# Thumb Instruction Set
# Quick Reference Card

| Operation | | Assembler | Lo regs | Hi regs | Cond codes | Action | Notes |
|---|---|---|---|---|---|---|---|
| **Move** | Immediate | `MOV Rd, #8_Bit_Value` | ✓ | ✗ | ✓ | Rd:= #8_Bit_Value | 8-bit immediate value |
| | Hi to Lo | `MOV Rd, Hs` | ✓ | ✓ | | Rd:= Hs | |
| | Lo to Hi | `MOV Hd, Rs` | ✓ | ✓ | | Hd:= Rs | |
| | Hi to Hi | `MOV Hd, Hs` | ✗ | ✓ | | Hd:= Hs | |
| **ALU** | Arithmetic | | | | | | |
| | Add | `ADD Rd, Rs, #3_Bit_Value` | ✓ | ✗ | ✓ | Rd:= Rs + #3_Bit_Value | 3-bit immediate value |
| | Lo and Lo | `ADD Rd, Rs, Rn` | ✓ | ✗ | ✓ | Rd:= Rn + Rs | |
| | Hi to Lo | `ADD Rd, Hs` | ✓ | ✓ | | Rd:= Rd + Hs | |
| | Lo to Hi | `ADD Hd, Rs` | ✓ | ✓ | | Hd:= Hd + Rs | |
| | Hi to Hi | `ADD Hd, Hs` | ✗ | ✓ | | Hd:= Hd + Hs | |
| | Immediate | `ADD Rd, #8_Bit_Value` | ✓ | ✗ | ✓ | Rd:= Rd + #8_Bit_Value | 8-bit immediate value |
| | Value to SP | `ADD SP, #Imm`<br>`ADD SP, #-Imm` | | | | SP:= SP + #Imm<br>SP:= SP + #-Imm | 7-bit immediate value |
| | with carry | `ADC Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd +Rs + C-bit | |
| | Subtract | `SUB Rd, Rs, Rn` | ✓ | ✗ | ✓ | Rd:= Rs - Rn | |
| | | `SUB Rd, Rs, #3_Bit_Value` | ✓ | ✗ | ✓ | Rd:= Rs - #3_Bit_Value | 3-bit immediate value |
| | Immediate | `SUB Rd, #8_Bit_Value` | ✓ | ✗ | ✓ | Rd:= Rd - #8_Bit_Value | 8-bit immediate value |
| | with carry | `SBC Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd - Rs - NOT C-bit | |
| | Negate | `NEG Rd, Rs` | ✓ | ✗ | ✓ | Rd:= -Rs | |
| | Multiply | `MUL Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd * Rs | |
| | Compare | | | | | | |
| | Lo and Lo | `CMP Rd, Rs` | ✓ | ✗ | ✓ | CPSR flags:= Rd - Rs | Set cond codes on Rd - Rs |
| | Lo and Hi | `CMP Rd, Hs` | ✓ | ✓ | ✓ | CPSR flags:= Rd - Hs | |
| | Hi and Lo | `CMP Hd, Rs` | ✓ | ✓ | ✓ | CPSR flags:= Hd - Rs | |
| | Hi and Hi | `CMP Hd, Hs` | ✗ | ✓ | ✓ | CPSR flags:= Hd - Hs | |
| | Negative | `CMN Rd, Rs` | ✓ | ✗ | ✓ | CPSR flags:= Rd + Rs | Set cond codes on Rd + Rs |
| | Immediate | `CMP Rd, #8_Bit_Value` | ✓ | ✗ | ✓ | CPSR flags:= Rd - #8_Bit_Value | 8-bit immediate value |
| | Logical | | | | | | |
| | AND | `AND Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd AND Rs | |
| | EOR | `EOR Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd EOR Rs | |
| | OR | `ORR Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd OR Rs | |
| | Bit clear | `BIC Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd AND NOT Rs | |
| | Move NOT | `MVN Rd, Rs` | ✓ | ✗ | ✓ | Rd:= NOT Rs | |
| | Test bits | `TST Rd, Rs` | ✓ | ✗ | ✓ | CPSR flags:= Rd AND Rs | Set cond codes on Rd AND Rs |
| | Shift/Rotate | | | | | | |
| | Logical shift left | `LSL Rd, Rs, #5_Bit_Offset` | ✓ | ✗ | ✓ | Rd:= Rs << #5_Bit_Offset | 5-bit immediate value |
| | | `LSL Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd << Rs | |
| | Logical shift right | `LSR Rd, Rs, #5_Bit_Offset` | ✓ | ✗ | ✓ | Rd:= Rs >> #5_Bit_Offset | 5-bit immediate value |
| | | `LSR Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd >> Rs | |
| | Arithmetic shift right | `ASR Rd, Rs, #5_Bit_Offset` | ✓ | ✗ | ✓ | Rd:= Rs ASR #5_Bit_Offset | 5-bit immediate value |
| | | `ASR Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd ASR Rs | |
| | Rotate right | `ROR Rd, Rs` | ✓ | ✗ | ✓ | Rd:= Rd ROR Rs | |

