

GlobalSel

Past, Present, and Future

Ahmed Bougacha
Quentin Colombet

LLVM DevMtg
October 2017

GlobalSel

Recap

GlobalSel

Recap

Revamp of our instruction selection framework

GlobalSel

Recap

Revamp of our instruction selection framework

LLVM Dev Meeting talks:

<https://www.youtube.com/watch?v=F6GGbYtae3g>

<https://www.youtube.com/watch?v=6tfb344A7w8>

GlobalSel

Recap

Revamp of our instruction selection framework

LLVM Dev Meeting talks:

<https://www.youtube.com/watch?v=F6GGbYtae3g>

<https://www.youtube.com/watch?v=6tfb344A7w8>

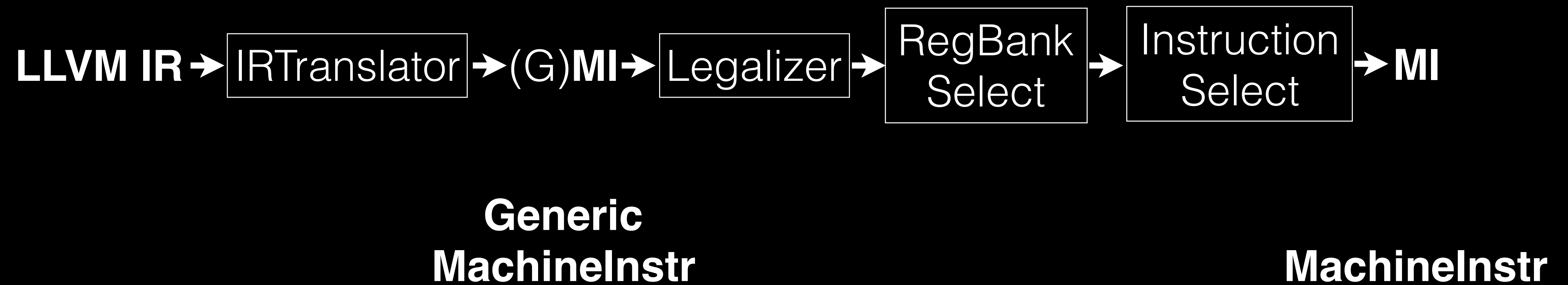
LLVM Dev RFCs:

<http://lists.llvm.org/pipermail/llvm-dev/2013-August/064696.html>

<http://lists.llvm.org/pipermail/llvm-dev/2015-November/092566.html>

GlobalSel

Pipeline



Past

Generic MachineInstr

More Concise Type System

Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%2( _, s64 ) = G_PTRTOINT %0  
%3( _, s64 ) = G_ADD %2, %1  
%4( _, p0 ) = G_INTTOPTR %3  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%2( _, s64 ) = G_PTRTOINT %0  
%3( _, s64 ) = G_ADD %2, %1  
%4( _, p0 ) = G_INTTOPTR %3  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%2( _, s64 ) = G_PTRTOINT %0  
%3( _, s64 ) = G_ADD %2, %1  
%4( _, p0 ) = G_INTTOPTR %3  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0(,p0) = COPY %x0  
%1(,s64) = G_CONSTANT 4  
%2(,s64) = G_PTRTOINT %0  
%3(,s64) = G_ADD %2, %1  
%4(,p0) = G_INTTOPTR %3  
%5(,s32) = G_LOAD %4(,p0)
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%2( _, s64 ) = G_PTRTOINT %0  
%3( _, s64 ) = G_ADD %2, %1  
%4( _, p0 ) = G_INTTOPTR %3  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0(,p0) = COPY %x0  
%1(,s64) = G_CONSTANT 4  
%2(,s64) = G_PTRTOINT %0  
%3(,s64) = G_ADD %2, %1  
%4(,p0) = G_INTTOPTR %3  
%5(,s32) = G_LOAD %4(,p0)
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%2( _, s64 ) = G_PTRTOINT %0  
%3( _, s64 ) = G_ADD %2, %1  
%4( _, p0 ) = G_INTTOPTR %3  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%2( _, s64 ) = G_PTRTOINT %0  
%3( _, s64 ) = G_ADD %2, %1  
%4( _, p0 ) = G_INTTOPTR %3  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```


Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%4( _, p0 ) = G_GEP %0, %1  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%4( _, p0 ) = G_GEP %0, %1  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```

Alignment

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT ~3  
%2( _, s64 ) = G_PTRTOINT %0  
%3( _, s64 ) = G_AND %2, %1  
%4( _, p0 ) = G_INTTOPTR %3
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT 4  
%4( _, p0 ) = G_GEP %0, %1  
%5( _, s32 ) = G_LOAD %4( _, p0 )
```

Alignment

```
%0( _, p0 ) = COPY %x0  
%1( _, s64 ) = G_CONSTANT ~3  
%2( _, s64 ) = G_PTRTOINT %0  
%3( _, s64 ) = G_AND %2, %1  
%4( _, p0 ) = G_INTTOPTR %3
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0(,p0) = COPY %x0  
%1(,s64) = G_CONSTANT 4  
%4(,p0) = G_GEP %0, %1  
%5(,s32) = G_LOAD %4(,p0)
```

Alignment

```
%0(,p0) = COPY %x0  
%4(,p0) = G_PTR_MASK %0, 2
```

Generic MachineInstr

More Concise Type System

Displacement

```
%0(,p0) = COPY %x0
%1(,s64) = G_PTRTOINT %0
%2(,s64) = G_CONSTANT 4
%3(,s64) = G_ADD %1, %2
%4(,p0) = G_INTTOPTR %3
%5(,s32) = G_LOAD %4(,p0)
```

