≠ *Rd2* ≠ *Rn*; offset a multiple of 4
[3] *Rd1*, *Rd2* ≠ **PC**; *Rd2* ≠ **SP**; *Rd1* ≠ *Rd2*; *Rd1* must be an even-numbered register; *Rd2* must be the following odd-numbered register
[4] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd2* ≠ **SP**; *Rd1* ≠ *Rd2* ≠ *Rn*; *Rd1* must be an even-numbered register; *Rd2* must be the following odd-numbered register
[5] *Rd1*, *Rd2*, *Rm* ≠ **PC**; *Rd2*, *Rm* ≠ **SP**; *Rd1* ≠ *Rd2* ≠ *Rm*; *Rd1* must be an even-numbered register; *Rd2* must be the following odd-numbered register
[6] *Rd1*, *Rd2*, *Rn*, *Rm* ≠ **PC**; *Rd2*, *Rm* ≠ **SP**; *Rd1* ≠ *Rd2* ≠ *Rn*; *Rd1* ≠ *Rd2* ≠ *Rm*; *Rd1* must be an even-numbered register; *Rd2* must be the following odd-numbered register

6.1.82 LDREX

Thumb syntax (32-bit, v6T2)

```
LDREX    Rd, [Rn, #0…1020]                [1]
```

Arm syntax (v6)

```
LDREX    Rd, [Rn]                         [2]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; offset a multiple of 4
[2] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.83 LDREXB

Thumb syntax (32-bit, v6T2)

LDREXB *Rd*, [*Rn*] [1]

Arm syntax (v6)

LDREXB *Rd*, [*Rn*] [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.84 LDREXD

Thumb syntax (32-bit, v)

LDREXD *Rd1*, *Rd2*, [*Rn*]	[1]

Arm syntax (v)

LDREXD *Rd1*, *Rd2*, [*Rn*]	[2]

Notes

[1] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd1*, *Rd2* ≠ **SP**; *Rd1* ≠ *Rd2*
[2] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd1*, *Rd2* ≠ **SP**; *Rd1* ≠ *Rd2*; *Rd1* must be an even–numbered register; *Rd2* must be the following odd–numbered register

6.1.85 LDREXH

Thumb syntax (32-bit, v6T2)

```
LDREXH  Rd, [Rn]                          [1]
```

Arm syntax (v6)

```
LDREXH  Rd, [Rn]                          [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.86 LDRH

Thumb syntax (16-bit, v4T)

```
LDRH    Rd, [Rn, #0…62]                   [1]
LDRH    Rd, [Rn, Rm]                      [2]
```

Thumb syntax (32-bit, v6T2)

```
LDRH    Rd, [Rn, #-255…4095]              [3]
LDRH    Rd, [PC, #-4095…4095]             [4]
LDRH    Rd, [Rn, #-255…255]!              [5]
LDRH    Rd, [Rn], #-255…255               [5]
LDRH    Rd, [Rn, Rm]                      [6]
LDRH    Rd, [Rn, Rm, LSL #1…3]            [7]
```

Arm syntax (v4T)

```
LDRH    Rd, [Rn, #-255…255]               [4]
LDRH    Rd, [Rn, #-255…255]!              [5]
LDRH    Rd, [Rn], #-255…255               [5]
LDRH    Rd, [Rn, ±Rm]                     [8]
LDRH    Rd, [Rn, ±Rm]!                    [9]
LDRH    Rd, [Rn], ±Rm                     [9]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**; offset a multiple of 2
[2] *Rd*, *Rn*, *Rm* must be **R0**…**R7**
[3] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**
[4] *Rd* ≠ **PC**; *Rd* ≠ **SP**
[5] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[6] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**
[7] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; **LSL #*0*** is also permitted
[8] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**
[9] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*

6.1.87 LDRHT

Thumb syntax (32-bit, v6T2)

```
LDRHT    Rd, [Rn, #0…255]                    [1]
```

Arm syntax (v6T2)

```
LDRHT    Rd, [Rn], #-255…255                 [2]
LDRHT    Rd, [Rn], ±Rm                       [3]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**
[2] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*

6.1.88 LDRSB

Thumb syntax (16-bit, v4T)

```
LDRSB   Rd, [Rn, Rm]                    [1]
```

Thumb syntax (32-bit, v6T2)

```
LDRSB   Rd, [Rn, #-255…4095]            [2]
LDRSB   Rd, [PC, #-4095…4095]           [3]
LDRSB   Rd, [Rn, #-255…255]!            [4]
LDRSB   Rd, [Rn], #-255…255             [4]
LDRSB   Rd, [Rn, Rm, LSL #1…3]          [5]
```

Arm syntax (v4T)

```
LDRSB   Rd, [Rn, #-255…255]             [3]
LDRSB   Rd, [Rn, #-255…255]!            [4]
LDRSB   Rd, [Rn], #-255…255             [4]
LDRSB   Rd, [Rn, ±Rm]                   [6]
LDRSB   Rd, [Rn, ±Rm]!                  [7]
LDRSB   Rd, [Rn], ±Rm                   [7]
```

Notes

[1] *Rd*, *Rn*, *Rm* must be **R0**…**R7**
[2] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**
[3] *Rd* ≠ **PC**; *Rd* ≠ **SP**
[4] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; **LSL #*0*** is also permitted
[6] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**
[7] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*

6.1.89 LDRSBT

Thumb syntax (32-bit, v6T2)

```
LDRSBT  Rd, [Rn, #0…255]                  [1]
```

Arm syntax (v6T2)

```
LDRSBT  Rd, [Rn], #-255…255               [2]
LDRSBT  Rd, [Rn], ±Rm                     [3]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**
[2] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*

6.1.90 LDRSH

Thumb syntax (16-bit, v4T)

```
LDRSH    Rd, [Rn, Rm]                        [1]
```

Thumb syntax (32-bit, v6T2)

```
LDRSH    Rd, [Rn, #-255…4095]                [2]
LDRSH    Rd, [PC, #-4095…4095]               [3]
LDRSH    Rd, [Rn, #-255…255]!                [4]
LDRSH    Rd, [Rn], #-255…255                 [4]
LDRSH    Rd, [Rn, Rm, LSL #1…3]              [5]
```

Arm syntax (v4T)

```
LDRSH    Rd, [Rn, #-255…255]                 [3]
LDRSH    Rd, [Rn, #-255…255]!                [4]
LDRSH    Rd, [Rn], #-255…255                 [4]
LDRSH    Rd, [Rn, ±Rm]                       [6]
LDRSH    Rd, [Rn, ±Rm]!                      [7]
LDRSH    Rd, [Rn], ±Rm                       [7]
```

Notes

[1] *Rd*, *Rn*, *Rm* must be **R0**…**R7**
[2] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**
[3] *Rd* ≠ **PC**; *Rd* ≠ **SP**
[4] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; **LSL #***0* is also permitted
[6] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**
[7] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*

6.1.91 LDRSHT

Thumb syntax (32-bit, v6T2)

```
LDRSHT   Rd, [Rn, #0…255]                    [1]
```

Arm syntax (v6T2)

```
LDRSHT   Rd, [Rn], #-255…255                 [2]
LDRSHT   Rd, [Rn], ±Rm                       [3]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**
[2] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*

6.1.92 LDRT

Thumb syntax (32-bit, v6T2)

```
LDRT     Rd, [Rn, #0…255]                [1]
```

Arm syntax (v4T)

```
LDRT     Rd, [Rn], #-4095…4095           [2]
LDRT     Rd, [Rn], ±Rm, shift            [3]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**
[2] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**; *Rd* ≠ *Rn*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *Rd* ≠ *Rn*

6.1.93 LSL

Thumb syntax (16-bit, v4T)

LSLS	*Rd*, *Rn*	[1]
LSLS	*Rd*, *Rn*, **#***0…31*	[1]

Thumb syntax (32-bit, v6T2)

LSL	*Rd*, *Rn*, *Rm*	[2]
LSL	*Rd*, *Rn*, **#***0…31*	[3]
LSLS	*Rd*, *Rn*, *Rm*	[2]
LSLS	*Rd*, *Rn*, **#***0…31*	[3]

Arm syntax (v4T)

LSL	*Rd*, *Rn*, *Rm*	[2]
LSL	*Rd*, *Rn*, **#***0…31*	[3]
LSLS	*Rd*, *Rn*, *Rm*	[2]
LSLS	*Rd*, *Rn*, **#***0…31*	[3]

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**
[3] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.94 LSR

Thumb syntax (16-bit, v4T)

LSRS	*Rd*, *Rn*	[1]
LSRS	*Rd*, *Rn*, **#***1…32*	[1]

Thumb syntax (32-bit, v6T2)

LSR	*Rd*, *Rn*, *Rm*	[2]
LSR	*Rd*, *Rn*, **#***1…32*	[3]
LSRS	*Rd*, *Rn*, *Rm*	[2]
LSRS	*Rd*, *Rn*, **#***1…32*	[3]

Arm syntax (v4T)

LSR	*Rd*, *Rn*, *Rm*	[2]
LSR	*Rd*, *Rn*, **#***1…32*	[3]
LSRS	*Rd*, *Rn*, *Rm*	[2]
LSRS	*Rd*, *Rn*, **#***1…32*	[3]

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**
[3] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.95 MCR

Thumb syntax (32-bit, v6T2)

`MCR`	*** Unknown ***

Arm syntax (v6T2)

`MCR`	*** Unknown ***

6.1.96 MCR2

Thumb syntax (32-bit, v6T2)

`MCR2`	*** Unknown ***

Arm syntax (v6T2)

`MCR2`	*** Unknown ***

6.1.97 MLA

Thumb syntax (32-bit, v6T2)

