Predication in ARM ISA

Suresh Purini, IIIT-H

January 29, 2012

In the ARM ISA, for almost all instructions we can add a predicate as a suffix which tells the processor to execute the corresponding instruction only if the respective conditions associated with the predicate are satisfied (refer Table)1. For example when we add the predicate MI to an ADD instruction to get an instruction of the form ADDMI, then the following things happen:

- The assembler packs the bits 0100 in the most significant nibble while construction constructing the 32-bit opcode of the ADDMI instruction. This is an *offline* task that happens at the program assembly time.
- When the processor encounters the ADDMI instruction, it executes it if and only if the N flag is equal to 1 (or set in other words). This is an *online* task that happens at the program execution time.

Usually we use this predicate table in conjunction with the CMP instruction [1]. A brief description of the CMP instruction follows. When the processor encounters CMP r1, r2 instruction, it will subtract the register $r2$ from $r1$ and uses the result to affect the N, Z, C, V bits of the CPSR register as follows. Let `alu_out = r1 - r2`.

- `N = alu_out[31]`
- `Z = if alu_out == 0 then 1 else 0`
- `C = if r1 >= r2 then 1 else 0 (assuming r1 and r2 as unsigned integers)`
- `V = 1 if the result of subtraction does not fit into 32 bits else 0 (assuming r1 and r2 as signed integers)`

Finally `alu_out` will be discarded without storing it in any register.

After the CMP instruction, we have to use the right predicate suffix to indicate the condition we would like to test. For example if we treat r1 and r2 as unsigned integers and want to check if $r1 <= r2$, then we have to use the suffix LS (for example BLS). However if treat r1 and r2 as signed integer to test the same condition we use the suffix LE (for example BLE). As assembly language programmers this much knowledge is sufficient for us to write our programs correctly. However as processor designers and computer scientists, we are interested in checking whether the condtions mentioned in the Column 3 and Column 4 of the Table 1 are equivalent or not. For example is the following statement true after the exectuion of the CMP r1, r2 instruction.

$$r1 \geq r2 \ \text{ (signed)} \iff N = V$$

Question: Check the equivalence of the conditions in Column 3 and Column 4 of the Table 1 are equivalent (after the execution of the CMP instruction).

[1] We can use these predicates in conjunction with other instructions also. For example SUBS instruction followed by ADDEQ.

Opcode [31:28]	Extension	Interpretation	Status flag state for execution
0000	EQ	Equal/equals zero	Z set
0001	NE	Not equal	Z clear
0010	CS/HS	Carry set/unsigned higher or same	C set
0011	CC/LO	Carry clear/unsgined lower	C clear
0100	MI	Minus/negative	N set
0101	PL	Plus/positive or zero	N clear
0110	VS	Overflow	V set
0111	VC	No overflow	V clear
1000	HI	Unsigned higher	C set and Z clear
1001	LS	Unsigned lower or same	C clear or Z set
1010	GE	Signed greater than or equal	N equals V
1011	LT	Signed less than	N is not equal to V
1100	GT	Signed greater than	Z clear and N equals V
1101	LE	Signed less than or equal	Z set or N is not equal to V
1110	AL	Always	any
1111	NV	Never (do not use!)	none

Table 1: ARM Predicate Table