```
%0(,p0) = COPY %x0
%1(,s64) = G_CONSTANT 4
%4(,p0) = G_GEP %0, %1
%5(,s32) = G_LOAD %4(,p0)
```

Alignment

```
%0(,p0) = COPY %x0
%1(,s64) = G_PTRTOINT %0
%2(,s64) = G_CONSTANT ~3
%3(,s64) = G_AND %1, %2
%4(,p0) = G_INTTOPTR %3
```

```
%0(,p0) = COPY %x0
%4(,p0) = G_PTR_MASK %0, 2
```

Fallback to SelectionDAG

Easier Bring Up



Fallback to SelectionDAG

Easier Bring Up



A diagram showing a memory layout for function call targets. It consists of a vertical rectangle with a folded top-right corner, containing the following text:

```
@fct1  
...  
@fctI  
...  
@fctN
```

Fallback to SelectionDAG

Easier Bring Up



Fallback to SelectionDAG

Easier Bring Up

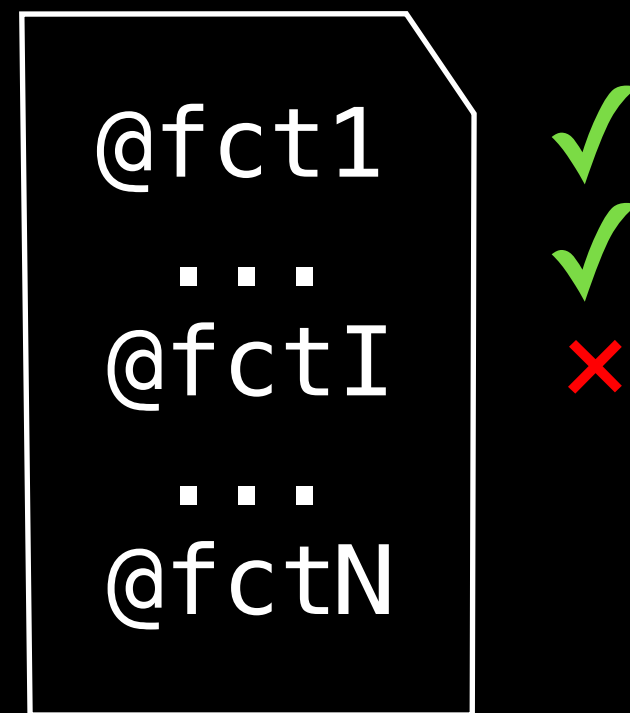


