LLVM Code Generator

Status and Ideas for Future Development

Evan Cheng

Apple Inc.

Lots of Love and Tender Care ...

Many Targets!

- Alpha
- ARM + Thumb
- IA64
- PowerPC
- Sparc
- X86
- Cell, MIPS, and more?

Common Passes

All are working pretty well...

- DAG Legalizer, Instruction Selector, Combiner
- Instruction Schedulers: reg-pressure and latency
- Register Allocator, Spiller, Register Scavenger
- If-converter (in development), branch folding, tail merging

Future!

- Refine existing passes
- Performance and compile time improvements
- There are some missing pieces!

Instruction Selection

- Patterns with accurate cost:

```
let AddedComplexity = 15 in
def MOVLHPSrr : PSI<0x16, MRMSrcReg, (ops VR128:$dst, VR128:$src1, VR128:$src2,
                  "movlhps {$src2, $dst|$dst, $src2}",
                  [(set VR128:$dst,
                    (v4f32 (vector_shuffle VR128:$src1, VR128:$src2,
                              MOVHP_shuffle_mask)))]>;
```

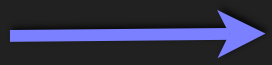
- BURG / dynamic programming algorithm
  - **Automatically picking between fp stack and sse regs where profitable!**

Instruction Selection

Whole Function ISel

```
extern void a(int);
extern void b(int);

void t(short p) {
  if (p == 3)
    a(p);
  else if (p == 4)
    b(p);
}
```



```
_t:
        ...
        cmpw $4, %ax
        je LBB1_3
LBB1_1: #entry
        cmpw $3, %ax
        jne LBB1_4
LBB1_2: #cond_true
        movswl %ax, %eax
        movl %eax, (%esp)
        call L_a$stub
        ...
LBB1_3: #cond_true9
        subl $16, %esp
        movswl %ax, %eax
        movl %eax, (%esp)
        call L_b$stub
        ...
```

Instruction Scheduling

- Focus should be on improving current heuristics
- How well would it work with “whole function isel”?
- Some targets may benefit from:
  - **Beyond-the-basic-block scheduling**
  - **Post-allocation scheduler**

Register Allocator
First Phase

- PHI elimination using dominator info
- Better live interval representation
  - **“Sane” copy coalescing**
  - **Live range splitting**
  - **Better heuristics for spill weight**

Register Allocator
Second Phase

- Proper re-materialization
- Sub-registers support
- Spiller peepholes, shrink wrapping (for prologue / epilogue)

Register Allocator
Long Term Ideas

- Alternative strategies
  - Graph coloring
  - Separate global / local allocators
- Predicate aware allocator

Missing Pieces

- Modeling condition code registers
- Tail-call
- Machine instruction level LICM
  - **May be subsumed by whole-function isel**

Conclusion

We need your participation!