10.15 Instructions unique to M32R

(Original section[1])

The most unusual feature of the M32R is a slight VLIW approach to the pairs of 16-bit instructions. A bit is reserved in the first instruction of the pair to say whether this instruction can be executed in parallel with the next instruction—that is, the two instructions are independent—or if these two must be executed sequentially. (An earlier machine that offered a similar option was the Intel i860.) This feature is included for future implementations of the architecture.

One surprise is that all branch displacements are shifted left 2 bits before being added to the PC, and the lower 2 bits of the PC are set to 0. Since some instructions are only 16 bits long, this shift means that a branch cannot go to any instruction in the program: it can only branch to instructions on word boundaries. A similar restriction is placed on the return address for the branch-and-link and jump-and-link instructions: they can only return to a word boundary. Thus, for a slightly larger branch distance, software must ensure that all branch addresses and all return addresses are aligned to a word boundary. The M32R code space is probably slightly larger, and it probably executes more `nop` instructions than it would if the branch address was only shifted left 1 bit.

However, the VLIW feature above means that a `nop` can execute in parallel with another 16-bit instruction so that the padding doesn't take more clock cycles. The code size expansion depends on the ability of the compiler to schedule code and to pair successive 16-bit instructions; Mitsubishi claims that code size overall is only 7% larger than that for the Motorola 6800 architecture.

The last remaining novel feature is that the result of the divide operation is the remainder instead of the quotient.

(*1) This section is in original form.