The AArch64 processor (aka arm64), part 2: Extended register operations

devblogs.microsoft.com/oldnewthing/20220727-00

July 27, 2022

Raymond Chen

There are a number of places where the instruction set permits the value in a register to be transformed before it is used. The set of valid transforms vary from instruction to instruction, but they share a common syntax.

	Operand	Meaning		
Shifted	`Rn/zr, LSL #n`	Logical shift left.		
	`Rn/zr, LSR #n`	Logical (unsigned) shift right.		
	`Rn/zr, ASR #n`	Arithmetic (signed) shift right.		
Extended	`Wn/sp, UXTB #n`	Unsigned	extend low byte	shifted left.
	`Wn/sp, UXTH #n`	Unsigned	extend low halfword	shifted left.
	`Wn/sp, UXTW #n`	Unsigned	extend low word	shifted left.
	`Xn/sp, UXTX #n`	Unsigned	extend low doubleword	shifted left.
	`Wn/sp, SXTB #n`	Signed	extend low byte	shifted left.
	`Wn/sp, SXTH #n`	Signed	extend low halfword	shifted left.
	`Wn/sp, SXTW #n`	Signed	extend low word	shifted left.
	`Xn/sp, SXTX #n`	Signed	extend low doubleword	shifted left.

The `LSL`, `LSR`, and `ASR` transformations are formally known as *shifted registers*. They take a value in a register and shift it.

The extend+shift transformations are formally known as *extended registers*. They extract a subset of the source register, extend it either as a signed or unsigned value to the full operand size, and then shift the extended result.

Shifting the zero register isn't particular useful since you still get zero, but the instruction encoding lets you do it. Similarly, there is no practical difference between `UXTX` and `SXTX` (unsigned and signed extension of the low doubleword of a 64-bit register) since the low doubleword of a 64-bit register is *the whole register*.

For extended registers, the assembler lets you omit the shift amount, in which case it defaults to zero. The shift amount is not optional for the shifted registers.

Before you get all excited about the possibilities, know that not all instructions support all of these transformations, and for the ones that they do, the shift amounts are limited. We'll look at the restrictions as they arise. For now, I just wanted to introduce the concepts.

As a convenience, if you are using an instruction that accepts only extended registers, but you want to use a `LSL`, you can write `LSL #n`, and the assembler will autoconvert it to `UXTW #n` or `UXTX #n`, depending on the operand size.

Next time, we'll start putting these transforms to use when we look at addressing modes.

Raymond Chen

Follow