```
MLA     Rd, Rn, Rm, Ra                  [1]
```

Arm syntax (v6T2)

```
MLA     Rd, Rn, Rm, Ra                  [1]
MLAS    Rd, Rn, Rm, Ra                  [1]
```

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.98 MLS

Thumb syntax (32-bit, v6T2)

MLS	*Rd*, *Rn*, *Rm*, *Ra*	[1]

Arm syntax (v6T2)

MLS	*Rd*, *Rn*, *Rm*, *Ra*	[1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.99 MOV

Thumb syntax (16-bit, v4T)

```
MOV     Rd, Rn
MOVS    *** Unknown ***
MOVS    Rd, Rn                      [1]
MOVS    Rd, #0                      [2]
```

Thumb syntax (32-bit, v6T2)

```
MOV     Rd, #0…65535                [3]
MOV     Rd, #const                  [4]
MOV     Rd, Rn, shift               [5]
MOVS    Rd, #const                  [4]
MOVS    Rd, Rn, shift               [6]
```

Arm syntax (v4T)

```
MOV     Rd, #0…65535                [3]
MOV     Rd, #const                  [7]
MOV     Rd, Rn, shift               [8]
MOVS    Rd, #const                  [7]
MOVS    Rd, Rn, shift               [9]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd* must be **R0**…**R7**
[3] *Rd* ≠ **PC**; *Rd* ≠ **SP**
[4] *Rd* ≠ **PC**; *Rd* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0…24*
[5] *Rd*, *Rn* ≠ **PC**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[6] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[7] *Rd* ≠ **PC**; *Rd* ≠ **SP**; *const* is *$xx* **ROR** 2n
[8] *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**
[9] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.100 MOVS

Thumb syntax (16-bit, v4T)

MOVS	*** Unknown ***	
MOVS	*Rd*, *Rn*	[1]
MOVS	*Rd*, #*0*	[2]

Thumb syntax (32-bit, v6T2)

MOVS	*Rd*, #*const*	[3]
MOVS	*Rd*, *Rn*, *shift*	[4]

Arm syntax (v4T)

MOVS	*Rd*, #*const*	[5]
MOVS	*Rd*, *Rn*, *shift*	[6]

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd* must be **R0**…**R7**
[3] *Rd* ≠ **PC**; *Rd* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0…24*
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[5] *Rd* ≠ **PC**; *Rd* ≠ **SP**; *const* is *$xx* **ROR** 2n
[6] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.101 MOVW

Thumb syntax (32-bit, v6T2)

MOVW *Rd*, **#***0…65535* [1]

Arm syntax (v6T2)

MOVW *Rd*, **#***0…65535* [1]

Notes

[1] *Rd* ≠ **PC**; *Rd* ≠ **SP**

6.1.102 MRC

Thumb syntax (32-bit, v6T2)

MRC	*** Unknown ***	[1]

Arm syntax (v6T2)

MRC	*** Unknown ***	[1]

Notes

[1] *Rm* ≠ **PC**; *Rm* ≠ **SP**

6.1.103 MRC2

Thumb syntax (32-bit, v6T2)

`MRC2` *** Unknown ***

Arm syntax (v6T2)

`MRC2` *** Unknown ***

6.1.104 MRRC

Thumb syntax (32-bit, v6T2)

`MRRC` *** Unknown ***

Arm syntax (v6T2)

`MRRC` *** Unknown ***

6.1.105 MRRC2

Thumb syntax (32-bit, v6T2)

`MRRC2` *** Unknown ***

Arm syntax (v6T2)

`MRRC2` *** Unknown ***

6.1.106 MRS

Thumb syntax (32-bit, v6-M)

MRS	*** Unknown ***	[1]

Arm syntax (v6T2)

MRS	*** Unknown ***	[1]

Notes

[1] *Rd* ≠ **PC**; *Rd* ≠ **SP**

6.1.107 MSR

Thumb syntax (32-bit, v6-M)

MSR	*** Unknown ***	[1]

Arm syntax (v6T2)

MSR	*** Unknown ***	[1]

Notes

[1] *Rn* ≠ **PC**; *Rn* ≠ **SP**

6.1.108 MUL

Thumb syntax (16-bit, v4T)

```
MULS    Rd, Rn                                   [1]
```

Thumb syntax (32-bit, v6T2)

```
MUL     Rd, Rn, Rm                               [2]
```

Arm syntax (v6)

```
MUL     Rd, Rn, Rm                               [2]
MULS    Rd, Rn, Rm                               [2]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**; ordering *Rd*, *Rn*, *Rd* is also permitted
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.109 MVN

Thumb syntax (16-bit, v4T)

```
MVNS     Rd, Rn                         [1]
```

Thumb syntax (32-bit, v6T2)

```
MVN      Rd, #const                     [2]
MVN      Rd, Rn, shift                  [3]
MVNS     Rd, #const                     [2]
MVNS     Rd, Rn, shift                  [3]
```

Arm syntax (v4T)

```
MVN      Rd, #const                     [4]
MVN      Rd, Rn, shift                  [5]
MVNS     Rd, #const                     [4]
MVNS     Rd, Rn, shift                  [5]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd* ≠ **PC**; *Rd* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0…24*
[3] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.110 NEG

Thumb syntax (16-bit, v4T)

```
NEGS     Rd, Rn                                [1]
```

Thumb syntax (32-bit, v6T2)

```
NEG      Rd, Rn                                [2]
NEGS     Rd, Rn                                [2]
```

Arm syntax (v4T)

```
NEG      Rd, Rn                                [2]
NEGS     Rd, Rn                                [2]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.111 NOP

Thumb syntax (16-bit, v4T)

```
NOP
```

Thumb syntax (32-bit, v6T2)

```
NOP
```

Arm syntax (v)

```
NOP
```

6.1.112 ORN

Thumb syntax (32-bit, v6T2)

ORN	*Rd*, *Rn*, **#***const*	[1]
ORN	*Rd*, *Rn*, *Rm*, *shift*	[2]
ORNS	*Rd*, *Rn*, **#***const*	[1]
ORNS	*Rd*, *Rn*, *Rm*, *shift*	[2]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0…24*
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**

6.1.113 ORR

Thumb syntax (16-bit, v4T)

```
ORRS    Rd, Rn                           [1]
```

Thumb syntax (32-bit, v6T2)

```
ORR     Rd, Rn, #const                   [2]
ORR     Rd, Rn, Rm, shift                [3]
ORRS    Rd, Rn, #const                   [2]
ORRS    Rd, Rn, Rm, shift                [3]
```

Arm syntax (v4T)

```
ORR     Rd, Rn, #const                   [4]
ORR     Rd, Rn, Rm, shift                [5]
ORRS    Rd, Rn, #const                   [4]
ORRS    Rd, Rn, Rm, shift                [5]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**; ordering *Rd*, *Rn*, *Rd* is also permitted
[2] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.114 PKHBT

Thumb syntax (32-bit, v7E-M)

PKHBT *Rd*, *Rn*, *Rm*, *shift* [1]

Arm syntax (v)

PKHBT *Rd*, *Rn*, *Rm*, *shift* [2]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.115 PKHTB

Thumb syntax (32-bit, v7E-M)

PKHTB *Rd*, *Rn*, *Rm*, *shift* [1]

Arm syntax (v)

PKHTB *Rd*, *Rn*, *Rm*, *shift* [2]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.116 PLD

Thumb syntax (32-bit, v6T2)

```
PLD     [Rd, #-4095…4095]                [1]
PLD     Rd, [Rn, Rm, LSL #1…3]           [2]
```

Arm syntax (v6T2)

```
PLD     [Rd, #-4095…4095]
```

Notes

[1] *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**; **LSL #***0* is also permitted

6.1.117 PLDW

Thumb syntax (32-bit, v6T2)

```
PLDW    [Rd, #-255…4095]              [1]
PLDW    Rd, [Rn, Rm, LSL #1…3]        [2]
```

Arm syntax (v6T2)

```
PLDW    [Rd, #-4095…4095]             [1]
```

Notes

[1] *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**; **LSL #***0* is also permitted

6.1.118 PLI

Thumb syntax (32-bit, v6T2)

```
PLI     [Rd, #-4095…4095]                  [1]
PLI     Rd, [Rn, Rm, LSL #1…3]             [2]
```

Arm syntax (v6T2)

```
PLI     [Rd, #-4095…4095]
```

Notes

[1] *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**; **LSL #***0* is also permitted

6.1.119 POP

Thumb syntax (16-bit, v4T)

```
POP     {Rn, Rm…}
```

Thumb syntax (32-bit, v6T2)

```
POP     {Rn, Rm…}
```

Arm syntax (v4T)

```
POP     {Rn, Rm…}
```

6.1.120 PUSH

Thumb syntax (16-bit, v4T)

```
PUSH    {Rn, Rm…}
```

Thumb syntax (32-bit, v6T2)

```
PUSH    {Rn, Rm…}
```

Arm syntax (v4T)

```
PUSH    {Rn, Rm…}
```

6.1.121 QADD

Thumb syntax (32-bit, v6T2)

```
QADD    Rd, Rn, Rm                    [1]
```

Arm syntax (v)

```
QADD    Rd, Rn, Rm                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.122 QADD16

Thumb syntax (32-bit, v6T2)

QADD16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

QADD16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.123 QADD8

Thumb syntax (32-bit, v6T2)

QADD8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

QADD8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.124 QASX

Thumb syntax (32-bit, v6T2)

```
QASX    Rd, Rn, Rm                    [1]
```

Arm syntax (v6T2)

