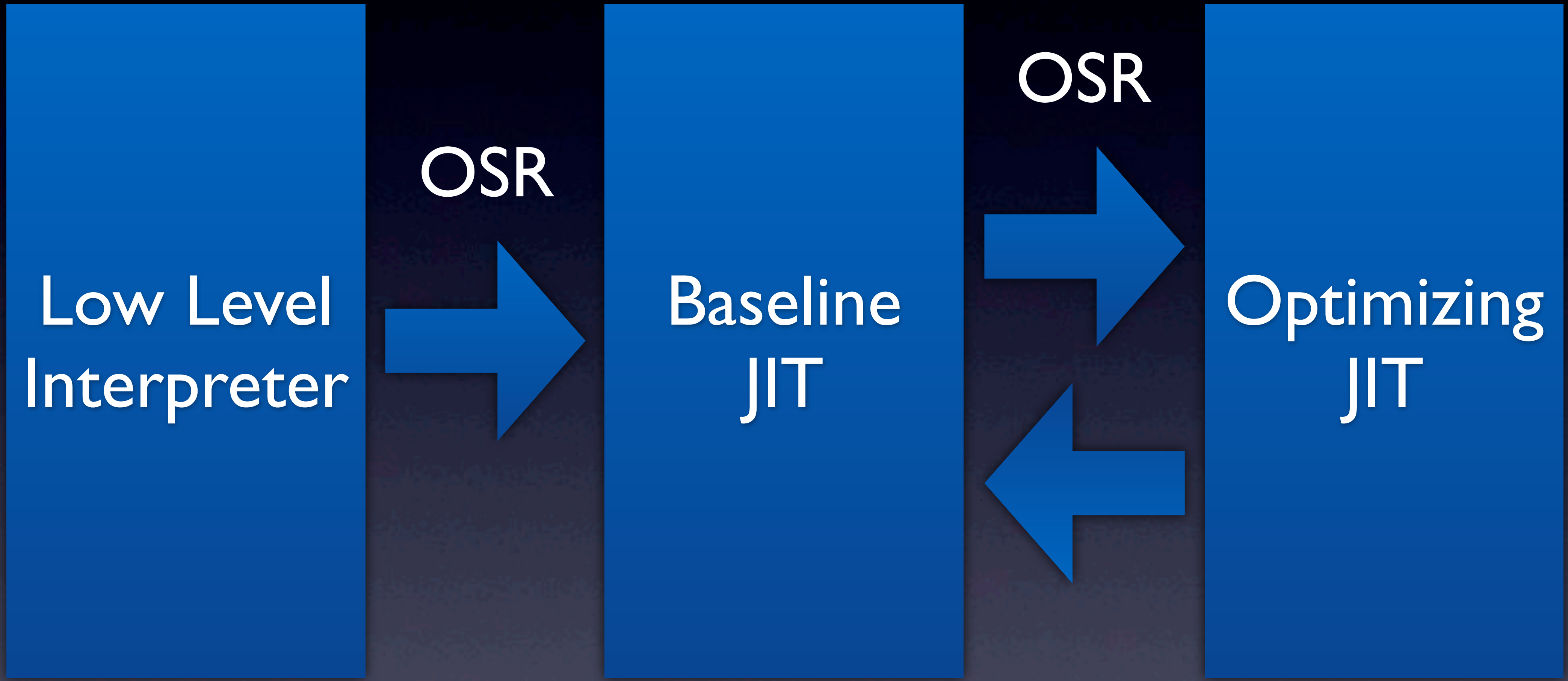
WebKit JavaScript JIT with LLVM

Filip Pizlo
Apple

WebKit

WebKit JavaScript



Low Level Interpreter

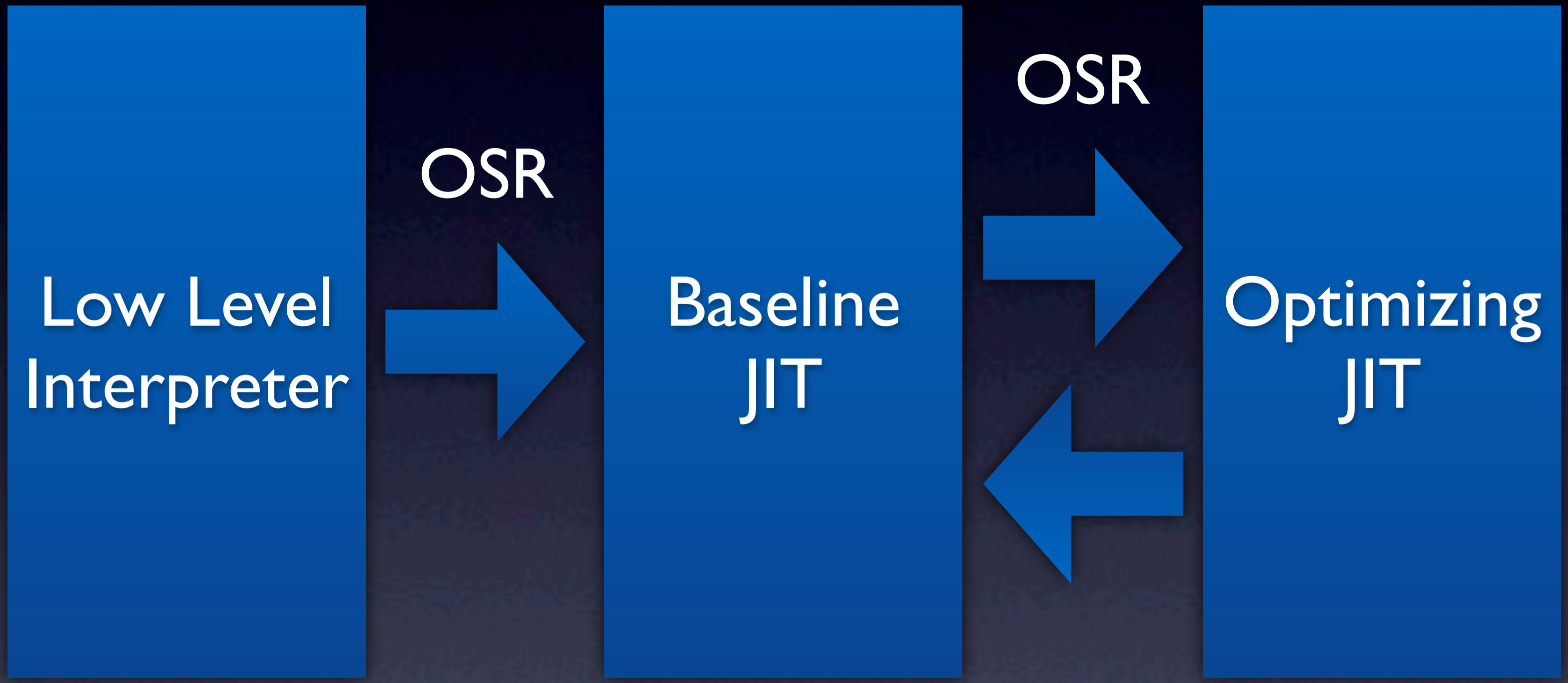
OSR

Baseline JIT

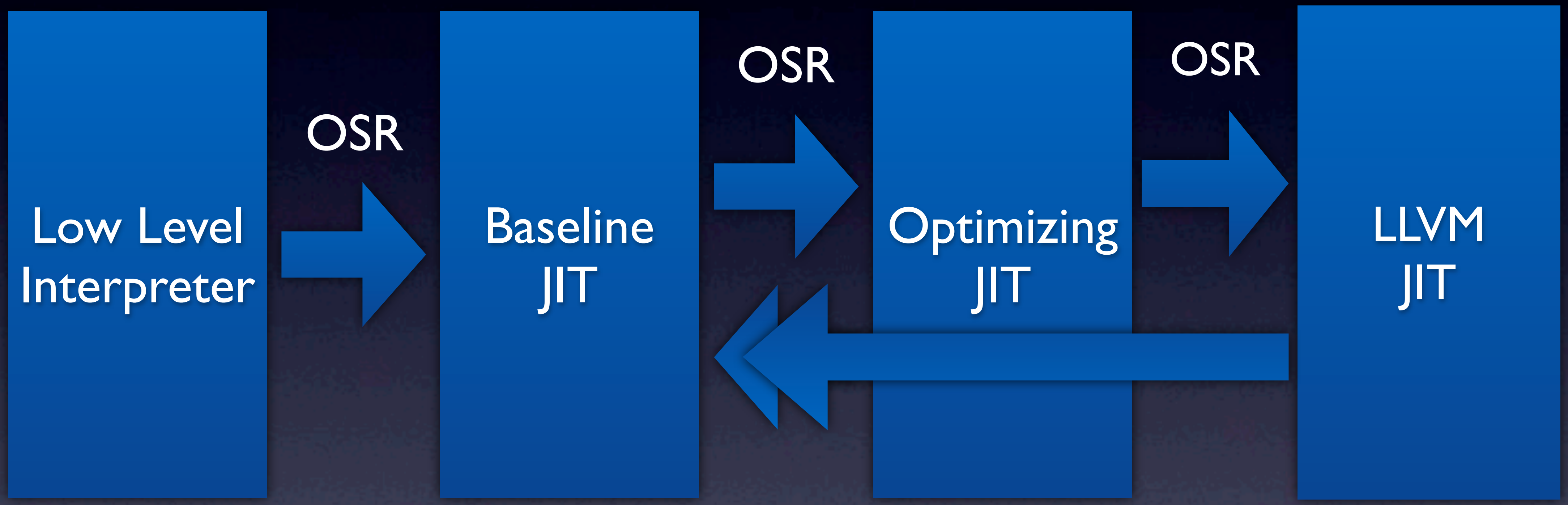
OSR

Optimizing JIT

WebKit JavaScript



WebKit JavaScript



