COL216 Computer Architecture

Lab Assignment 2 Stage 6: Implement all multiply group instructions

Multiply group instructions

The instructions to be implemented at this stage are as follows. For short multiply instructions producing 32-bit results (group a), there are no separate signed and unsigned versions unlike long multiply instructions that produce 64-bit results (group b). Lec09, slides 3-8 explain the reason.

a)	`mul`	multiply with 32-bit output	32x32 => 32
	`mla`	multiply-accumulate with 32-bit output	32x32+32 => 32
b)	`umull`	unsigned multiply with 64-bit output	32x32 => 64
	`umlal`	unsigned multiply-accumuate with 64-bit output	32x32+64 => 64
	`smull`	signed multiply with 64-bit output	32x32 => 64
	`smlal`	signed multiply-accumuate with 64-bit output	32x32+64 => 64

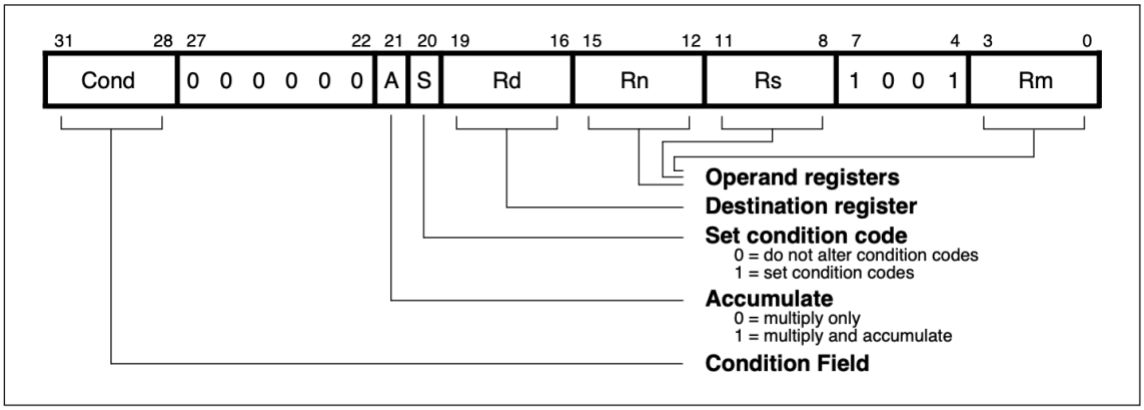
Implementation approach

Slides 9-13 of Lec09 describe how to express signed and unsigned multiplication in VHDL and how to combine the two efficiently. The addition part of multiply-accumulate operations can be done using ALU. However, for 64-bit multiply-accumulate, it would involve two cycles. For the present exercise, a simpler approach may be followed, that is, an independent multiply-accumulate module with 64-bit result may be designed for implementing all the six instructions mentioned above, without involving ALU. This module would contain a combined signed/unsigned multiplier as described in slide 13 of Lec09, along with a 64 bit adder.

Instruction formats

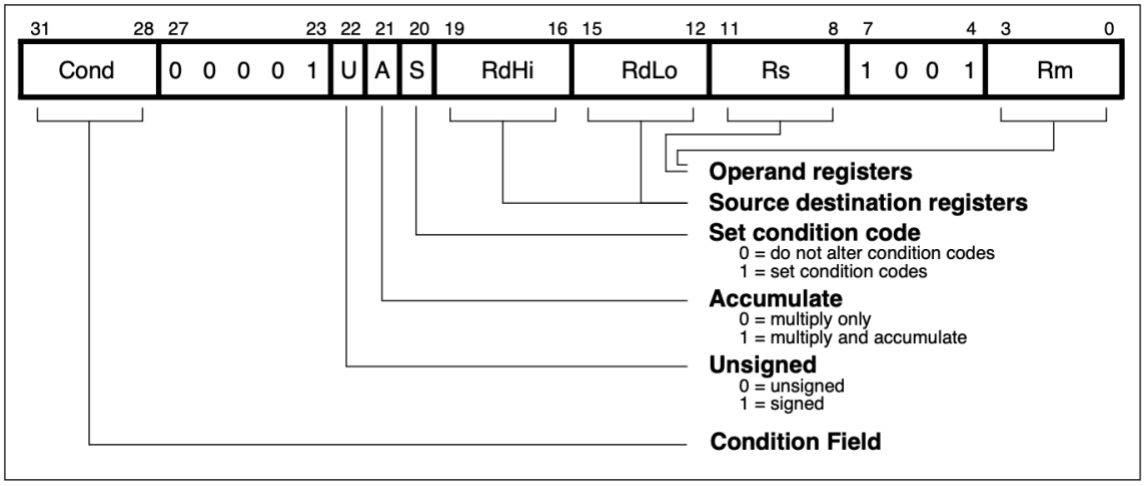
(a) Format for short multiply instructions is shown below. Rn field should be 0 for `mul`.

`mul`: Rd ← Rm * Rs `mla`: Rd ← Rm * Rs +Rn



(b) Format for long multiply instructions is shown below. Here Rd denotes a 64-bit register formed by RdHi and RdLo.

`smull,umull`: Rd ← Rm * Rs `smlal,umlal`: Rd ← Rm * Rs + Rd



The above two formats are distinguished from each other by bit 23 of the instruction, '0' for 32-bit result and '1' for 64-bit result. For the latter, bit 22 specifies unsigned | signed variant, whereas for the former, this bit is '0'. Another point to be noted is that these formats have F field = "00", I-bit = '0', bit 7 = '1' and bit 4 = '1' like half-word and signed DT instructions and some other instructions like `swp` and `bex`.

Decoding instructions

The chart below shows how these instructions can be distinguished from each other. Bits shown as 'x' carry some information relevant to the instruction and are not required for decoding purpose. In fact, only the bits shown in bold face are required for decoding. Instructions `swp` and `bex` are beyond the scope of the current exercise. These are shown in the chart only for the sake of completeness.

Instruction groups	bits 24-20	bits 6-5 S H
Short multiply	**0 0** 0 x x	**0 0**
Long multiply	**0 1** x x x	
Swap (`swp`)	**1 0** x **0** 0	
Branch & exchange (`bex`)	**1 0** 0 **1** 0	
Unsigned half-word load/store	x x x x x	**0 1**
Signed byte load	x x x x 1	**1 0**
Signed half-word load	x x x x 1	**1 1**