```
QASX    Rd, Rn, Rm                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.125 QDADD

Thumb syntax (32-bit, v6T2)

```
QDADD    Rd, Rn, Rm                        [1]
```

Arm syntax (v)

```
QDADD    Rd, Rn, Rm                        [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.126 QDSUB

Thumb syntax (32-bit, v6T2)

QDSUB *Rd*, *Rn*, *Rm* [1]

Arm syntax (v)

QDSUB *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.127 QSAX

Thumb syntax (32-bit, v6T2)

QSAX	*Rd*, *Rn*, *Rm*	[1]

Arm syntax (v6T2)

QSAX	*Rd*, *Rn*, *Rm*	[1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.128 QSUB

Thumb syntax (32-bit, v6T2)

```
QSUB    Rd, Rn, Rm                    [1]
```

Arm syntax (v)

```
QSUB    Rd, Rn, Rm                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.129 QSUB16

Thumb syntax (32-bit, v6T2)

QSUB16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

QSUB16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.130 QSUB8

Thumb syntax (32-bit, v6T2)

```
QSUB8    Rd, Rn, Rm                    [1]
```

Arm syntax (v6T2)

```
QSUB8    Rd, Rn, Rm                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.131 RBIT

Thumb syntax (32-bit, v6T2)

```
RBIT    Rd, Rn                       [1]
```

Arm syntax (v6T2)

```
RBIT    Rd, Rn                       [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.132 REV

Thumb syntax (16-bit, v6)

REV	*Rd*, *Rn*	[1]

Thumb syntax (32-bit, v6T2)

REV	*Rd*, *Rn*	[1]

Arm syntax (v6T2)

REV	*Rd*, *Rn*	[1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.133 REV16

Thumb syntax (16-bit, v6)

REV16 *Rd*, *Rn* [1]

Thumb syntax (32-bit, v6T2)

REV16 *Rd*, *Rn* [1]

Arm syntax (v6T2)

REV16 *Rd*, *Rn* [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.134 REVSH

Thumb syntax (16-bit, v6)

REVSH *Rd*, *Rn* [1]

Thumb syntax (32-bit, v6T2)

REVSH *Rd*, *Rn* [1]

Arm syntax (v6T2)

REVSH *Rd*, *Rn* [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.135 RFE

Thumb syntax (32-bit, v6)

```
RFE      SP                                        [1]
RFE      SP!                                       [1]
```

Arm syntax (v6)

```
RFE      SP                                        [1]
RFE      SP!                                       [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.136 RFEDA

Arm syntax (v6)

```
RFEDA   SP                                  [1]
RFEDA   SP!                                 [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.137 RFEDB

Thumb syntax (32-bit, v6)

```
RFEDB   SP                               [1]
RFEDB   SP!                              [1]
```

Arm syntax (v6)

```
RFEDB   SP                               [1]
RFEDB   SP!                              [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.138 RFEIA

Thumb syntax (32-bit, v6)

```
RFEIA   SP                                [1]
RFEIA   SP!                               [1]
```

Arm syntax (v6)

```
RFEIA   SP                                [1]
RFEIA   SP!                               [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.139 RFEIB

Arm syntax (v6)

```
RFEIB   SP                                [1]
RFEIB   SP!                               [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.140 ROR

Thumb syntax (16-bit, v4T)

```
RORS    Rd, Rn                          [1]
```

Thumb syntax (32-bit, v6T2)

```
ROR     Rd, Rn, Rm                      [2]
ROR     Rd, Rn, #1…31                   [3]
RORS    Rd, Rn, Rm                      [2]
RORS    Rd, Rn, #1…31                   [3]
```

Arm syntax (v4T)

```
ROR     Rd, Rn, Rm                      [2]
ROR     Rd, Rn, #1…31                   [3]
RORS    Rd, Rn, Rm                      [2]
RORS    Rd, Rn, #1…31                   [3]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**
[3] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.141 RRX

Thumb syntax (32-bit, v6T2)

RRX	*Rd*, *Rn*	[1]
RRXS	*Rd*, *Rn*	[1]

Arm syntax (v4T)

RRX	*Rd*, *Rn*	[1]
RRXS	*Rd*, *Rn*	[1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.142 RSB

Thumb syntax (16-bit, v4T)

RSBS	*Rd*, *Rn*, #*0*	[1]

Thumb syntax (32-bit, v6T2)

RSB	*Rd*, *Rn*, #*const*	[2]
RSB	*Rd*, *Rn*, *Rm*, *shift*	[3]
RSBS	*Rd*, *Rn*, #*const*	[2]
RSBS	*Rd*, *Rn*, *Rm*, *shift*	[3]

Arm syntax (v4T)

RSB	*Rd*, *Rn*, #*const*	[4]
RSB	*Rd*, *Rn*, *Rm*, *shift*	[5]
RSBS	*Rd*, *Rn*, #*const*	[4]
RSBS	*Rd*, *Rn*, *Rm*, *shift*	[5]

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.143 RSC

Arm syntax (v4T)

RSC	*Rd*, *Rn*, **#***const*	[1]
RSC	*Rd*, *Rn*, *Rm*, *shift*	[2]
RSCS	*Rd*, *Rn*, **#***const*	[1]
RSCS	*Rd*, *Rn*, *Rm*, *shift*	[2]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[2] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.144 SADD16

Thumb syntax (32-bit, v6T2)

SADD16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SADD16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.145 SADD8

Thumb syntax (32-bit, v6T2)

SADD8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SADD8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.146 SASX

Thumb syntax (32-bit, v6T2)

SASX	*Rd*, *Rn*, *Rm*	[1]

Arm syntax (v6T2)

SASX	*Rd*, *Rn*, *Rm*	[1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.147 SBC

Thumb syntax (16-bit, v4T)

```
SBCS    Rd, Rn                     [1]
```

Thumb syntax (32-bit, v6T2)

```
SBC     Rd, Rn, #const             [2]
SBC     Rd, Rn, Rm, shift          [3]
SBCS    Rd, Rn, #const             [2]
SBCS    Rd, Rn, Rm, shift          [3]
```

Arm syntax (v4T)

```
SBC     Rd, Rn, #const             [4]
SBC     Rd, Rn, Rm, shift          [5]
SBCS    Rd, Rn, #const             [4]
SBCS    Rd, Rn, Rm, shift          [5]
```

Notes

[1] *Rd*, *Rn* must be **R0**…**R7**
[2] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[3] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[4] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.148 SBFX

Thumb syntax (32-bit, v6T2)

```
SBFX     Rd, Rn, #lsb, #width               [1]
```

Arm syntax (v6T2)

```
SBFX     Rd, Rn, #lsb, #width               [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.149 SDIV

Thumb syntax (32-bit, v7)

SDIV	*Rd*, *Rn*, *Rm*	[1]

Arm syntax (v)

SDIV	*Rd*, *Rn*, *Rm*	[1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.150 SEL

Thumb syntax (32-bit, v6T2)

```
SEL     Rd, Rn, Rm
```

Arm syntax (v6T2)

```
SEL     Rd, Rn, Rm
```

6.1.151 SETEND

Thumb syntax (16-bit, v6)

SETEND *endian*	[1]

Arm syntax (v6)

SETEND *endian*	[1]

Notes

[1] *endian* is ’LE’ or ’BE’; not permitted in an IT block

6.1.152 SETPAN

Thumb syntax (16-bit, v)

```
SETPAN  #0…1                              [1]
```

Arm syntax (v)

```
SETPAN  #0…1
```

Notes

[1] not permitted in an IT block

6.1.153 SEV

Thumb syntax (16-bit, v6K)

```
SEV
```

Thumb syntax (32-bit, v7)

```
SEV
```

Arm syntax (v6K)

```
SEV
```

6.1.154 SEVL

Thumb syntax (16-bit, v)

```
SEVL
```

Thumb syntax (32-bit, v)

```
SEVL
```

Arm syntax (v)

```
SEVL
```

6.1.155 SHADD16

Thumb syntax (32-bit, v6T2)

SHADD16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SHADD16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.156 SHADD8

Thumb syntax (32-bit, v6T2)

SHADD8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SHADD8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.157 SHASX

Thumb syntax (32-bit, v6T2)

SHASX *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SHASX *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.158 SHSAX

Thumb syntax (32-bit, v6T2)

SHSAX *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SHSAX *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.159 SHSUB16

Thumb syntax (32-bit, v6T2)

SHSUB16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SHSUB16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.160 SHSUB8

Thumb syntax (32-bit, v6T2)

SHSUB8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SHSUB8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.161 SMC

Thumb syntax (32-bit, v)

```
SMC      #0...15
```

Arm syntax (v)

```
SMC      #0...15
```

6.1.162 SMLABB

Thumb syntax (32-bit, v6T2)

SMLABB *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLABB *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.163 SMLABT

Thumb syntax (32-bit, v6T2)

SMLABT *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLABT *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.164 SMLAD

Thumb syntax (32-bit, v6T2)

SMLAD *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLAD *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.165 SMLADX

Thumb syntax (32-bit, v6T2)

SMLADX *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLADX *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.166 SMLAL

Thumb syntax (32-bit, v6T2)

```
SMLAL   Rd, Rn, Rm, Ra                  [1]
```

Arm syntax (v6T2)

```
SMLAL   Rd, Rn, Rm, Ra                  [1]
SMLALS  Rd, Rn, Rm, Ra                  [1]
```

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.167 SMLALBB

Thumb syntax (32-bit, v6T2)

SMLALBB *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLALBB *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.168 SMLALBT

Thumb syntax (32-bit, v6T2)

`SMLALBT` *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

`SMLALBT` *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **`PC`**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **`SP`**; *Rd* ≠ *Rn*

6.1.169 SMLALD

Thumb syntax (32-bit, v7E-M)