A diagram showing a function table with two green checkmarks, indicating successful compilation or selection. The table contains the following entries:

@fct1
...
@fctI
...
@fctN

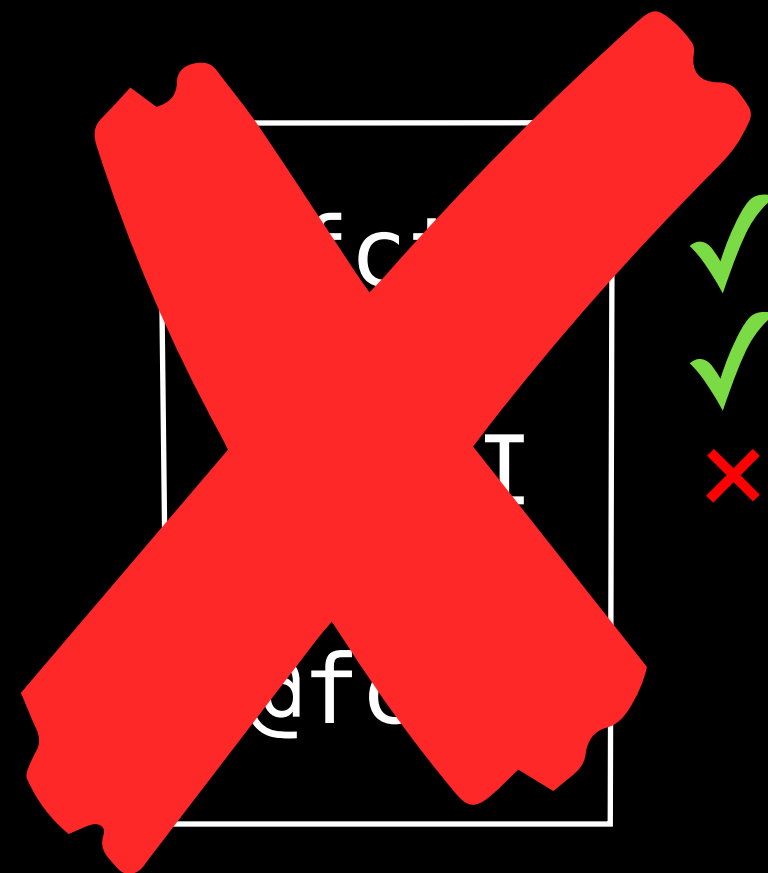
Fallback to SelectionDAG

Easier Bring Up



Fallback to SelectionDAG

Easier Bring Up



Fallback to SelectionDAG

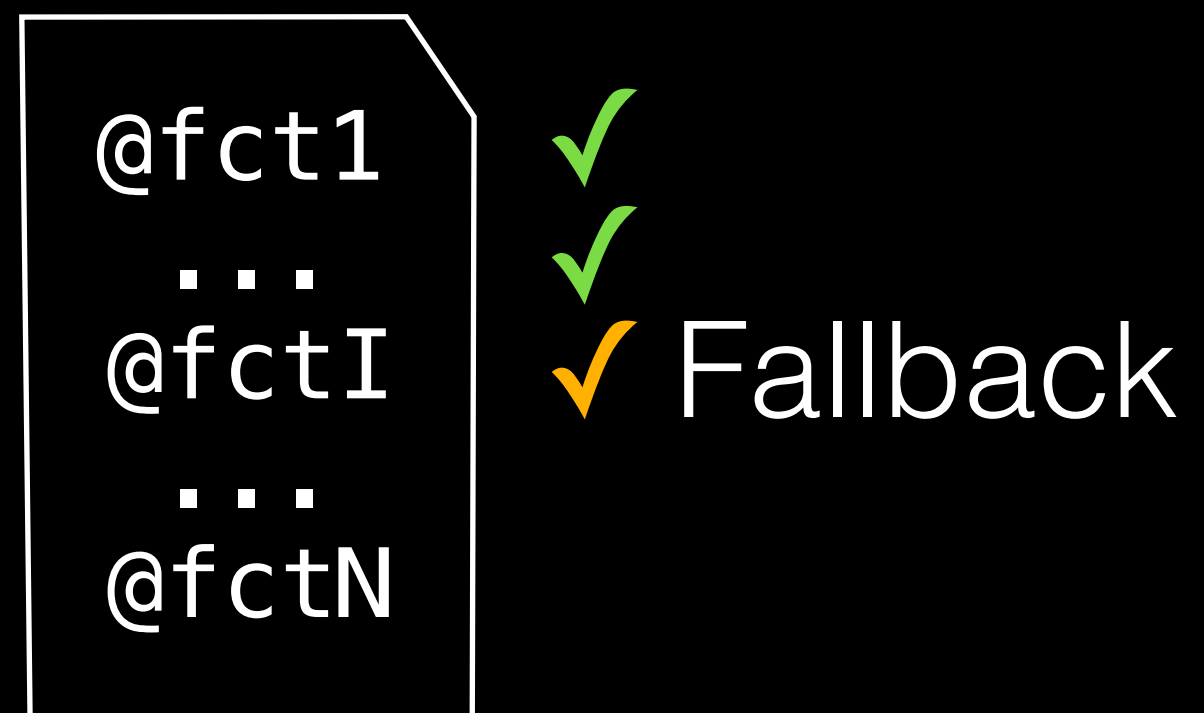
Easier Bring Up



A diagram showing a function table with entries `@fct1`, `...`, `@fctI`, `...`, and `@fctN`. To the right of the table are two green checkmarks, indicating successful compilation or verification.

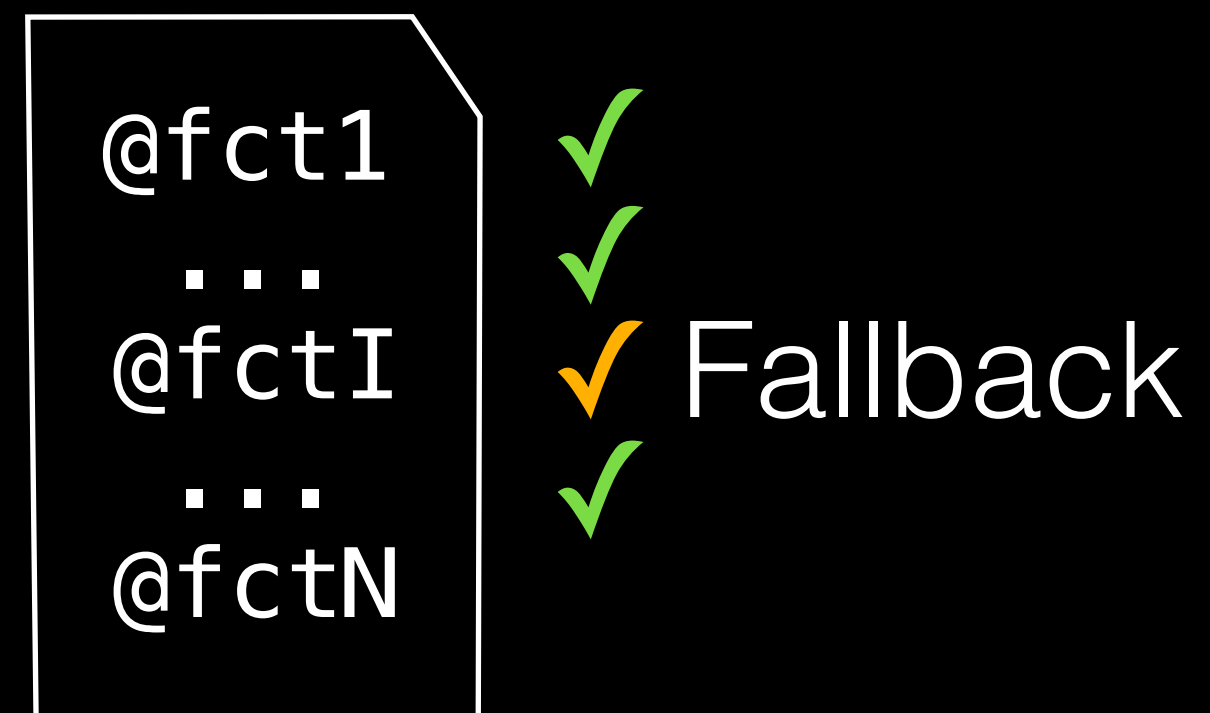
Fallback to SelectionDAG

Easier Bring Up



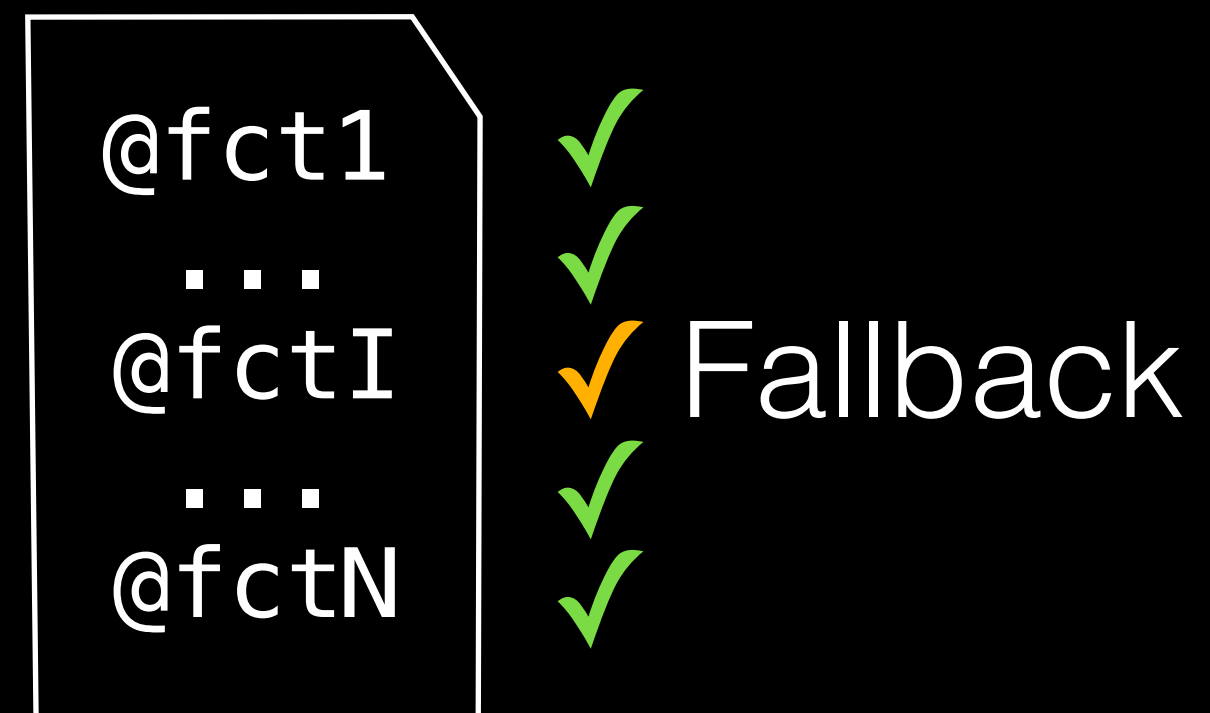
Fallback to SelectionDAG

Easier Bring Up