# Thumb Instruction Set
# Quick Reference Card

| Operation | | Assembler | Lo regs | Hi regs | Cond codes | Action | Notes |
|---|---|---|---|---|---|---|---|
| **Branch** | Conditional | | | | | | 9-bit two's complement address, halfword aligned (label>>1) |
| | if Z set | `BEQ label` | | | | | Equal |
| | if Z clear | `BNE label` | | | | | Not equal |
| | if C set | `BCS label` | | | | | Unsigned higher or same |
| | if C clear | `BCC label` | | | | | Unsigned lower |
| | if N set | `BMI label` | | | | | Negative |
| | if N clear | `BPL label` | | | | | Positive or zero |
| | if V set | `BVS label` | | | | | Overflow |
| | if V clear | `BVC label` | | | | | No overflow |
| | if C set and Z clear | `BHI label` | | | | | Unsigned higher |
| | if C clear and Z set | `BLS label` | | | | | Unsigned lower or same |
| | if N set and V set, or if N clear and V clear | `BGE label` | | | | | Greater or equal |
| | if N set and V clear, or if N clear and V set | `BLT label` | | | | | Less than |
| | if Z clear, and N or V set, or if Z clear, and N or V clear | `BGT label` | | | | | Greater than |
| | if Z set,or N set and V clear, or N clear and V set | `BLE label` | | | | | |
| | unconditional | `B label` | | | | | 12-bit two's complement address, word aligned (label <<1) |
| | long branch with link | `BL label` | | | | | label is 23-bit two's complement halfword offset, split into two 11-bit halves (ignoring bit 0). Encoded as 2 Thumb instructions. |
| | Optional state change | | | | | | |
| | to Lo | `BX Rs` | ✓ | ✕ | | | Toggles between ARM and Thumb state |
| | to Hi | `BX Hs` | ✕ | ✓ | | | |
| **Load** | with immediate offset | | | | | | |
| | word | `LDR Rd, [Rb, #Imm]` | ✓ | ✕ | | Rd:= [Rb + #Imm] | 7-bit immediate offset |
| | halfword | `LDRH Rd, [Rb, #Imm]` | ✓ | ✕ | | Rd:= [Rb + #Imm] | 8-bit immediate offset. Loads bits 0-15 and sets bits 16-31 to 0 |
| | byte | `LDRB Rd, [Rb, #Imm]` | ✓ | ✕ | | Rd:= [Rb + #Imm] | 5-bit immediate offset Loads bit 0-7 and sets bits 8-31 to 0 |
| | with register offset | | | | | | |
| | word | `LDR Rd, [Rb, Ro]` | ✓ | ✕ | | Rd:= [Rb + Ro] | |
| | halfword | `LDRH Rd, [Rb, Ro]` | ✓ | ✕ | | Rd:= [Rb + Ro] | Loads bits 0-15 and sets bits 16-31 to 0 |
| | signed halfword | `LDRSH Rd, [Rb, Ro]` | ✓ | ✕ | | Rd:= [Rb + Ro] | Loads bits 0-15 and sets bits 16-31 to bit 15 |
| | byte | `LDRB Rd, [Rb, Ro]` | ✓ | ✕ | | Rd:= [Rb + Ro] | Loads bits 0-7 and sets bits 8-31 to 0 |
| | signed byte | `LDRSB Rd, [Rb, Ro]` | ✓ | ✕ | | Rd:= [Rb + Ro] | Loads bits 0-7 and sets bits 8-31 to bit 7 |
| | PC-relative | `LDR Rd, [PC, #Imm]` | ✓ | ✕ | | Rd:= [PC + #Imm] | 10-bit unsigned immediate offset (word-aligned). PC bit 1 read as 0. |
| | SP-relative | `LDR Rd, [SP, #Imm]` | ✓ | ✕ | | Rd:= [SP + #Imm] | 10-bit unsigned immediate offset (word-aligned) |
| | Address | | | | | | |
| | using PC | `ADD Rd, PC, #Imm` | ✓ | ✕ | | Rd:= [PC + #Imm] | 10-bit unsigned immediate offset (word-aligned). PC bit 1 read as 0. |
| | using SP | `ADD Rd, SP, #Imm` | ✓ | ✕ | | Rd:= [SP + #Imm] | 10-bit unsigned immediate offset (word-aligned) |
| | Multiple | `LDMIA Rb!, { Rlist }` | ✓ | ✕ | | | Loads list of registers, starting at base address in Rb. Writes back new address. |