```
SMLALD  Rd, Rn, Rm, Ra                    [1]
```

Arm syntax (v6T2)

```
SMLALD  Rd, Rn, Rm, Ra                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.170 SMLALDX

Thumb syntax (32-bit, v7E-M)

`SMLALDX` *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

`SMLALDX` *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.171 SMLALTB

Thumb syntax (32-bit, v6T2)

SMLALTB *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLALTB *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.172 SMLALTT

Thumb syntax (32-bit, v6T2)

SMLALTT *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLALTT *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.173 SMLATB

Thumb syntax (32-bit, v6T2)

SMLATB *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLATB *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.174 SMLATT

Thumb syntax (32-bit, v6T2)

```
SMLATT  Rd, Rn, Rm, Ra                    [1]
```

Arm syntax (v6T2)

```
SMLATT  Rd, Rn, Rm, Ra                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.175 SMLAWB

Thumb syntax (32-bit, v6T2)

SMLAWB *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLAWB *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.176 SMLAWT

Thumb syntax (32-bit, v6T2)

SMLAWT *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLAWT *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.177 SMLSD

Thumb syntax (32-bit, v6T2)

SMLSD *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLSD *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.178 SMLSDX

Thumb syntax (32-bit, v6T2)

SMLSDX *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMLSDX *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.179 SMLSLD

Thumb syntax (32-bit, v6T2)

`SMLSLD` *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

`SMLSLD` *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.180 SMLSLDX

Thumb syntax (32-bit, v6T2)

`SMLSLDX` *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

`SMLSLDX` *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.181 SMMLA

Thumb syntax (32-bit, v6T2)

SMMLA *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMMLA *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.182 SMMLAR

Thumb syntax (32-bit, v6T2)

`SMMLAR` *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

`SMMLAR` *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.183 SMMLS

Thumb syntax (32-bit, v6T2)

SMMLS *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMMLS *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.184 SMMLSR

Thumb syntax (32-bit, v6T2)

SMMLSR *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

SMMLSR *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.185 SMMUL

Thumb syntax (32-bit, v6T2)

```
SMMUL   Rd, Rn, Rm                        [1]
```

Arm syntax (v6T2)

```
SMMUL   Rd, Rn, Rm                        [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.186 SMMULR

Thumb syntax (32-bit, v6T2)

SMMULR *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SMMULR *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.187 SMUAD

Thumb syntax (32-bit, v6T2)

```
SMUAD    Rd, Rn, Rm                         [1]
```

Arm syntax (v6T2)

```
SMUAD    Rd, Rn, Rm                         [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.188 SMUADX

Thumb syntax (32-bit, v6T2)

SMUADX *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SMUADX *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.189 SMULBB

Thumb syntax (32-bit, v6T2)

SMULBB *Rd*, *Rn*, *Rm* [1]

Arm syntax (v)

SMULBB *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.190 SMULBT

Thumb syntax (32-bit, v6T2)

SMULBT *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SMULBT *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.191 SMULL

Thumb syntax (32-bit, v6T2)

```
SMULL   Rd, Rn, Rm, Ra                  [1]
```

Arm syntax (v6)

```
SMULL   Rd, Rn, Rm, Ra                  [1]
SMULLS  Rd, Rn, Rm, Ra                  [1]
```

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.192 SMULTB

Thumb syntax (32-bit, v6T2)

SMULTB *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SMULTB *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.193 SMULTT

Thumb syntax (32-bit, v6T2)

```
SMULTT  Rd, Rn, Rm                         [1]
```

Arm syntax (v6T2)

```
SMULTT  Rd, Rn, Rm                         [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.194 SMULWB

Thumb syntax (32-bit, v6T2)

SMULWB *Rd*, *Rn*, *Rm* [1]

Arm syntax (v)

SMULWB *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.195 SMULWT

Thumb syntax (32-bit, v6T2)

SMULWT *Rd*, *Rn*, *Rm* [1]

Arm syntax (v)

SMULWT *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.196 SMUSD

Thumb syntax (32-bit, v6T2)

SMUSD *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SMUSD *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.197 SMUSDX

Thumb syntax (32-bit, v6T2)

SMUSDX *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SMUSDX *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.198 SRS

Thumb syntax (32-bit, v6)

```
SRS      SP, #0…31
SRS      SP!, #0…31
```

Arm syntax (v6)

```
SRS      SP, #0…31
SRS      SP!, #0…31
```

6.1.199 SRSDA

Arm syntax (v6)

```
SRSDA   SP, #0…31
SRSDA   SP!, #0…31
```

6.1.200 SRSDB

Thumb syntax (32-bit, v6)

```
SRSDB   SP, #0…31
SRSDB   SP!, #0…31
```

Arm syntax (v6)

```
SRSDB   SP, #0…31
SRSDB   SP!, #0…31
```

6.1.201 SRSIA

Thumb syntax (32-bit, v6)

```
SRSIA   SP, #0…31
SRSIA   SP!, #0…31
```

Arm syntax (v6)

```
SRSIA   SP, #0…31
SRSIA   SP!, #0…31
```

6.1.202 SRSIB

Arm syntax (v6)

```
SRSIB   SP, #0…31
SRSIB   SP!, #0…31
```

6.1.203 SSAT

Thumb syntax (32-bit, v6T2)

SSAT *Rd*, **#***1…32*, *Rn* [1]

Arm syntax (v6T2)

SSAT *Rd*, **#***1…32*, *Rn* [1]

Notes

[1] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**

6.1.204 SSAT16

Thumb syntax (32-bit, v6T2)

SSAT16 *Rd*, **#***1…16*, *Rn* [1]

Arm syntax (v6T2)

SSAT16 *Rd*, **#***1…16*, *Rn* [1]

Notes

[1] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**

6.1.205 SSAX

Thumb syntax (32-bit, v6T2)

```
SSAX    Rd, Rn, Rm                    [1]
```

Arm syntax (v6T2)

```
SSAX    Rd, Rn, Rm                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.206 SSUB16

Thumb syntax (32-bit, v6T2)

SSUB16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

SSUB16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.207 SSUB8

Thumb syntax (32-bit, v6T2)

SSUB8 *Rd*, *Rn*, *Rm*	[1]

Arm syntax (v6T2)

SSUB8 *Rd*, *Rn*, *Rm*	[1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.208 STL

Thumb syntax (32-bit, v)

STL	*Rd*, [*Rn*]	[1]

Arm syntax (v)

STL	*Rd*, [*Rn*]	[1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.209 STLB

Thumb syntax (32-bit, v)

STLB	*Rd*, **[***Rn***]**	[1]

Arm syntax (v)

STLB	*Rd*, **[***Rn***]**	[1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.210 STLH

Thumb syntax (32-bit, v)

```
STLH    Rd, [Rn]                          [1]
```

Arm syntax (v)

```
STLH    Rd, [Rn]                          [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.211 STM

Thumb syntax (16-bit, v4T)

```
STM     Rd, {Rn, Rm…}
STM     Rd!, {Rn, Rm…}                    [1]
```

Thumb syntax (32-bit, v6T2)

```
STM     Rd, {Rn, Rm…}                     [2]
STM     Rd!, {Rn, Rm…}                    [2]
```

Arm syntax (v4T)

```
STM     Rd, {Rn, Rm…}                     [2]
STM     Rd!, {Rn, Rm…}                    [2]
```

Notes

[1] *Rd* must be **R0**…**R7**; *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**

6.1.212 STMDA

Thumb syntax (16-bit, v4T)

STMDA *Rd*, {*Rn*, *Rm*...}

Thumb syntax (32-bit, v6T2)

STMDA *Rd*, {*Rn*, *Rm*...} [1]
STMDA *Rd*!, {*Rn*, *Rm*...} [1]

Arm syntax (v4T)

STMDA *Rd*, {*Rn*, *Rm*...} [1]
STMDA *Rd*!, {*Rn*, *Rm*...} [1]

Notes

[1] *Rd* ≠ **PC**

6.1.213 STMDB

Thumb syntax (16-bit, v4T)

```
STMDB   SP!, {Rn, Rm…}
```

Thumb syntax (32-bit, v6T2)

```
STMDB   Rd, {Rn, Rm…}                    [1]
STMDB   Rd!, {Rn, Rm…}                   [1]
```

Arm syntax (v4T)

```
STMDB   Rd, {Rn, Rm…}                    [1]
STMDB   Rd!, {Rn, Rm…}                   [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.214 STMEA

Thumb syntax (16-bit, v4T)

```
STMEA   SP!, {Rn, Rm…}
```

Thumb syntax (32-bit, v6T2)

```
STMEA   Rd, {Rn, Rm…}                  [1]
STMEA   Rd!, {Rn, Rm…}                 [1]
```

Arm syntax (v4T)

```
STMEA   Rd, {Rn, Rm…}                  [1]
STMEA   Rd!, {Rn, Rm…}                 [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.215 STMED

Thumb syntax (16-bit, v4T)

```
STMED   Rd, {Rn, Rm…}                    [1]
```

Thumb syntax (32-bit, v6T2)

```
STMED   Rd, {Rn, Rm…}                    [1]
STMED   Rd!, {Rn, Rm…}                   [1]
```

Arm syntax (v4T)

```
STMED   Rd, {Rn, Rm…}                    [1]
STMED   Rd!, {Rn, Rm…}                   [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.216 STMFA

Thumb syntax (16-bit, v4T)

```
STMFA   Rd, {Rn, Rm…}
```

Thumb syntax (32-bit, v6T2)