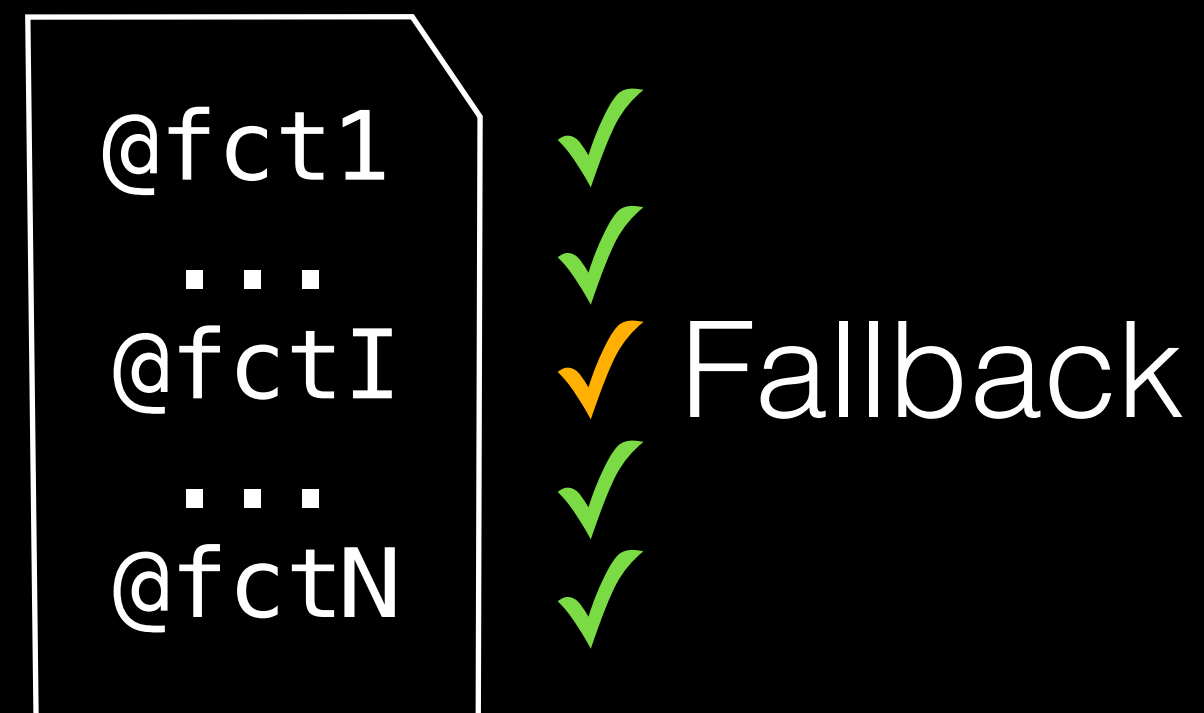
Fallback to SelectionDAG

Easier Bring Up



Fallback to SelectionDAG

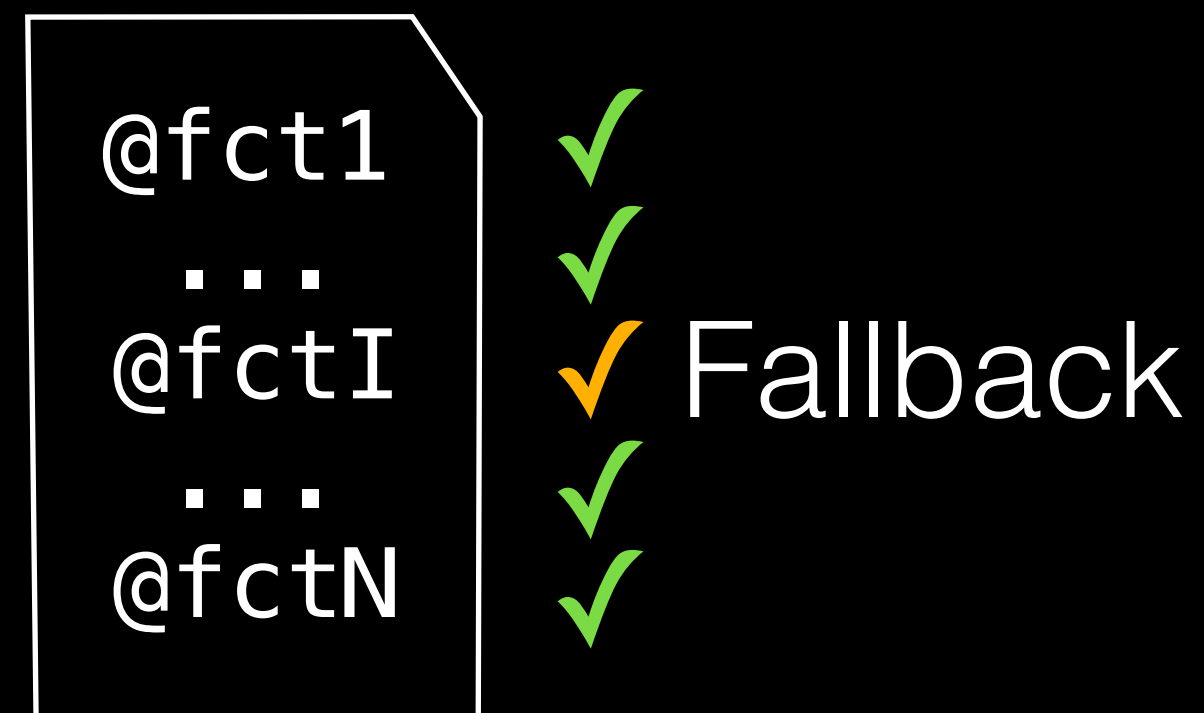
Easier Bring Up



`-global-isel-abort=0` or `-global-isel-abort=2` (for warnings)

Fallback to SelectionDAG

Easier Bring Up



`-global-isel-abort=0` or `-global-isel-abort=2` (for warnings)

Fallback to SelectionDAG

Easier Bring Up



fctI:

Fallback to SelectionDAG

Easier Bring Up



fctI:

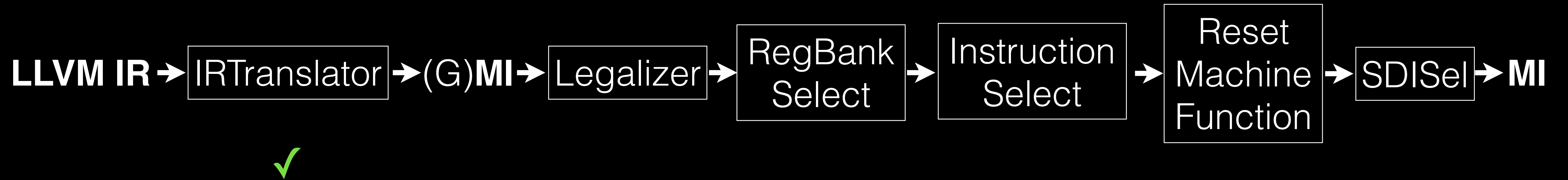
... = op1 ...

...

... = opN ...

Fallback to SelectionDAG

Easier Bring Up



fctI:

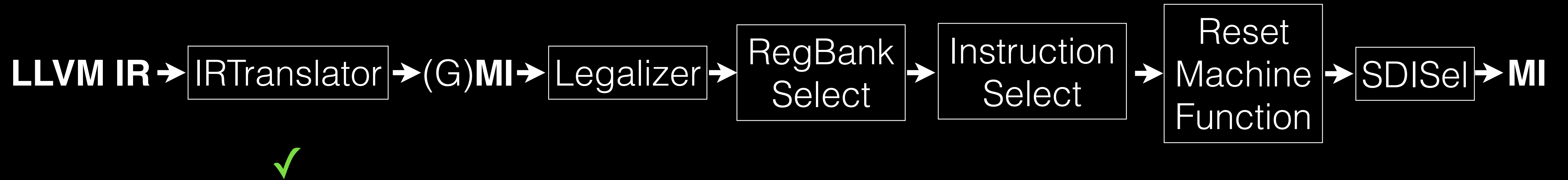
... = op1 ...

...

... = opN ...

Fallback to SelectionDAG

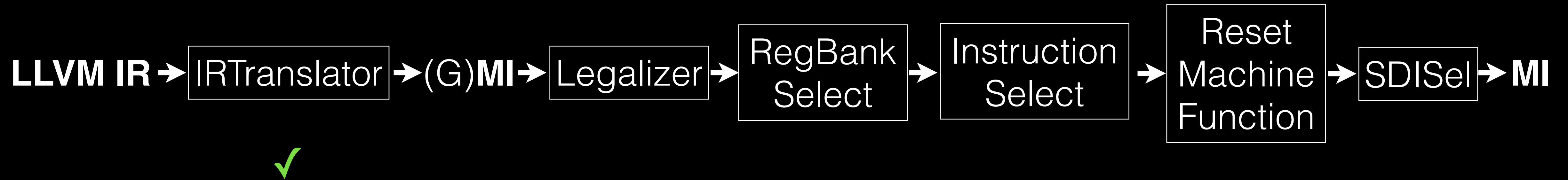
Easier Bring Up