# Thumb Instruction Set
# Quick Reference Card

| Operation | | Assembler | Lo regs | Hi regs | Cond codes | Action | Notes |
|---|---|---|---|---|---|---|---|
| **Store** | with immediate offset | | | | | | |
| | word | `STR Rd, [Rb, #Imm]` | ✓ | ✗ | | [Rb + #Imm]:= Rd | 7-bit immediate offset |
| | halfword | `STRH Rd, [Rb, #Imm]` | ✓ | ✗ | | [Rb + #Imm]:= Halfword value from Rd | 8-bit immediate offset. Stores bits 0-15. |
| | byte | `STRB Rd, [Rb, #Imm]` | ✓ | ✗ | | [Rb + #Imm]:= Byte value from Rd | 7-bit immediate offset |
| | with register offset | | | | | | |
| | word | `STR Rd, [Rb, Ro]` | ✓ | ✗ | | [Rb + Ro]:= Rd | Pre-indexed word store |
| | halfword | `STRH Rd, [Rb, Ro]` | ✓ | ✗ | | [Rb + Ro]:= Halfword value from Rd | Stores bits 0-15 |
| | byte | `STRB Rd, [Rb, Ro]` | ✓ | ✗ | | [Rb + Ro]:= Byte value from Rd | Pre-indexed byte store |
| | SP-relative | `STR Rd, [SP, #Imm]` | ✓ | ✗ | | [SP + #Imm]:= Rd | 10-bit unsigned immediate offset (word-aligned) |
| | Multiple | `STMIA Rb!, { Rlist }` | ✓ | ✗ | | | Stores list of registers, starting at base address in Rb. Writes back new address. |
| **Push/ Pop** | Push registers onto stack | `PUSH { Rlist }` | ✓ | ✗ | | | Full descending stack |
| | Push LR and registers onto stack | `PUSH { Rlist, LR}` | ✓ | ✗ | | | |
| | Pop registers from stack | `POP { Rlist}` | ✓ | ✗ | | | Full descending stack |
| | Pop registers and PC from stack | `POP { Rlist, PC }` | ✓ | ✗ | | | |
| **Software Interrupt** | | `SWI #8_Bit_Value` | | | | | |

## Proprietary Notice



## Document Number

**ARM QRC 0001B**

## Change Log

| Issue | Date | By | Change |
|---|---|---|---|
| Beta | April 1995 | BJH | Beta draft |
| A | June 1995 | BJH | First Release |
| B | Sept 1996 | BJH | Second Release |

**ENGLAND**
Advanced RISC Machines Limited
Fulbourn Road
Cherry Hinton
Cambridge CB1 4JN
England

Telephone: +44 1223 400400
Facsimile: +44 1223 400410
Email: info@armltd.co.uk

**GERMANY**
Advanced RISC Machines Limited
Otto-Hahn Str. 13b
85521 Ottobrun-Riemerling
Munich
Germany

Telephone: +49 89 608 75545
Facsimile: +49 98 608 75599
Email: info@armltd.co.uk

**JAPAN**
Advanced RISC Machines K.K.
KSP West Bldg, 3F 300D, 3-2-1 Sakado
Takatsu-ku, Kawasaki-shi
Kanagawa, 213
Japan

Telephone: +81 44 850 1301
Facsimile: +81 44 850 1308
Email: info@armltd.co.uk

**USA**
ARM USA
Suite 5, 985 University Avenue
Los Gatos
California 95030
USA

Telephone: +1 408 399 5199
Facsimile: +1 408 399 8854
Email: info@arm.com

http://www.arm.com