```
STMFA   Rd, {Rn, Rm…}                   [1]
STMFA   Rd!, {Rn, Rm…}                  [1]
```

Arm syntax (v4T)

```
STMFA   Rd, {Rn, Rm…}                   [1]
STMFA   Rd!, {Rn, Rm…}                  [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.217 STMFD

Thumb syntax (16-bit, v4T)

```
STMFD   Rd, {Rn, Rm…}
STMFD   Rd!, {Rn, Rm…}                   [1]
```

Thumb syntax (32-bit, v6T2)

```
STMFD   Rd, {Rn, Rm…}                    [2]
STMFD   Rd!, {Rn, Rm…}                   [2]
```

Arm syntax (v4T)

```
STMFD   Rd, {Rn, Rm…}                    [2]
STMFD   Rd!, {Rn, Rm…}                   [2]
```

Notes

[1] *Rd* must be **R0**…**R7**; *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**

6.1.218 STMIA

Thumb syntax (16-bit, v4T)

```
STMIA   Rd, {Rn, Rm...}
STMIA   Rd!, {Rn, Rm...}                 [1]
```

Thumb syntax (32-bit, v6T2)

```
STMIA   Rd, {Rn, Rm...}                  [2]
STMIA   Rd!, {Rn, Rm...}                 [2]
```

Arm syntax (v4T)

```
STMIA   Rd, {Rn, Rm...}                  [2]
STMIA   Rd!, {Rn, Rm...}                 [2]
```

Notes

[1] *Rd* must be **R0**...**R7**; *Rd* ≠ **PC**
[2] *Rd* ≠ **PC**

6.1.219 STMIB

Thumb syntax (16-bit, v4T)

```
STMIB   Rd, {Rn, Rm…}                      [1]
```

Thumb syntax (32-bit, v6T2)

```
STMIB   Rd, {Rn, Rm…}                      [1]
STMIB   Rd!, {Rn, Rm…}                     [1]
```

Arm syntax (v4T)

```
STMIB   Rd, {Rn, Rm…}                      [1]
STMIB   Rd!, {Rn, Rm…}                     [1]
```

Notes

[1] *Rd* ≠ **PC**

6.1.220 STR

Thumb syntax (16-bit, v4T)

```
STR     Rt, [Rn, #0…124]                  [1]
STR     Rt, [SP, #0…1020]                 [2]
STR     Rt, [Rn, Rm]                      [3]
```

Thumb syntax (32-bit, v6T2)

```
STR     Rt, [Rn, #-255…4095]              [4]
STR     Rt, [Rn, #-255…255]!              [5]
STR     Rt, [Rn], #-255…255               [5]
STR     Rt, [Rn, Rm, LSL #1…3]            [6]
```

Arm syntax (v4T)

```
STR     Rt, [Rn, #-4095…4095]             [5]
STR     Rt, [Rn, #-4095…4095]!            [5]
STR     Rt, [Rn], #-4095…4095             [5]
STR     Rt, [Rn], ±Rm, shift              [7]
STR     Rt, [Rn, ±Rm]                     [7]
STR     Rt, [Rn, ±Rm]!                    [7]
```

Notes

[1] *Rt*, *Rn* must be **R0**…**R7**; offset a multiple of 4
[2] offset a multiple of 4
[3] *Rt*, *Rn*, *Rm* must be **R0**…**R7**
[4] *Rt*, *Rn* ≠ **PC**
[5] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ *Rn*
[6] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; **LSL #***0* is also permitted
[7] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *Rt* ≠ *Rn*

6.1.221 STRB

Thumb syntax (16-bit, v4T)

```
STRB    Rt, [Rn, #0…31]                 [1]
STRB    Rt, [Rn, Rm]                    [2]
```

Thumb syntax (32-bit, v6T2)

```
STRB    Rt, [Rn, #-255…4095]            [3]
STRB    Rt, [Rn, #-255…255]!            [4]
STRB    Rt, [Rn], #-255…255             [4]
STRB    Rt, [Rn, Rm, LSL #1…3]          [5]
```

Arm syntax (v4T)

```
STRB    Rt, [Rn, #-4095…4095]           [3]
STRB    Rt, [Rn, #-4095…4095]!          [4]
STRB    Rt, [Rn], #-4095…4095           [4]
STRB    Rt, [Rn], ±Rm, shift            [6]
STRB    Rt, [Rn, ±Rm]                   [7]
STRB    Rt, [Rn, ±Rm]!                  [6]
```

Notes

[1] *Rt*, *Rn* must be **R0**…**R7**
[2] *Rt*, *Rn*, *Rm* must be **R0**…**R7**
[3] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**
[4] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**; *Rt* ≠ *Rn*
[5] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**; **LSL #***0* is also permitted
[6] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**; *Rt* ≠ *Rn*
[7] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**

6.1.222 STRBT

Thumb syntax (32-bit, v6T2)

STRBT *Rt*, [*Rn*, #*0…255*] [1]

Arm syntax (v4T)

STRBT *Rt*, [*Rn*], #*-4095…4095* [2]
STRBT *Rt*, [*Rn*], ±*Rm*, *shift* [3]

Notes

[1] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**
[2] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**; *Rt* ≠ *Rn*
[3] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**; *Rt* ≠ *Rn*

6.1.223 STRD

Thumb syntax (32-bit, v6T2)

```
STRD    Rd1, Rd2, [Rn, #-1020…1020]       [1]
STRD    Rd1, Rd2, [Rn, #-1020…1020]!      [2]
STRD    Rd1, Rd2, [Rn], #-1020…1020       [2]
```

Arm syntax (v)

```
STRD    Rd1, Rd2, [Rn, #-255…255]         [3]
STRD    Rd1, Rd2, [Rn, #-255…255]!        [4]
STRD    Rd1, Rd2, [Rn], #-255…255         [3]
STRD    Rd1, Rd2, [Rn, ±Rm]               [5]
STRD    Rd1, Rd2, [Rn, ±Rm]!              [6]
STRD    Rd1, Rd2, [Rn], ±Rm               [6]
```

Notes

[1] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd1*, *Rd2* ≠ **SP**; offset a multiple of 4
[2] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd1*, *Rd2* ≠ **SP**; *Rd1* ≠ *Rn*; *Rd2* ≠ *Rn*; offset a multiple of 4
[3] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd2* ≠ **SP**; *Rd1* must be an even-numbered register; *Rd2* must be the following odd-numbered register
[4] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd2* ≠ **SP**; *Rd1* ≠ *Rn*; *Rd2* ≠ *Rn*; *Rd1* must be an even-numbered register; *Rd2* must be the following odd-numbered register
[5] *Rd1*, *Rd2*, *Rn*, *Rm* ≠ **PC**; *Rd2*, *Rm* ≠ **SP**; *Rd1* ≠ *Rd2*; *Rd1* must be an even-numbered register; *Rd2* must be the following odd-numbered register
[6] *Rd1*, *Rd2*, *Rn*, *Rm* ≠ **PC**; *Rd2*, *Rm* ≠ **SP**; *Rd1* ≠ *Rd2* ≠ *Rm*; *Rd1* must be an even-numbered register; *Rd2* must be the following odd-numbered register

6.1.224 STREX

Thumb syntax (32-bit, v6T2)

```
STREX    Rd1, Rd2, [Rn, #0…1020]          [1]
```

Arm syntax (v6T2)

```
STREX    Rd1, Rd2, [Rn]                   [2]
```

Notes

[1] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd1*, *Rd2* ≠ **SP**; *Rd1* ≠ *Rd2*; *Rd1* ≠ *Rn*; offset a multiple of 4
[2] *Rd1*, *Rd2*, *Rn* ≠ **PC**; *Rd1*, *Rd2* ≠ **SP**; *Rd1* ≠ *Rd2*; *Rd1* ≠ *Rn*

6.1.225 STREXB

Thumb syntax (32-bit, v6T2)

STREXB *Rd1*, *Rd*2, [*Rn*] [1]

Arm syntax (v6T2)

STREXB *Rd1*, *Rd*2, [*Rn*] [1]

Notes

[1] *Rd1*, *Rd*2, *Rn* ≠ **PC**; *Rd1*, *Rd*2 ≠ **SP**; *Rd1* ≠ *Rd*2; *Rd1* ≠ *Rn*

6.1.226 STREXD

Thumb syntax (32-bit, v6T2)

STREXD	*** Unknown ***	[1]

Arm syntax (v6T2)

STREXD	*** Unknown ***	[2]

Notes

[1] *Rt*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rt*, *Rn*, *Rm* ≠ **SP**; *Rt* ≠ *Rn*; *Rt* ≠ *Rm*; *Rt* ≠ *Ra*
[2] *Rt*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rt*, *Rn*, *Rm* ≠ **SP**; *Rt* ≠ *Rn*; *Rt* ≠ *Rm*; *Rt* ≠ *Ra*; *Rt* must be an even–numbered register; *Rn* must be the following odd–numbered register

6.1.227 STREXH

Thumb syntax (32-bit, v6T2)

STREXH *Rd1*, *Rd*2, [*Rn*] [1]

Arm syntax (v6T2)

STREXH *Rd1*, *Rd*2, [*Rn*] [1]

Notes

[1] *Rd1*, *Rd*2, *Rn* ≠ **PC**; *Rd1*, *Rd*2 ≠ **SP**; *Rd1* ≠ *Rd*2; *Rd1* ≠ *Rn*

6.1.228 STRH

Thumb syntax (16-bit, v4T)

```
STRH    Rt, [Rn, #0…62]                 [1]
STRH    Rt, [Rn, Rm]                    [2]
```