```
fctI:
... = op1 ...
...
... = opN ...
```

Fallback to SelectionDAG

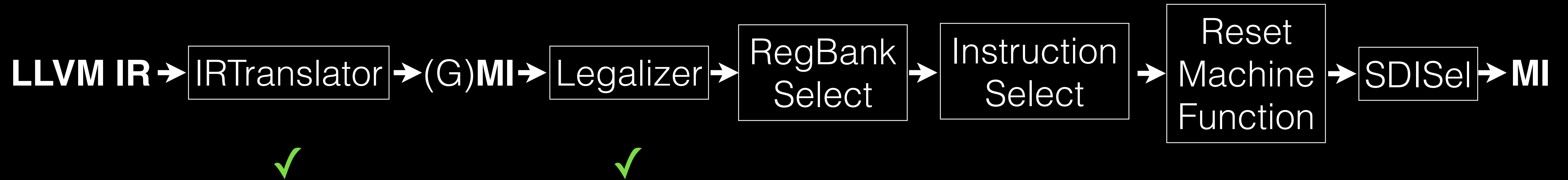
Easier Bring Up



```
fctI:
... = leg0p1 ...
...
... = leg0pN ...
```

Fallback to SelectionDAG

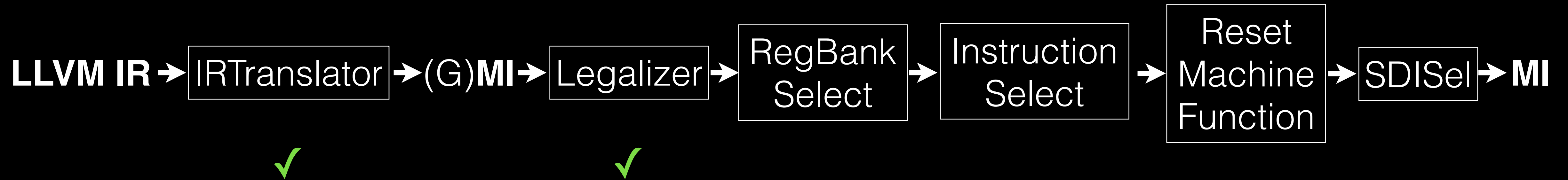
Easier Bring Up



