ARM Notes for CPEN 212

Tom Wang

Spring 2024

1 Registers

1.1 General Purpose Registers

We have integer registers, and floating point registers.

There are 31 integer registers, each 64 bits wide named as X0 to X30. The first 32 bits can be accessed as W0 to W30. Operations on w or X would correspond to 32 or 64 bit operations respectively. When w is written, the upper 32 bits of X are set to 0.

Example of addition:

```
ADD W0, W1, W2 //32 bit addition
ADD X0, X1, X2 //64 bit addition
```

There are 32 floating point registers, each 128 bits wide named as Q0 to Q31. There are ways to access the lower bits of them. B0 to B31 accesses the lower 8 bits, H0 to H31 accesses the lower 16 bits, S0 to S31 accesses the lower 32 bits. D0 to D31 accesses the lower 64 bits.

Example of addition:

```
FADD B0, B1, B2 //8 bit addition
FADD H0, H1, H2 //16 bit addition
FADD S0, S1, S2 //32 bit addition
FADD D0, D1, D2 //64 bit addition
FADD Q0, Q1, Q2 //128 bit addition
```

These registers can also be referred to as V registers. When the V form is used, the register is treated as being a vector. This means that it is treated as though it contains multiple independent values, instead of a single value. This example performs vector floating point addition:

```
FADD V0.4S, V1.4S, V2.4S //4 32 bit additions
FADD V0.2D, V1.2D, V2.2D //2 64 bit additions
//or integer addition
ADD V0.4S, V1.4S, V2.4S //4 32 bit additions
ADD V0.2D, V1.2D, V2.2D //2 64 bit additions
```

1.2 Special Purpose Registers

- The zero register always contains the value 0. It is read-only. They are XZRR and WZR. They ignore writing.
- Stack pointer (SP) can be used to access the stack. Armv8-A has multiple stack pointers, and each one is associated with a specific Exception level. When SP is used in an instruction, it means the current stack pointer.
- X30 is used as the Link Register and can be referred to as LR. Separate registers, ELR_ELx, are used for returning from exceptions.
- The Program Counter (PC) is not a general-purpose register in A64, and it cannot be used with data processing instructions. The PC can be read using:

  ```
  ADR X0, .
  //PC is the address of the current instruction
  ```

  '.' refers to the current location in the program. The PC can be modified using:

  ```
  B . //Branch to the current location
  ```

1.3 System Registers

System registers are used to configure the processor and to control systems such as the MMU and exception handling.

System registers cannot be used directly by data processing or load/store instructions. Instead, the contents of a system register need to be read into an X register, operated on, and then written back to the system register. There are two specialist instructions for accessing system registers:

- MRS (Move to Register from System register) copies the contents of a system register into an X register.
- MSR (Move to System register from Register) copies the contents of an X register into a system register.

System register names end with _ELx. The _ELx specifies the minimum privilege necessary to access the register. Attempting to access the register with insufficient privilege results in an exception.

1.4 System calls

System calls cause an exception which allows controlled entry into a more privileged Exception level.

- SVC Supervisor call causes an exception targeting EL1. Used by an application to call the OS.
- HVC Hypervisor call causes an exception targeting EL2. Used by an OS to call the hypervisor, not available at EL0.
- SMC Secure monitor call causes an exception targeting EL3. Used by an OS or hypervisor to call the EL3 firmware, not available at EL0.