コード生成の基本方針 (O-o-O vs. in-order編)

今田俊寛

バックグラウンド

- 世の中には基本的に2種類のハードウェアがある
  - in-order
    - 実行順序保証
  - O-o-O (Out of Order)
    - 命令を結果が合う範囲で並べ替える
  - (Other)
    - 両者の混合

コード生成する時・・・

- in-orderとO-o-Oで決定的に違う事とは。
  - in-order
    - ソフトウェアが命令レベル並列性(ILP)を担保
  - O-o-O
    - ハードウェアがソフトウェアと協調してILPを担保

具体的には・・・

- O-o-Oでは、可能な限りライブレンジを切る
  - レジスタプレッシャは高くなるが、問題ではない。
  - 何故か？
    - レジスタプレッシャが高い＝元々ILPが高い
    - 高いILPのコードの効率的実行はハードウェアが保証する。
  - 但し、勿論限度はあって、主にROB(Re Order Buffer)のエントリ数等に依り決定される。

in-orderケース

- defineしたものはなるべく後でuseしたい(ソフトウェア視点)
- O-o-Oケースで担保していたものをソフトウェアで保証する。
- 結果としてコード生成はO-o-Oとin-orderで全く異なるものとなる。

結論

- コンパイラはハードウェア構成に依存してコード生成をしなければならない(事の一例を示した)。

終わり