```
fctI:
... = leg0p1 ...
...
... = leg0pN ...
```

Fallback to SelectionDAG

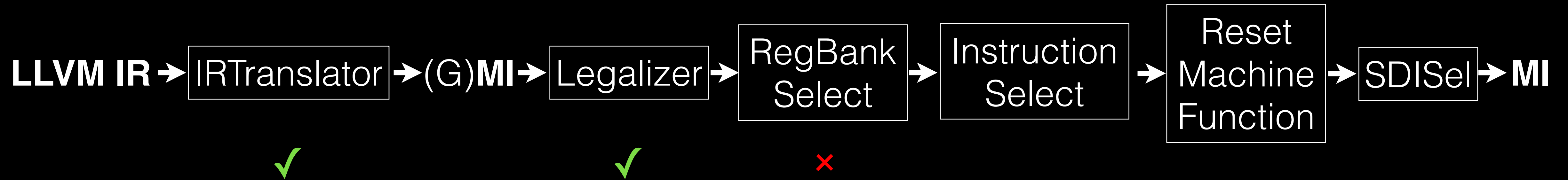
Easier Bring Up



```
fctI:
... = legOp1 ...
...
... = legOpN ...
```


Fallback to SelectionDAG

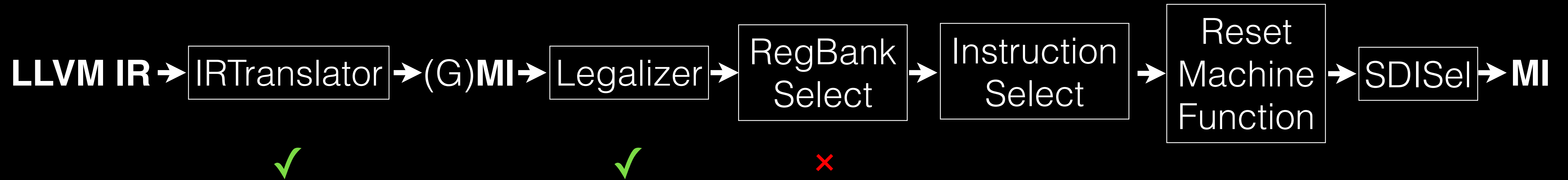
Easier Bring Up