Thumb syntax (32-bit, v6T2)

```
STRH    Rt, [Rn, #-255…4095]            [3]
STRH    Rt, [Rn, #-255…255]!            [4]
STRH    Rt, [Rn], #-255…255             [4]
STRH    Rt, [Rn, Rm, LSL #1…3]          [5]
```

Arm syntax (v4T)

```
STRH    Rt, [Rn, #-255…255]             [3]
STRH    Rt, [Rn, #-255…255]!            [4]
STRH    Rt, [Rn], #-255…255             [4]
STRH    Rt, [Rn, ±Rm]                   [6]
STRH    Rt, [Rn, ±Rm]!                  [7]
STRH    Rt, [Rn], ±Rm                   [7]
```

Notes

[1] *Rt*, *Rn* must be **R0**…**R7**; offset a multiple of 2
[2] *Rt*, *Rn*, *Rm* must be **R0**…**R7**
[3] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**
[4] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**; *Rt* ≠ *Rn*
[5] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**; **LSL #***0* is also permitted
[6] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**
[7] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**; *Rt* ≠ *Rn*

6.1.229 STRHT

Thumb syntax (32-bit, v6T2)

STRHT *Rt*, [*Rn*, #*0…255*] [1]

Arm syntax (v6T2)

STRHT *Rt*, [*Rn*], #*−255…255* [2]
STRHT *Rt*, [*Rn*], ±*Rm* [3]

Notes

[1] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**
[2] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**; *Rt* ≠ *Rn*
[3] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**; *Rt* ≠ *Rn*

6.1.230 STRT

Thumb syntax (32-bit, v6T2)

```
STRT    Rt, [Rn, #0…255]                [1]
```

Arm syntax (v4T)

```
STRT    Rt, [Rn], #-4095…4095           [2]
STRT    Rt, [Rn], ±Rm, shift            [3]
```

Notes

[1] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**
[2] *Rt*, *Rn* ≠ **PC**; *Rt* ≠ **SP**; *Rt* ≠ *Rn*
[3] *Rt*, *Rn*, *Rm* ≠ **PC**; *Rt*, *Rm* ≠ **SP**; *Rt* ≠ *Rn*

6.1.231 SUB

Thumb syntax (16-bit, v4T)

```
SUB     SP, SP, #0…508              [1]
SUB     Rd, SP, #0                  [2]
SUB     Rd, PC, #0                  [2]
SUBS    Rd, Rn, Rm                  [3]
SUBS    Rd, #0…255                  [2]
SUBS    Rd, Rn, #0…7                [4]
```

Thumb syntax (32-bit, v6T2)

```
SUB     Rd, Rn, #0…4095
SUB     Rd, Rn, #const              [5]
SUB     Rd, Rn, Rm, shift           [6]
SUBS    Rd, Rn, #const              [5]
SUBS    Rd, Rn, Rm, shift           [6]
```

Arm syntax (v4T)

```
SUB     Rd, Rn, #const              [7]
SUB     Rd, Rn, Rm, shift           [8]
SUB     Rd, SP, Rm, shift           [9]
SUBS    Rd, Rn, #const              [7]
SUBS    Rd, Rn, Rm, shift           [8]
SUBS    Rd, SP, Rm, shift           [9]
```

Notes

[1] offset a multiple of 4
[2] *Rd* must be **R0**…**R7**
[3] *Rd*, *Rn*, *Rm* must be **R0**…**R7**
[4] *Rd*, *Rn* must be **R0**…**R7**
[5] *Rd* ≠ **PC**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0…24*
[6] *Rd*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*, **LSR #***1…32*, **ASR #***1…32*, **RRX**
[7] *const* is *$xx* **ROR** 2n
[8] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**
[9] *Rd*, *Rm* ≠ **PC**; *Rm* ≠ **SP**; *shift* is one of **LSL #***0…31*/*Rs*, **LSR #***1…32*/*Rs*, **ASR #***1…32*/*Rs*, **ROR #***1…31*/*Rs*, **RRX**

6.1.232 SUBW

Thumb syntax (32-bit, v6T2)

```
SUBW    Rd, Rn, #0…4095
```

6.1.233 SVC

Thumb syntax (16-bit, v4T)

```
SVC     #0...255
```

Arm syntax (v4T)

```
SVC     *** Unknown ***
```

6.1.234 SXTAB

Thumb syntax (32-bit, v6T2)

SXTAB *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

SXTAB *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.235 SXTAB16

Thumb syntax (32-bit, v6T2)

SXTAB16 *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

SXTAB16 *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.236 SXTAH

Thumb syntax (32-bit, v6T2)

```
SXTAH    Rd, Rn, Rm{, ROR #imm}                [1]
```

Arm syntax (v6T2)

```
SXTAH    Rd, Rn, Rm{, ROR #imm}                [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.237 SXTB

Thumb syntax (16-bit, v6)

SXTB *Rd*, *Rn*

Thumb syntax (32-bit, v6T2)

SXTB *Rd*, *Rn*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

SXTB *Rd*, *Rn*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.238 SXTB16

Thumb syntax (32-bit, v6T2)

SXTB16 *Rd*, *Rn*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

SXTB16 *Rd*, *Rn*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.239 SXTH

Thumb syntax (16-bit, v6)

SXTH *Rd*, *Rn*

Thumb syntax (32-bit, v6T2)

SXTH *Rd*, *Rn*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

SXTH *Rd*, *Rn*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.240 TBB

Thumb syntax (32-bit, v6T2)

```
TBB      [Rd, Rn]
```

6.1.241 TBH

Thumb syntax (32-bit, v6T2)

```
TBH     [Rd, Rn, LSL #1]
```

6.1.242 TEQ

Thumb syntax (32-bit, v6T2)

```
TEQ     Rn, #const                    [1]
TEQ     Rn, Rm, shift                 [2]
```

Arm syntax (v4T)

```
TEQ     Rn, #const                    [3]
TEQ     Rn, Rm, shift                 [4]
```

Notes

[1] *Rn* ≠ **PC**; *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[2] *Rn*, *Rm* ≠ **PC**; *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[3] *Rn* ≠ **PC**; *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[4] *Rn*, *Rm* ≠ **PC**; *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.243 TST

Thumb syntax (16-bit, v4T)

```
TST     Rn, Rm                          [1]
```

Thumb syntax (32-bit, v6T2)

```
TST     Rn, #const                      [2]
TST     Rn, Rm, shift                   [3]
```

Arm syntax (v4T)

```
TST     Rn, #const                      [4]
TST     Rn, Rm, shift                   [5]
```

Notes

[1] *Rn*, *Rm* must be **R0**…**R7**
[2] *Rn* ≠ **PC**; *Rn* ≠ **SP**; *const* is one of *$00xx00xx*, *$xx00xx00*, *$xx*xxxxxx, *$xx* **LSL** *0*…*24*
[3] *Rn*, *Rm* ≠ **PC**; *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*, **LSR #***1*…*32*, **ASR #***1*…*32*, **RRX**
[4] *Rn* ≠ **PC**; *Rn* ≠ **SP**; *const* is *$xx* **ROR** 2n
[5] *Rn*, *Rm* ≠ **PC**; *Rn*, *Rm* ≠ **SP**; *shift* is one of **LSL #***0*…*31*/*Rs*, **LSR #***1*…*32*/*Rs*, **ASR #***1*…*32*/*Rs*, **ROR #***1*…*31*/*Rs*, **RRX**

6.1.244 TT

Thumb syntax (32-bit, v8-M.BASE)

```
TT      Rd, Rn                    [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.245 TTA

Thumb syntax (32-bit, v8-M.BASE)

```
TTA       Rd, Rn                              [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.246 TTAT

Thumb syntax (32-bit, v8-M.BASE)

```
TTAT    Rd, Rn                    [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.247 TTT

Thumb syntax (32-bit, v8-M.BASE)

```
TTT     Rd, Rn                    [1]
```

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd* ≠ **SP**

6.1.248 UADD16

Thumb syntax (32-bit, v6T2)

UADD16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UADD16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.249 UADD8

Thumb syntax (32-bit, v6T2)

UADD8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UADD8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.250 UASX

Thumb syntax (32-bit, v6T2)

UASX *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UASX *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.251 UBFX

Thumb syntax (32-bit, v6T2)

UBFX *Rd*, *Rn*, **#***lsb*, **#***width* [1]

Arm syntax (v6T2)

UBFX *Rd*, *Rn*, **#***lsb*, **#***width* [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.252 UDF

Thumb syntax (16-bit, v4T)

```
UDF     #0…255
```

Thumb syntax (32-bit, v6T2)

```
UDF     #0…65535
```

Arm syntax (v4T)

```
UDF     Rd, #0…65535
```

6.1.253 UDIV

Thumb syntax (32-bit, v7)

UDIV *Rd*, *Rn*, *Rm* [1]

Arm syntax (v)

UDIV *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.254 UHADD16

Thumb syntax (32-bit, v6T2)

UHADD16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UHADD16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.255 UHADD8

Thumb syntax (32-bit, v6T2)

UHADD8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UHADD8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.256 UHASX

Thumb syntax (32-bit, v6T2)

```
UHASX    Rd, Rn, Rm                     [1]
```

Arm syntax (v6T2)

```
UHASX    Rd, Rn, Rm                     [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.257 UHSAX

Thumb syntax (32-bit, v6T2)

`UHSAX` *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

`UHSAX` *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.258 UHSUB16

Thumb syntax (32-bit, v6T2)

UHSUB16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UHSUB16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.259 UHSUB8

Thumb syntax (32-bit, v6T2)

```
UHSUB8   Rd, Rn, Rm                      [1]
```

Arm syntax (v6T2)