Low Level Interpreter

OSR

Baseline JIT

OSR

Optimizing JIT

OSR

LLVM JIT

Goals

- Generate fast code
- Real world JavaScript
- Compile the hottest functions

- Deoptimization
- Inline caches
- Stronger typing

• Deoptimization	*stackmap*
• Inline caches	*patchpoint*
• Stronger typing	*TBAA on calls*

```
var x = o.f;
```

```
cmpl $0xd1e7beef, 4(%rax)
jne Lslow
movq 8(%rax), %rax
movq 0(%rax), %rax
```

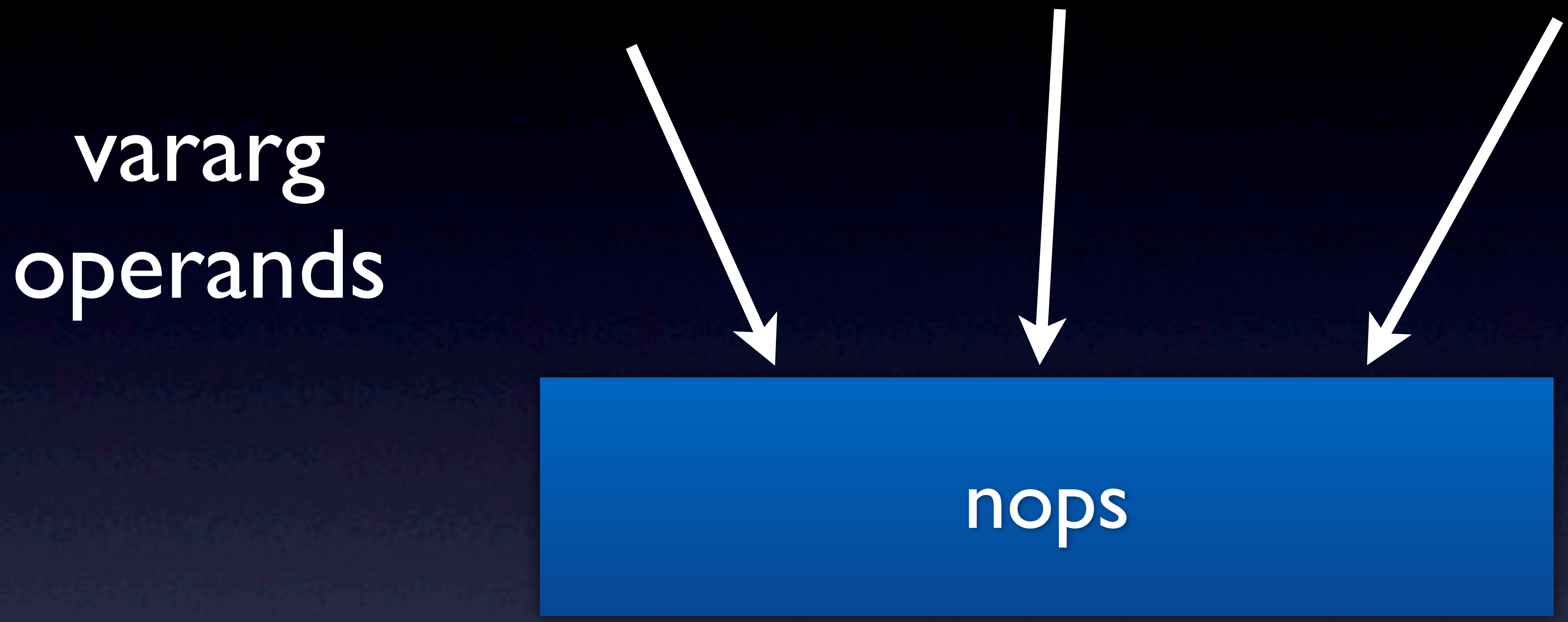
```
cmpl $42, 4(%rax)
jne Lslow
leaq 8(%rax), %rax
movq 8(%rax), %rax
```

```
cmpl $53, 4(%rax)
jne Lslow
movq 8(%rax), %rax
movq -16(%rax), %rax
```

```
jmp Lstub
<garbage>
```