```
fctI:
... = legOp1 ...
...
... = legOpN ...
```

Fallback to SelectionDAG

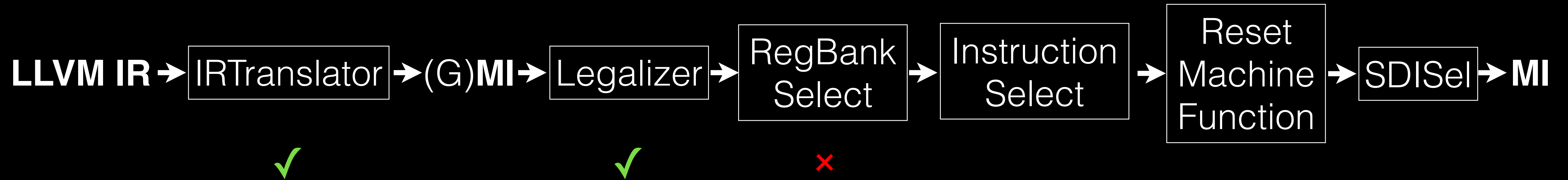
Easier Bring Up



fctI: x
... = legOp1 ...
...
... = legOpN ...

Fallback to SelectionDAG

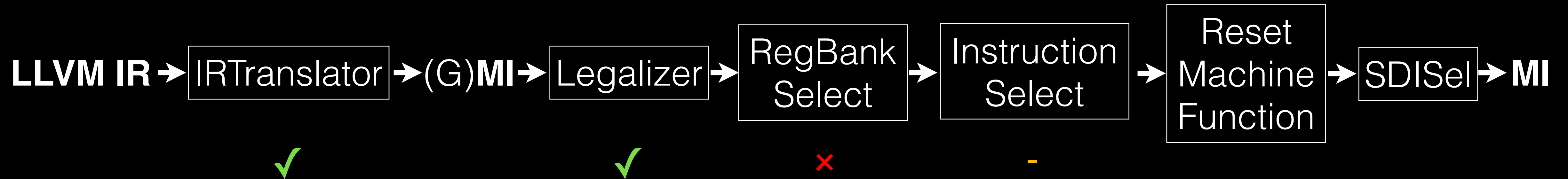
Easier Bring Up



fctI: x
... = legOp1 ...
...
... = legOpN ...

Fallback to SelectionDAG

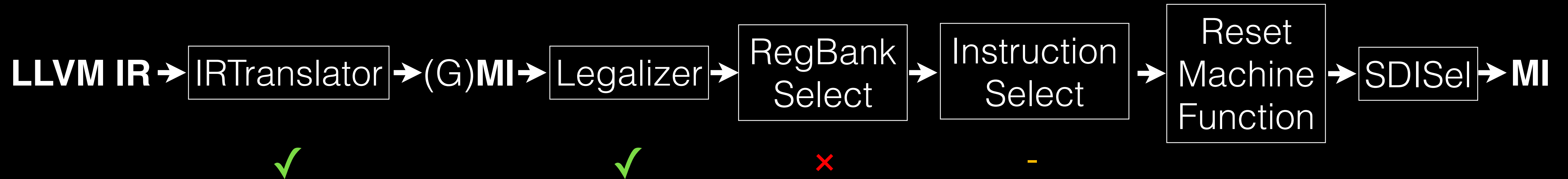
Easier Bring Up



fctI: x
... = legOp1 ...
...
... = legOpN ...

Fallback to SelectionDAG

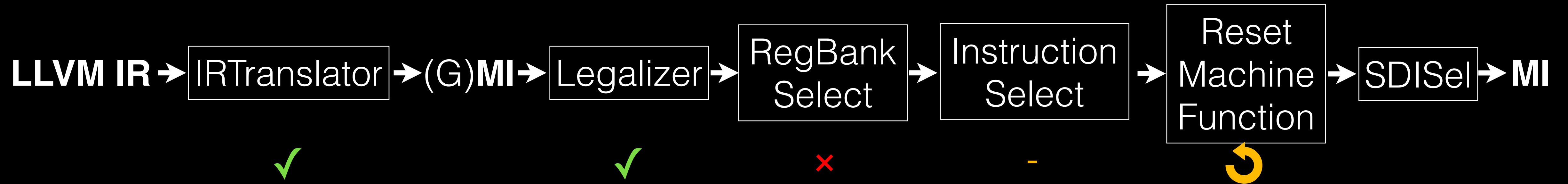
Easier Bring Up



fctI: x
... = leg0p1 ...
...
... = leg0pN ...

Fallback to SelectionDAG

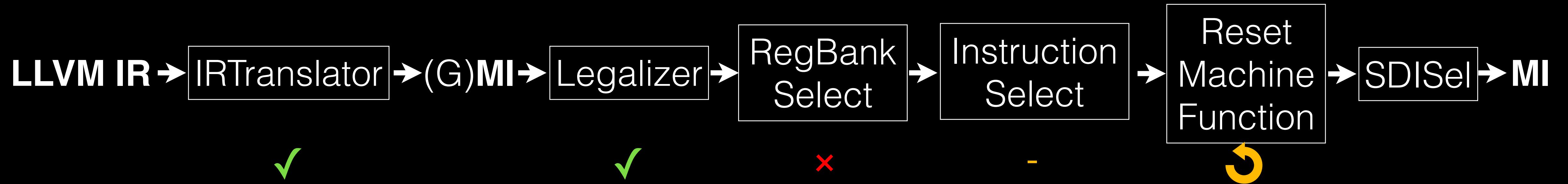
Easier Bring Up



fctI: x
... = leg0p1 ...
...
... = leg0pN ...

Fallback to SelectionDAG

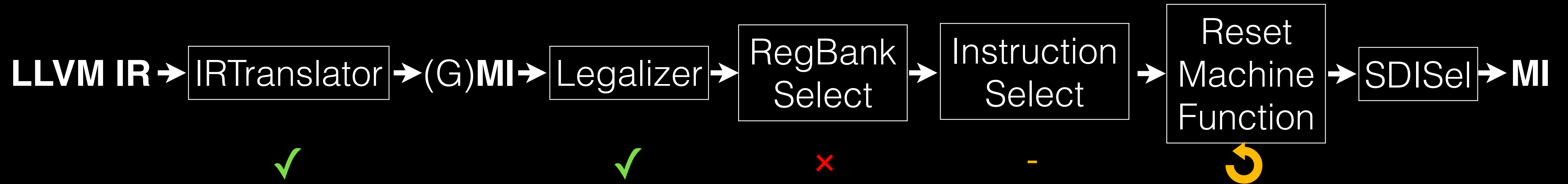
Easier Bring Up



fctI:

Fallback to SelectionDAG

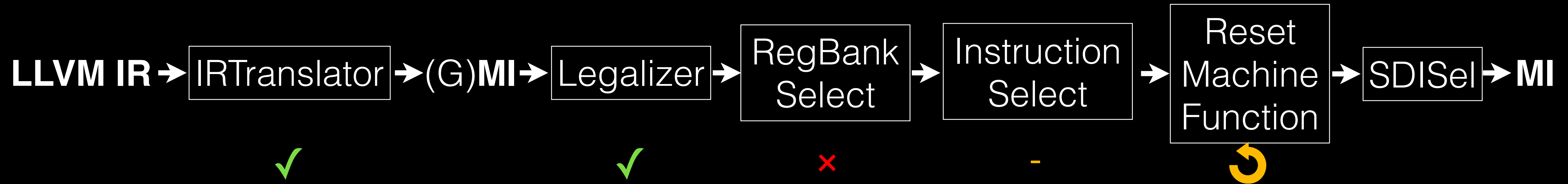
Easier Bring Up



fctI:

Fallback to SelectionDAG

Easier Bring Up



