- Compiler support needed (if-conversion)
- ISA support needed (conditional move instructions)
- Used for dealing w/ "control hazards"
  (Hazards caused by control dependencies)

Lesson 5

Predication

Some branches are hard to predict, such as small if-then-else statements. For these instances predication should be used.

Predication is doing the work of both directions and the choosing the correct one and throwing away the incorrect path work.

Loops - prediction is better than predication
Function Calls and Returns - prediction is better than predication
Large If-then-Else - prediction is better than predication
Small If-the-Else - predication is better than prediction, unless the predictor is very accurate

To convert if-then-else statements use Conditional Instructions. Conditional instructions are instructions that do two functions atomically, such as MOVC.
MOVC

1. compiler support is required to convert if-then-else.
2. will remove hard-to-predict branches
3. more registers are needed
4. more instructions are executed

(#3 and #4 require extensive hardware support to achieve full predication)

MOVZ Rd, Rs, Rt → if (Rt == 0) Rd = Rs
MOVN Rd, Rs, Rt → if (Rt != 0) Rd = Rs

X = cond? X1 : X2 → if-conversion →
R3 = cond
R1 = X1
R2 = X2
MOVN X, R1, R3
MOVZ X, R2, R3

Full Predication HW Support

Predication bits need to be added to every instruction. These bits tell the processor where the qualifying predicate can be found.