- Self-modifying code
- Arbitrary effects
- Critical for performance

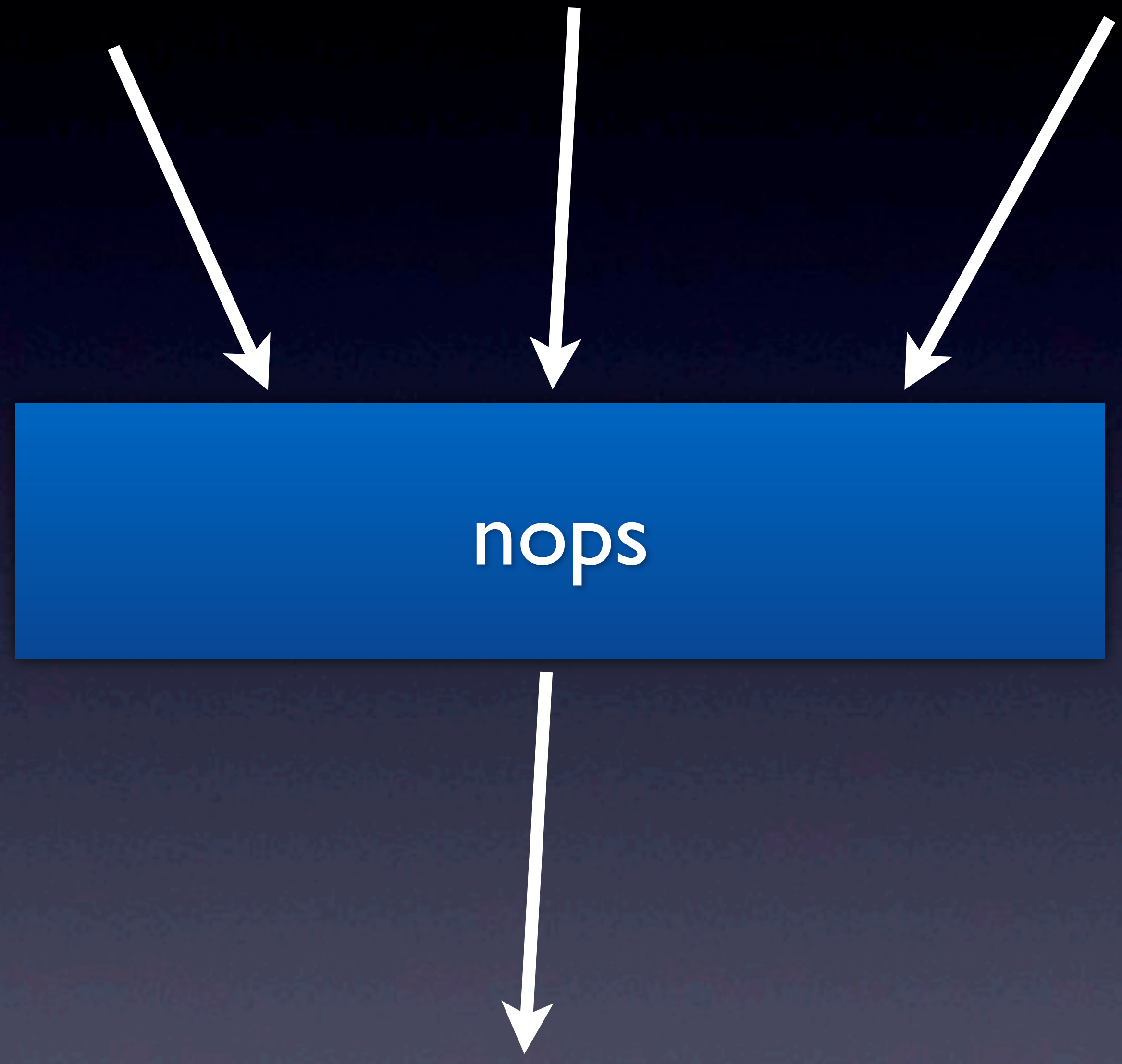
nops



vararg
operands

nops

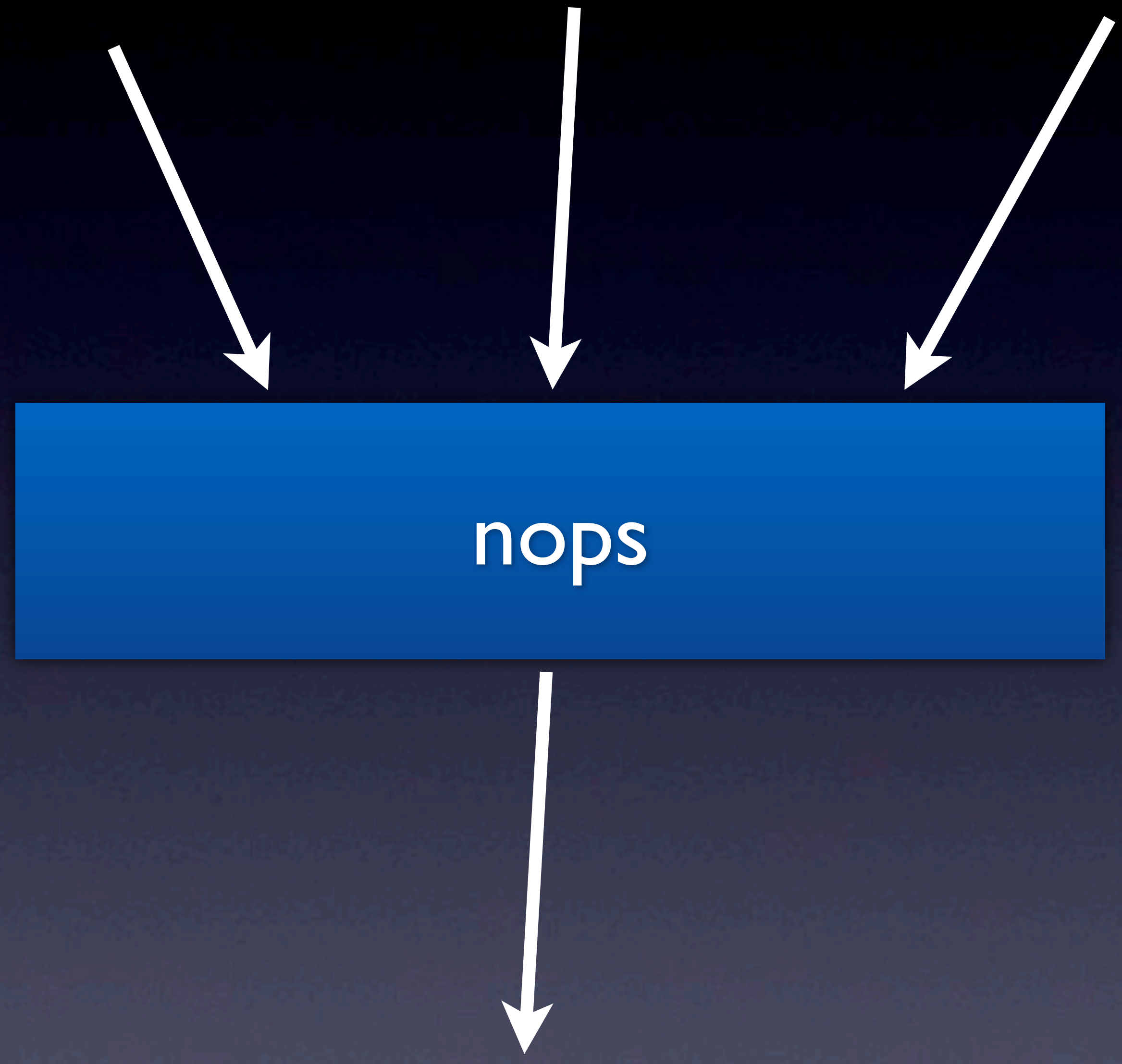