```
UHSUB8   Rd, Rn, Rm                      [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.260 UMAAL

Thumb syntax (32-bit, v6T2)

```
UMAAL    Rd, Rn, Rm, Ra                    [1]
```

Arm syntax (v6T2)

```
UMAAL    Rd, Rn, Rm, Ra                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.261 UMLAL

Thumb syntax (32-bit, v6T2)

```
UMLAL   Rd, Rn, Rm, Ra                 [1]
```

Arm syntax (v6T2)

```
UMLAL   Rd, Rn, Rm, Ra                 [1]
UMLALS  Rd, Rn, Rm, Ra                 [1]
```

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.262 UMULL

Thumb syntax (32-bit, v6T2)

UMULL	*Rd*, *Rn*, *Rm*, *Ra*	[1]

Arm syntax (v6)

UMULL	*Rd*, *Rn*, *Rm*, *Ra*	[1]
UMULLS	*Rd*, *Rn*, *Rm*, *Ra*	[1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**; *Rd* ≠ *Rn*

6.1.263 UQADD16

Thumb syntax (32-bit, v6T2)

UQADD16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UQADD16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.264 UQADD8

Thumb syntax (32-bit, v6T2)

UQADD8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UQADD8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.265 UQASX

Thumb syntax (32-bit, v6T2)

UQASX *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UQASX *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.266 UQSAX

Thumb syntax (32-bit, v6T2)

```
UQSAX    Rd, Rn, Rm                    [1]
```

Arm syntax (v6T2)

```
UQSAX    Rd, Rn, Rm                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.267 UQSUB16

Thumb syntax (32-bit, v6T2)

UQSUB16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UQSUB16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.268 UQSUB8

Thumb syntax (32-bit, v6T2)

UQSUB8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

UQSUB8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.269 USAD8

Thumb syntax (32-bit, v6T2)

USAD8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

USAD8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.270 USADA8

Thumb syntax (32-bit, v6T2)

USADA8 *Rd*, *Rn*, *Rm*, *Ra* [1]

Arm syntax (v6T2)

USADA8 *Rd*, *Rn*, *Rm*, *Ra* [1]

Notes

[1] *Rd*, *Rn*, *Rm*, *Ra* ≠ **PC**; *Rd*, *Rn*, *Rm*, *Ra* ≠ **SP**

6.1.271 USAT

Thumb syntax (32-bit, v6T2)

USAT *Rd*, **#***0…31*, *Rn* [1]

Arm syntax (v6T2)

USAT *Rd*, **#***0…31*, *Rn* [1]

Notes

[1] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**

6.1.272 USAT16

Thumb syntax (32-bit, v6T2)

USAT16 *Rd*, **#***0…15*, *Rn* [1]

Arm syntax (v6T2)

USAT16 *Rd*, **#***0…15*, *Rn* [1]

Notes

[1] *Rd*, *Rm* ≠ **PC**; *Rd*, *Rm* ≠ **SP**

6.1.273 USAX

Thumb syntax (32-bit, v6T2)

```
USAX    Rd, Rn, Rm                    [1]
```

Arm syntax (v6T2)

```
USAX    Rd, Rn, Rm                    [1]
```

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.274 USUB16

Thumb syntax (32-bit, v6T2)

USUB16 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

USUB16 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.275 USUB8

Thumb syntax (32-bit, v6T2)

USUB8 *Rd*, *Rn*, *Rm* [1]

Arm syntax (v6T2)

USUB8 *Rd*, *Rn*, *Rm* [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.276 UXTAB

Thumb syntax (32-bit, v6T2)

UXTAB *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

UXTAB *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.277 UXTAB16

Thumb syntax (32-bit, v6T2)

UXTAB16 *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

UXTAB16 *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.278 UXTAH

Thumb syntax (32-bit, v6T2)

UXTAH *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

UXTAH *Rd*, *Rn*, *Rm*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn*, *Rm* ≠ **PC**; *Rd*, *Rn*, *Rm* ≠ **SP**

6.1.279 UXTB

Thumb syntax (16-bit, v6)

UXTB *Rd*, *Rn*

Thumb syntax (32-bit, v6T2)

UXTB *Rd*, *Rn*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

UXTB *Rd*, *Rn*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.280 UXTB16

Thumb syntax (32-bit, v6T2)

UXTB16 *Rd*, *Rn*{, **ROR #***imm*} [1]

Arm syntax (v6T2)

UXTB16 *Rd*, *Rn*{, **ROR #***imm*} [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.281 UXTH

Thumb syntax (16-bit, v6)

UXTH *Rd*, *Rn*

Thumb syntax (32-bit, v6T2)

UXTH *Rd*, *Rn*{, **ROR** #*imm*} [1]

Arm syntax (v6T2)

UXTH *Rd*, *Rn*{, **ROR** #*imm*} [1]

Notes

[1] *Rd*, *Rn* ≠ **PC**; *Rd*, *Rn* ≠ **SP**

6.1.282 WFE

Thumb syntax (16-bit, v6K)

```
WFE
```

Thumb syntax (32-bit, v7)

```
WFE
```

Arm syntax (v)

```
WFE
```

6.1.283 WFI

Thumb syntax (16-bit, v6K)

```
WFI
```

Thumb syntax (32-bit, v7)

```
WFI
```

Arm syntax (v)

```
WFI
```

6.1.284 YIELD

Thumb syntax (16-bit, v6K)

```
YIELD
```

Thumb syntax (32-bit, v7)

```
YIELD
```

Arm syntax (v)

```
YIELD
```

Chapter 7

Command-line options

Command line option naming is generally compatible with the following toolsets:

- GNU assembler `as`
- Arm assembler `armasm`
- IAR assembler `iasmarm`

7.1 Input file options

7.1.1 --auto-import

Summary

Automatically import symbols.

Syntax

```
--auto-import
```

Description

This option instructs the assembler to automatically define as symbol as an external (imported) symbol if it is used and has not been declared with a different storage class.

See also

--no-auto-import on page 316

7.1.2 --define

Summary

Define symbol.

Syntax

`--define` *symbol*=*value*
`-D`*symbol*=*value*

Description

This option defines the symbol *symbol* to the numeric value *value*.

7.1.3 --dependency-file

Summary

Write dependencies to file.

Syntax

`--dependency-file` *filename*
`--MD` *filename*

Description

This option instructs the assembler to write the list of dependencies (included source files and included binary files) to the fle *filename*.

7.1.4 --include

Summary

Add include directory.

Syntax

`--include` *path*
`-I`*path*

Description

This option adds the path *path* to the end of the list of directories to search for included source files and included binary files.

7.1.5 --no-auto-import

Summary

Do not automatically import symbols.

Syntax

```
--no-auto-import
```

Description

This option instructs the assembler not to automatically define as symbol as an external (imported) symbol and to require that all symbols are declared as local or external before their first use.

See also

--auto-import on page 312

7.1.6 --syntax

Summary

Set source file syntax.

Syntax

```
--input-syntax=name
-fsyntax=name
```

Description

This option selects the source file syntax. The syntaxes accepted by the assembler are:

- **arm** — Format is compatible with ARM Limited’s `armasm` assembler.
- **iar** — Format is compatible with IAR’s `iasmarm` assembler.
- **gnu** — Format is compatible with The Free Software Foundation’s `gas` assembler.

There is one additional syntax identifier:

- **auto** — Source format is automatically derived from the input file.

The default is **auto** syntax where the assembler automatically determines the appropriate source file syntax using heuristics.

7.1.7 --via

Synopsis

Read additional options and input files from file.

Syntax

--via *filename*
--via=*filename*
-f *filename*
@*filename*

Description

This option reads the file *filename* for additional options and input files. Options are separated by spaces or newlines, and file names which contain special characters, such as spaces, must be enclosed in double quotation marks.

Notes

This option can only be provided on the command line and cannot appear in an indirect file.

7.2 Output file options

7.2.1 --keep-empty-sections

Summary

Keep empty sections in Elf file.

Syntax

```
--keep-empty-sections
```

Description

This option instructs the assembler to emit empty sections that have been created during assembly. An empty section is a section that is created by a section directive but has no instructions or data allocated to it.

The default is to eliminate empty sections.

See also

--no-keep-empty-sections on page 320

7.2.2 --no-keep-empty-sections

Summary

Eliminate empty sections from Elf file.

Syntax

```
--no-keep-empty-sections
```

Description

This option instructs the assembler to eliminate empty sections that have been created during assembly. An empty section is a section that is created by a section directive but has no instructions or data allocated to it.

The default is to eliminate empty sections.

See also

--keep-empty-sections on page 319

7.2.3 --no-mapping-symbols

Summary

Do not generate mapping symbols.

Syntax

```
--no-mapping-symbols
```

Description

This option produces an Arm ELF file that contains no code and data mapping symbols.

See also

--mapping-symbols on page 325

7.2.4 --no-section-padding

Summary

Pad sections to alignment boundary.

Syntax

```
--no-section-padding
```

Description

This option instructs the assembler not to emit additional data to a section such that its size is a multiple of its alignment. Padding a section to a multiple of its alignment may waste space that a linker could otherwise utilize for section packing.

The default is not to pad sections.

See also

--section-padding on page 327

7.2.5 --no-symbols

Summary

Discard symbol table.

Syntax

```
--no-symbols
```

Description

This option eliminates all symbols from the Arm ELF file that are not required which includes the mapping symbols.

See also

--symbols on page 326

7.2.6 --output

Summary

Set output file name.

Syntax

--output *filename*
-o *filename*

--output=*filename*
-o=*filename*

Description

This option sets the Arm ELF output filename, typically with extension “o”.

