The Arm Architecture

- Reduced Instruction Set Computer (RISC) architecture.
  - A large uniform register file.
  - A load/store architecture, where data-processing operations only operate on register contents, not directly on memory contents.
  - Simple addressing modes, with all load/store addresses determined from register contents and instruction fields only.
- Armv8 architecture supports backwards compatibility.
  - A 64-bit Execution state, AArch64.
  - A 32-bit Execution state, AArch32, that is compatible with previous versions of the Arm architecture (Armv7-A architecture profile + extensions).
  - Multiple permitted extensions to architecture.
- AArch64 execution state
  - Addresses are held in 64-bit registers.
  - Instructions in the base instruction set can use 64-bit registers for their processing.
  - Supports the A64 instruction set.

Armv8: Execution State (Registers)

- 31 64-bit GPRs, `R0` through `R30`.
  - Each register can be addressed as a 64-bit GPR named `X0` through `X30`. The least significant 32 bits of each register can be addressed as a 32-bit GPR named `W0` through `W30`.
  - `X30` is used as the procedure link register.
  - The Zero Register (ZR) is sometimes written as `X31` in pseudocode.
- A dedicated 64-bit Stack Pointer register, `SP`.
  - The least significant 32 bits can be accessed using the register name `WSP`.
  - Aligned to a 16-byte boundary (configurable).
- A 64-bit Program Counter (`PC`) holding the address of the current instruction.
  - Software cannot directly write to `PC`.
  - Word-aligned (word = 32 bits = 4 bytes).

Armv8: Execution State (Condition Flags)

- Four condition flags.
  - N: Negative Condition.
  - Z: Zero Condition.
  - C: Carry Condition (unsigned overflow).
  - V: Overflow Condition (signed overflow).
- Set by flag-setting instructions.
- Conditional instructions test the N, Z, C and V Condition flags, combining them with the Condition code for the instruction to determine whether the instruction must be executed.
- Can also be directly read or written.
  - The `NZCV` special-purpose register stores these values.
  - The instruction `MRS` reads the `NZCV` register.
  - The instruction `MSR` *immediate* writes *immediate* to the `NZCV` register.

Armv8: Execution State (Memory)

- Supports 64-bit virtual addressing.
- Address calculations are performed using 64-bit registers.
  - Supervisory software can configure the top eight address bits for use as a tag.
- Alignment
  - A64 instructions must be word-aligned.
  - Data accesses must be aligned to the size of the data element being accessed.
- Endianness
  - A64 instructions have a fixed length of 32 bits and are always little-endian.
  - Data endianness is configurable by supervisory software.

Armv8: The A64 Instruction Set

- A64 instructions have a fixed length of 32 bits and are always little-endian.
- Functional groups by encoding structure
  - A miscellaneous group of branch instructions and other stuff.
  - Data-processing instructions associated with GPRs.
    - All are held in registers.
    - Include an operand with a constant immediate value.
  - Load and store instructions associated with the GPR.
- Regular bit encoding structure.
- A64 assembly language
  - Instruction mnemonics overloaded; exact form distinguished based on operand types.
  - 16 condition codes: `EQ`, `NE`; `CS`, `CC`; `MI`, `PL`; `VS`, `VC`; `HI`, `LS`; `GE`, `LT`, `GT`, `LE`; `AL`, `NV`.

Armv8: Address Generation

- Supports 64-bit virtual addresses.
- Register indexed addressing
  - The A64 instruction set allows a 64-bit index register to be added to the 64-bit base register, with optional scaling of the index by the access size. Additionally it allows for sign-extension or zero-extension of a 32-bit value within an index register, followed by optional scaling.
- PC-relative addressing
  - Support for position-independent code and data addressing.
  - PC-relative literal loads have an offset range of ± 1MB.
  - Compare-based conditional branches have a range of ± 1MB.
  - Test bit conditional branches have a range of ± 32KB.
  - Unconditional branches, including branch and link, have a range of ± 128MB.

Armv8: Load/Store Addressing Modes

- 64-bit base address from a GPR (X0-X30) or SP.
- Optional immediate or register offset.
- Addressing modes
  - Base register only (no offset): `[base{, #0}]`
  - Base plus offset
    - Immediate: `[base{, #imm}]`
    - Register: `[base, Xm{, LSL #imm}]`
    - Extended Register: `[base, Wm, (S|U)XT(X|W){#imm}]`
  - Pre-indexed: `[base, #imm]!`
  - Post-indexed: `[base], #imm`
  - Literal (PC-relative): `label`

Armv8: Procedure call/return

- A64 instructions
  - Branch with Link: `BL <label>`
    - Branches to a PC-relative offset, setting the register `X30` to `PC+4`.
  - Branch with Link to Register: `BLR <Xn>`
    - Calls a subroutine at an address in a register, setting register `X30` to `PC+4`.
  - Return from subroutine: `RET {<Xn>}`
    - Branches unconditionally to an address in a register.
    - Register defaults to `X30` if absent.
- Parameter passing and result return
  - Arguments passed in GPRs `R0` through `R7` and on the stack (left-to-right).
  - Primitive results returned in `R0`. Otherwise, the caller reserves block of memory for result and passes the address to the callee in `X8`.

[Source: Arm® Architecture Reference Manual Armv8, for Armv8-A architecture profile ARM DDI 0487F.c (ID072120), issued 2020-07-17.]

[Source: Procedure Call Standard for the Arm® 64-bit Architecture (AArch64) 2021Q1, issued 2021-04-12.]