# 5.14 Real stuff: The rest of the ARMv8 special instructions

This section has been set as optional by your instructor.

The figure below lists the 63 remaining ARMv8 instructions in the special purpose and systems category. We'll describe the instructions in the figure top-down, from left to right.

Figure 5.14.1: The list of assembly language instructions for the systems and special operations in the full ARMv8 Instruction set. Bold means the Instruction is also in LEGv8 (COD Figure 5.48).

| Type | Mnemonic | Instruction |
|---|---|---|
| Non-cache | LDNP | Load Non-temporal Pair |
| | STNP | Store Non-temporal Pair |
| Barrier | CLREX | Clear exclusive monitor |
| | DSB | Data synchronization barrier |
| | DMB | Data memory barrier |
| | ISB | Instruction synchronization barrier |
| CRC | CRC32B | CRC-32 sum from byte |
| | CRC32H | CRC-32 sum from halfword |
| | CRC32W | CRC-32 sum from word |
| | CRC32X | CRC-32 sum from doubleword |
| | CRC32CB | CRC-32C sum from byte |
| | CRC32CH | CRC-32C sum from halfword |
| | CRC32CW | CRC-32C sum from word |
| | CRC32CX | CRC-32C sum from doubleword |
| Crypto | AESD | AES single round decryption |
| | AESE | AES single round encryption |
| | AESIMC | AES inverse mix columns |
| | AESMC | AES mix columns |
| | PMULL | Polynomial multiply long |
| | SHA1C | SHA1 hash update (choose) |
| | SHA1H | SHA1 fixed rotate |
| | SHA1M | SHA1 hash update (majority) |
| | SHA1P | SHA1 hash update (parity) |
| | SHA1SU0 | SHA1 schedule update 0 |
| | SHA1SU1 | SHA1 schedule update 1 |
| | SHA256H | SHA256 hash update (part 1) |
| | SHA256H2 | SHA256 hash update (part 2) |
| | SHA256SU0 | SHA256 schedule update 0 |
| | SHA256SU1 | SHA256 schedule update 1 |
| Sys Reg | MRS | Move system register to general-purpose register |
| | MSR | Move general-purpose register or immediate to system register |

| Type | Mnemonic | Instruction |
|---|---|---|
| Unprivileged | LDTR | Load Unprivileged register |
| | LDTRB | Load Unprivileged byte |
| | LDTRSB | Load Unprivileged signed byte |
| | LDTRH | Load Unprivileged halfword |
| | LDTRSH | Load Unprivileged signed halfword |
| | LDTRSW | Load Unprivileged signed word |
| | STTR | Store Unprivileged register |
| | STTRB | Store Unprivileged byte |
| | STTRH | Store Unprivileged halfword |
| Exception | BRK | Software breakpoint instruction |
| | HLT | Halting software breakpoint instruction |
| | HVC | Generate exception targeting Exception level 2 |
| | SMC | Generate exception targeting Exception level 3 |
| | **SVC** | Generate exception targeting Exception level 1 |
| | **ERET** | Exception return using current ELR and SPSR |
| Debug | DCPS1 | Debug switch to Exception level 1 |
| | DCPS2 | Debug switch to Exception level 2 |
| | DCPS3 | Debug switch to Exception level 3 |
| | DRPS | Debug restore PE state |
| System | SYS | System instruction |
| | SYSL | System instruction with result |
| | IC | Instruction cache maintenance |
| | DC | Data cache maintenance |
| | AT | Address translation |
| | TLBI | TLB Invalidate |
| Hint | NOP | No operation |
| | YIELD | Yield hint |
| | WFE | Wait for event |
| | WFI | Wait for interrupt |
| | SEV | Send event |
| | SEVL | Send event local |
| | HINT | Unallocated hint |

The two "non-cache" or "no allocate" load pair (`LDNP`) and store pair (`STNP`) are intended for streaming through lots of data, so the data are unlikely to be used in the future; that is, no temporal locality. These instructions give hints to the memory hierarchy to *not* put the data read from or written to memory into caches, but to transfer data directly between main memory and processor registers. By specifying a pair of registers, large data transfers can proceed more quickly.

The barrier instructions provide synchronization barriers for instructions (`ISB`) and data (`DSB` and `DMB`). The latter two instructions are both barriers that affect data memory access ordering, and differ only in their strictness. The clear exclusive instruction (`CLREX`) tells the processor to give up exclusive access to a memory location that it requested earlier.

The eight CRC instructions are useful in calculating a cyclic redundancy checksum (CRC-32 or CRC-32C) on bytes, halfwords, words, or double words to help catch errors in large data sets (see the *Elaboration* in Section 5.5). Similarly, the 14 Cryptographic instructions use the SIMD registers to accelerate the computation of Advanced Encryption Standard (`AES`) encryption, Galois/Counter Mode (`GCM`) encryption, and Secure Hash Algorithm (`SHA`) encryption.

The two move system register instructions either move data into the processor state registers from a general purpose register or an immediate (`MSR`) or move data from the processor state registers into a general purpose register (`MRS`). The 31 instructions above bring us to the halfway point of surveying the special and system instructions of ARMv8.

Starting at the top of the right column of the figure above, the nine "unprivileged" loads and stores of different data widths let a processor at the EL1 interrupt level execute loads and stores that behave as if they were operating at the EL0 level, which means they can have protection faults. They act as normal loads and stores if executed at EL0 or levels higher than EL1.

The exception generation instructions include software breakpoint (`BRK`), halting software breakpoint (`HLT`), the supervisor call instructions (`SVC`), and two instructions similar to `SVC` except they go to higher exception levels: `HVC` goes to the hypervisor level (EL2) and `SMC` goes to the secure monitor level (EL3). Exception return (`ERET`), naturally enough, allows the program to return from an exception.

The four debugging instructions include three instructions that switch to a higher exception level (`DCPS1, DCPS2, DCPS3`) and one to restore to the previous processor element state (`DRPS`).

The six system management instructions include two general purpose ones (`SYS` and `SYSL`) and four for managing the memory hierarchy: instruction cache, data cache, address translation, and TLB invalidation.

Finally, the seven hint instructions provide a variety of architectural hints, including wait for interrupt, wait for event, send an event, yield, and NOP. These 32 instructions from the right column of the figure above bring the total to 63 special and system instructions in the full ARMv8 instruction set.

**PARTICIPATION ACTIVITY** 5.14.1: ARMv8 instructions for the systems and special operations.

`WFI` `MRS` `ERET` `LDNP, STNP` `CRC32X`

Non-cache load pair and store pair instructions

Cyclic redundancy checksum sum from doubleword instruction

A hint instruction

Move system register to general-purpose register instruction

Exception return instruction

Reset

Provide feedback on this section