7.2.7 --mapping-symbols

Summary

Include mapping symbols.

Syntax

```
--mapping-symbols
```

Description

This option includes mapping symbols in the Arm ELF file.

See also

--no-mapping-symbols on page 321

7.2.8 --symbols

Summary

Include symbol table.

Syntax

```
--symbols
```

Description

This option includes the symbol table in the Arm ELF file. This does not automatically enable mapping symbol inclusion.

See also

--no-symbols on page 323

7.2.9 --section-padding

Summary

Pad sections to alignment boundary.

Syntax

```
--section-padding
```

Description

This option instructs the assembler to emit additional data to a section such that its size is a multiple of its alignment. Padding a section to a multiple of its alignment may waste space that a linker could otherwise utilize for section packing.

The default is not to pad sections.

See also

--no-section-padding on page 322

7.3 Target selection options

7.3.1 --cpu=name

Summary

Set target core or architecture.

Syntax

--cpu=*name*
-cpu=*name*
-mcpu=*name*

Description

This option selects the target processor for the application and controls the construction of appropriate veneers when reqiored.

The core names accepted are:

- **cortex-m0**, **cortex-m0plus**, **cortex-m0+**, **cortex-m1**, **cortex-m3**, **cortex-m4**, **cortex-m7**, **cortex-m23**, **cortex-m33**
- **cortex-a5**, **cortex-a7**, **cortex-a8**, **cortex-a9**, **cortex-a12**, **cortex-a15**, **cortex-a17**, **cortex-a32**, **cortex-a35**, **cortex-a53**, **cortex-a55**, **cortex-a57**, **cortex-a72**, **cortex-a73**, **cortex-a75**
- **cortex-r4**, **cortex-r4f**, **cortex-r5**, **cortex-r7**, **cortex-r8**, **cortex-r52**

The architecture names accepted are:

- **4t**, **5t**, **5te**
- **6**, **6j**, **6k**, **6kz**, **6t2**, **6z**, **6zk**, **6-m**, **6s-m**
- **7**, **7ve**, **7-a**, **7-r**, **7-m**, **7e-m**
- **8-a**, **8.1-a**, **8.2-a**, **8.3-a**, **8.4-a**, **8-r**, **8-m.base**, **8-m.main**

The default is `--cpu=cortex-m0`.

7.3.2 -marm

Summary

Select Arm instruction set.

Syntax

```
-marm
```

Description

This option sets the default assembler instruction set to Arm (A32). The default instruction set is the instruction set if the source file does not select a specific instruction set using assembler source directives.

See also

-mthumb on page 332

7.3.3 -mbig-endian

Summary

Select big-endian byte order.

Syntax

```
-mbig-endian
-EB
```

Description

This option selects big-endian byte ordering for data and instructions.

The default is little-endian byte ordering.

See also

-mlittle-endian on page 331

7.3.4 -mlittle-endian

Summary

Select little-endian byte order.

Syntax

```
-mlittle-endian
-EL
```

Description

This option selects little-endian byte ordering for data and instructions.

The default is little-endian byte ordering.

See also

-mbig-endian on page 330

7.3.5 -mthumb

Summary

Select Thumb instruction set.

Syntax

```
-mthumb
```

Description

This option sets the default assembler instruction set to Thumb (T32). The default instruction set is the instruction set if the source file does not select a specific instruction set using assembler source directives.

See also

-marm on page 329

7.4 List file options

7.4.1 --list-file

Summary

Generate an assembler listing file.

Syntax

--list-file *filename*
--list-file=*filename*
-L*filename*

Description

Generates an assembler listing to the given filename.

7.4.2 --list-form-feed

Summary

Issue form feed between pages.

Syntax

```
--list-form-feed
```

Description

This option instructs the assembler to write a form feed between listing pages.

See also

--list-page-length on page 335, *--no-list-form-feed* on page 337

7.4.3 --list-page-length

Summary

Set number of lines per page.

Syntax

`--list-page-length=`*`value`*

Description

This option sets the number of lines per page to *value*. If *value* is zero, which is the default, the listing is not divided into pages.

See also

--list-form-feed on page 334

7.4.4 --list-symbols

Summary

Print symbol table to listing.

Syntax

```
--list-symbols
```

Description

This option instructs the assembler to print a symbol table at the end of the listing.

See also

--no-list-symbols on page 338

7.4.5 --no-list-form-feed

Summary

Do not issue form feed between pages.

Syntax

```
--no-list-form-feed
```

Description

This option ibhibits the assembler writing a form feed between listing pages.

See also

--list-form-feed on page 334

7.4.6 --no-list-symbols

Summary

Do not print symbol table to listing.

Syntax

```
--no-list-symbols
```

Description

This option instructs the assembler not to print a symbol table at the end of the listing.

See also

--list-symbols on page 336

7.5 Control options

7.5.1 --no-warnings

Summary

Suppress warnings.

Syntax

`--no-warnings`

Description

This option disables all warning diagnostics issued by the assembler. Although warnings are suppressed, the total number of warnings that are suppressed by the assembler is shown at the end of linking:

```
C:> segger-ld --via=app.ind
Copyright (c) 2017-2018 SEGGER Microcontroller GmbH    www.segger.com
SEGGER Assembler 2.10 compiled Apr 11 2018 10:50:34

Assembly complete: 0 errors, 1 warnings suppressed, 0 remarks

C:> _
```

See also

--warnings on page 346, *--warnings-are-errors* on page 347

7.5.2 --no-remarks

Summary

Suppress remarks.

Syntax

`--no-remarks`

Description

This option disables all remark diagnostics issued by the assembler. Although remarks are suppressed, the total number of remarks that are suppressed by the assembler is shown at the end of linking:

```
C:> segger-as --via=app.ind
Copyright (c) 2017-2018 SEGGER Microcontroller GmbH    www.segger.com
SEGGER Assembler 2.10 compiled Apr 11 2018 10:50:34

Assembly complete: 0 errors, 0 warnings, 2 remarks suppressed

C:> _
```

See also

--remarks-are-warnings on page 343, *--remarks-are-errors* on page 342

7.5.3 --remarks

Summary

Issue remarks.

Syntax

```
--remarks
```

Description

This option instructs the assembler to issue remarks for potential issues during assembly. This is the default.

See also

--no-remarks on page 340, *--remarks-are-warnings* on page 343, *--remarks-are-errors* on page 342

7.5.4 --remarks-are-errors

Summary

Elevate remarks to errors.

Syntax

```
--remarks-are-errors
```

Description

This option elevates all remark diagnostics issued by the assembler to errors.

See also

--no-remarks on page 340, *--remarks* on page 341, , *--remarks-are-warnings* on page 343

7.5.5 --remarks-are-warnings

Summary

Elevate remarks to warnings.

Syntax

```
--remarks-are-warnings
--remarks_are_warnings
```

Description

This option elevates all remark diagnostics issued by the assembler to warnings.

See also

--no-remarks on page 340, *--remarks* on page 341, , *--remarks-are-errors* on page 342

7.5.6 --silent

Summary

Do not show output.

Syntax

```
--silent
-q
```

Description

This option inhibits all assembler status messages; only diagnostic messages are shown.

See also

--verbose on page 345

7.5.7 --verbose

Summary

Increase verbosity.

Syntax

```
--verbose
-v
```

Description

This option increase the verbosity of the assembler by one level.

See also

--silent on page 344

7.5.8 --warnings

Summary

Issue warnings.

Syntax

```
--warnings
```

Description

This option instructs the assembler to issue warnings for dubious use or inputs. This is the default.

See also

--no-warnings on page 339, *--warnings-are-errors* on page 347

7.5.9 --warnings-are-errors

Summary

Elevate warnings to errors.

Syntax

```
--warnings-are-errors
--fatal-warnings
```

Description

This option elevates all warning diagnostics issued by the assembler to errors.

See also

--no-warnings on page 339, *--warnings* on page 346

Chapter 8

Differences between assemblers

8.1 Syntax and Encoding differences

The following sections document the differences in UAL input syntax and the binary encoding of UAL source code. This is particularly important if you expect identical instruction encodings between assemblers, especially when using the SEGGER assembler to assemble existing source code.

8.1.1 MUL and MULS

UAL allows a multiply instruction to be written as `MUL Rd, Rn`. When the only encoding of this UAL input is a three-operand instructions, `MUL Rd, Rn` must be interpreted as `MUL Rd, Rd, Rn`.

The following are the encodings of `MUL Rd, Rn` and `MULS Rd, Rn` where only three-operand encoding is possible. The SEGGER assembler follows Arm's encoding, whereas GNU deviates and IAR rejects correct UAL.

SEGGER	Arm	GNU	IAR
`MUL Rd, Rd, Rn`	`MUL Rd, Rd, Rn`	`MUL Rd, Rn, Rd`	Rejects
`MULS Rd, Rd, Rn`	`MULS Rd, Rd, Rn`	`MULS Rd, Rn, Rd`	Rejects

8.1.2 MULS

UAL allows a multiply instruction to be written as `MULS Rd, Rn`. The assembler can choose between two-operand form and three-operand form based on the registers `Rd` and `Rn`.

The following are the encodings of `MULS R1, R2`:

SEGGER	Arm	GNU	IAR
`MULS.N R1, R2`	Rejects	`MULS.N R1, R2`	`MULS.N R1, R2`

And `MULS R1, R9`:

SEGGER	Arm	GNU	IAR
`MULS.W R1, R1, R9`	`MULS.W R1, R1, R9`	`MULS.W R1, R9, R1`	`MULS.W R1, R1, R9`

Chapter 9

Indexes

9.1 Subject index