vararg
operands



nops

optional result

vararg
operands

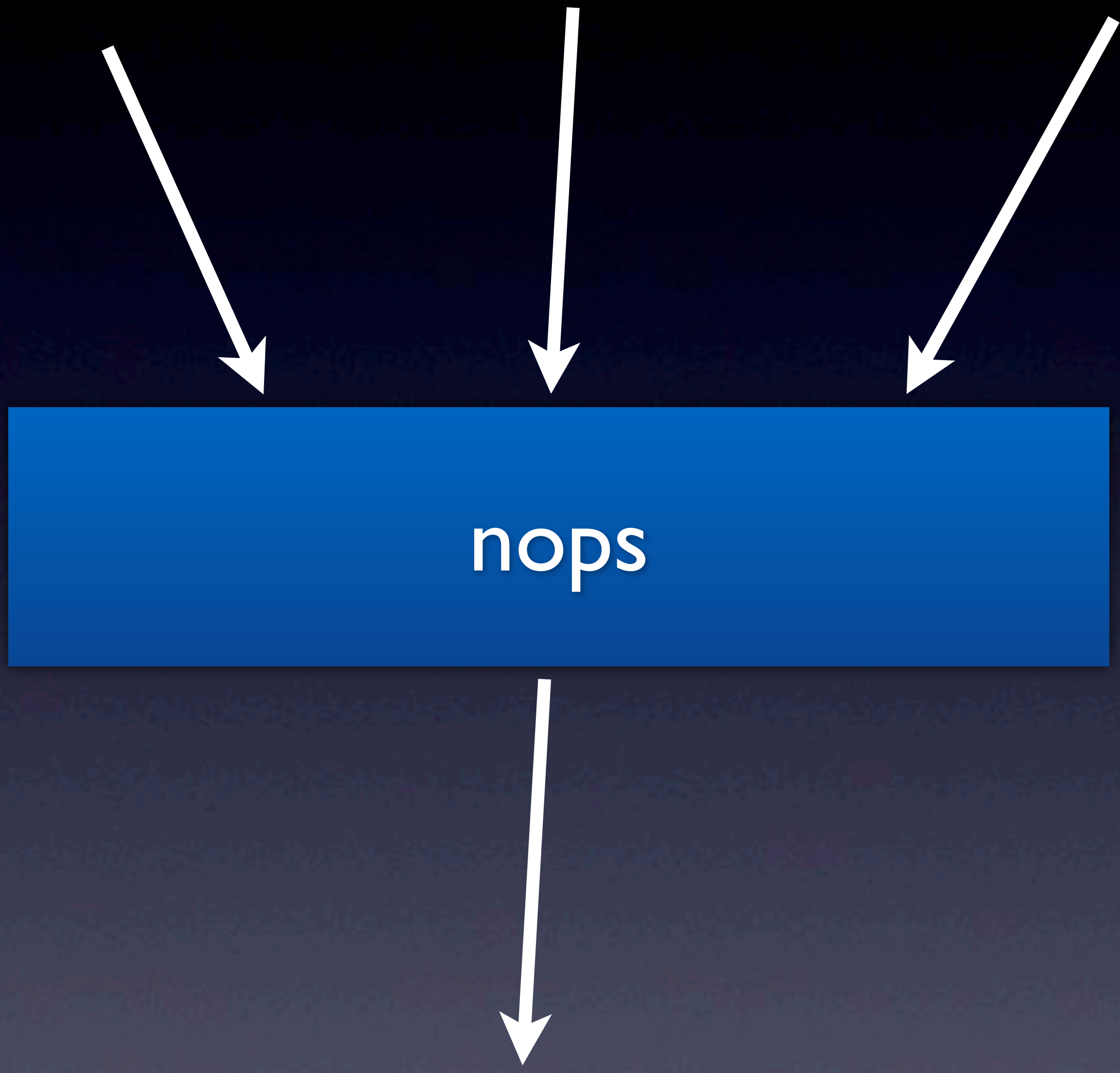


nops

call-like
effects

optional result

vararg operands



