ARM64 Instructions for General-Purpose Registers

General format

INSTR destination, op1, op2, op3 …

Datatypes

X Register (doubleword) 64-bit C long, Swift Int
W register (word) 32-bit C int, Swift Int32
(halfword) 16-bit C short, Swift Int16
(byte) 8-bit C char, Swift Int8

Load a register

LDR -- X or W registers
H unsigned halfword 16-bit
B unsigned byte 8-bit
SW signed word 32-bit
SH signed halfword 16-bit
SB signed byte 8-bit

Extend into a register

sign-extending **S**
zero-extending **U**
--- **XT** ---
W word 32-bit
H halfword 16-bit
B byte 8-bit

Move register contents

MOV

Register pairs

LDP load
STP store

Store a register

STR -- X or W register
H halfword in W (lsb)
B byte in W (lsb)

NB **STR** source, operand

Move 16-bit immediates

MOVK keeps other bits
MOVN applies bitwise NOT
MOVZ zeros other bits

Addition

ADD add
ADDS add, set flags
ADC add with Carry
ADCS add with Carry, set flags

Subtraction, negation

SUB subtract op1 – op2
SUBS subtract, set flags
SBC subtract with Carry (-ve)
SBCS subtract with Carry (-ve), set flags

NEG negate
NEGS negate, set flags
NGC negate with Carry (-ve)
NGCS negate with Carry (-ve), set flags

Carry -ve means subtract 1 if the Carry flag is clear

can take extending instruction to extend size

Multiplication

MUL multiply
MADD multiply-add (op1 x op2) + op3
MSUB/MNEG multiply-subtract (op1 x op2) – op3

SMULH signed multiply high X registers, high doubleword in destination X
SMULL signed multiply long W registers, long doubleword in destination X
SMADDL signed multiply-add long
SMSUBL/SMNEGL signed multiply-subtract long
UMULH unsigned multiply high X registers, high doubleword in destination X
UMULL unsigned multiply long W registers, long doubleword in destination X
UMADDL unsigned multiply-add long
UMSUBL/UMNEGL unsigned multiply-subtract long

Division

SDIV signed division op1/op2
UDIV unsigned division op1/op2

Bit operations

LSL left shift, moving in 0s
LSR right shift, moving in 0s
ASR right shift, preserving sign bit
ROR rotate right

AND bitwise AND
ANDS bitwise AND, setting flags
BIC bitwise AND with complement
BICS bitwise AND with complement, setting flags
ORR bitwise OR
ORN bitwise OR with complement
EOR bitwise XOR
EON bitwise XOR with complement
MVN bitwise inverse

BFC, **BFI**, **BFM**, **BFXIL** clear, insert, move bitfields
REV reverses byte order, also
REV16, **REV32**