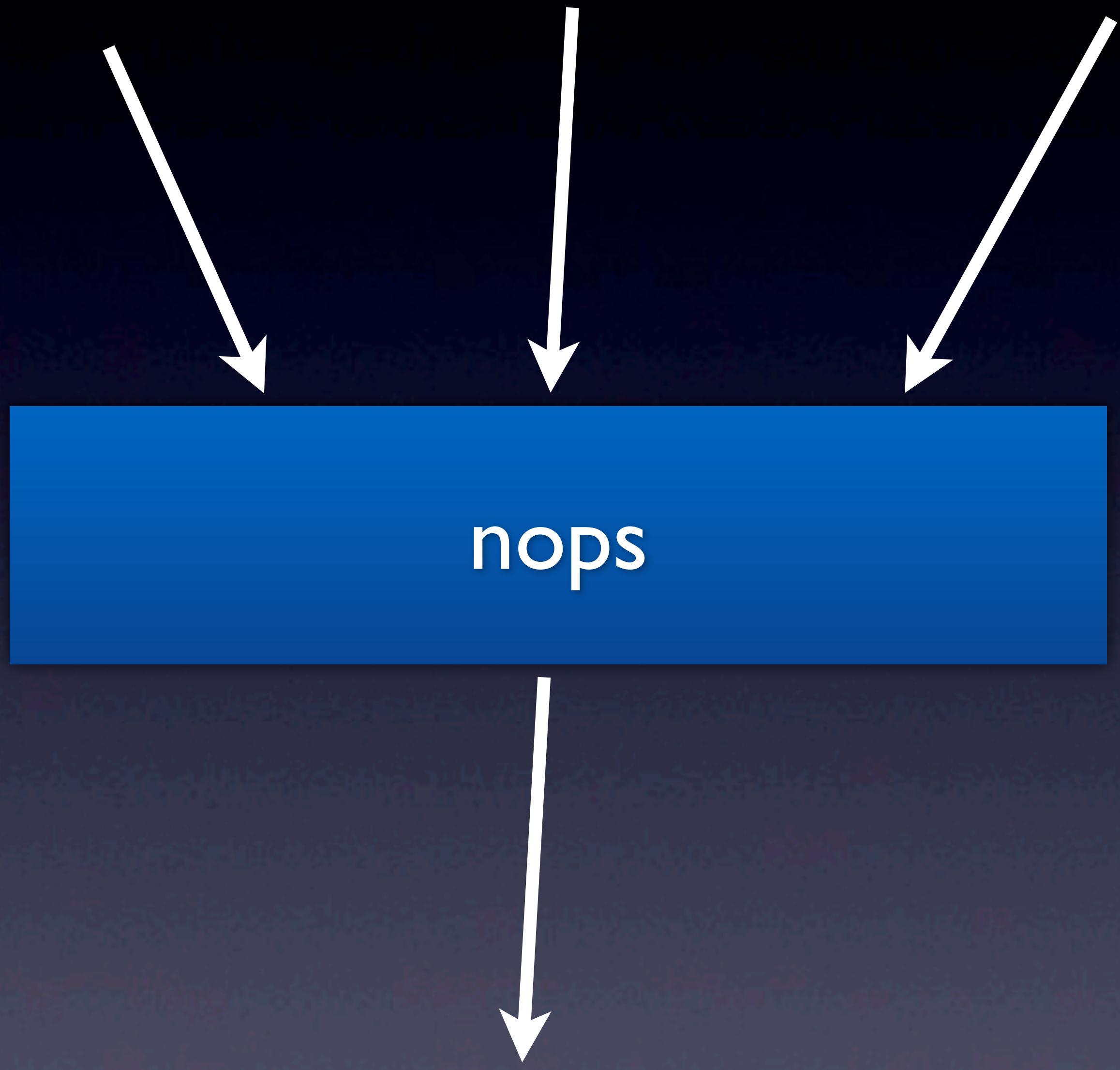
nops

Side-channel reports code offset and register state

call-like effects

optional result

Patchpoint



vararg operands

nops

call-like effects

Side-channel reports code offset and register state

optional result

• Deoptimization	*stackmap*
• Inline caches	*patchpoint*
• Stronger typing	*TBAA on calls*