```
fctI:  
... = tgtOp1  
...  
... = tgtOpN
```





What Changed?



What Changed?



CallLowering API
More instructions supported



Legalizer

Fundamentals

```
%6( _, s128) = G_OR %4, %5
```

Legalizer

Fundamentals

```
%6.low( _, s64 ) = G_OR %4.low, %5.low  
%6.high( _, s64 ) = G_OR %4.high, %5.high
```


Legalizer

Fundamentals

```
%7(, s64) = G_OR %4.low, %5.low  
%8(, s64) = G_OR %4.high, %5.high  
%6(, s128) = G_SEQUENCE %7 %8
```

Legalizer

Fundamentals

```
%7(, s64) = G_OR %4.low, %5.low  
%8(, s64) = G_OR %4.high, %5.high  
%6(, s128) = G_SEQUENCE %7, %8
```

Legalizer

Fundamentals

```
%7(, s64) = G_OR %4.low, %5.low  
%8(, s64) = G_OR    .high,    .high  
%6(, s128) = G_SEQUENCE %7, %8
```

Legalizer

Fundamentals

```
%0( _, s64 ), %1( _, s64 ) = G_EXTRACT %4, 0, 64
%2( _, s64 ), %3( _, s64 ) = G_EXTRACT %5, 0, 64
%7( _, s64 ) = G_OR %0, %2
%8( _, s64 ) = G_OR %1, %3
%6( _, s128 ) = G_SEQUENCE %7, %8
```

Legalizer

Fundamentals

```
%0(, s64), %1(, s64) = G_EXTRACT %4, 0, 64
%2(, s64), %3(, s64) = G_EXTRACT %5, 0, 64
%7(, s64) = G_OR %0, %2
%8(, s64) = G_OR %1, %3
%6(, s128) = G_SEQUENCE %7, %8
```

Legalization Artifacts

Simpler More Regular Semantic

```
%0( _, s64 ), %1( _, s64 ) = G_EXTRACT %4, 0, 64  
%2( _, s64 ), %3( _, s64 ) = G_EXTRACT %5, 0, 64  
%7( _, s64 ) = G_OR %0, %2  
%8( _, s64 ) = G_OR %1, %3  
%6( _, s128 ) = G_SEQUENCE %7, %8
```

Legalization Artifacts

Simpler More Regular Semantic

```
%0( _, s64 ), %1( _, s64 ) = G_EXTRACT %4, 0, 64
%2( _, s64 ), %3( _, s64 ) = G_EXTRACT %5, 0, 64
%7( _, s64 ) = G_OR %0, %2
%8( _, s64 ) = G_OR %1, %3
%6( _, s128 ) = G_SEQUENCE %7, %8
```

Legalization Artifacts

Simpler More Regular Semantic

```
%0( _, s64 ), %1( _, s64 ) = G_EXTRACT %4, 0, 64  
%2( _, s64 ), %3( _, s64 ) = G_EXTRACT %5, 0, 64  
%7( _, s64 ) = G_OR %0, %2  
%8( _, s64 ) = G_OR %1, %3  
%6( _, s128 ) = G_SEQUENCE %7, %8
```


Legalization Artifacts

Simpler More Regular Semantic

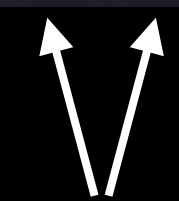
```
%0(_ , s64), %1(_ , s64) = G_EXTRACT %4, 0, 64
%2(_ , s64), %3(_ , s64) = G_EXTRACT %5, 0, 64
%7(_ , s64) = G_OR %0, %2
%8(_ , s64) = G_OR %1, %3
%6(_ , s128) = G_MERGE_VALUES %7, %8
```

Legalization Artifacts

Simpler More Regular Semantic

```
%0(_ , s64), %1(_ , s64) = G_EXTRACT %4, 0, 64  
%2(_ , s64), %3(_ , s64) = G_EXTRACT %5, 0, 64  
%7(_ , s64) = G_OR %0, %2  
%8(_ , s64) = G_OR %1, %3  
%6(_ , s128) = G_MERGE_VALUES %7, %8
```

Same type



Legalization Artifacts

Simpler More Regular Semantic

```
%0(_ , s64), %1(_ , s64) = G_EXTRACT %4, 0, 64
%2(_ , s64), %3(_ , s64) = G_EXTRACT %5, 0, 64
%7(_ , s64) = G_OR %0, %2
%8(_ , s64) = G_OR %1, %3
%6(_ , s128) = G_MERGE_VALUES %7, %8
```

Legalization Artifacts

Simpler More Regular Semantic

```
%0(_ , s64), %1(_ , s64) = G_EXTRACT %4, 0, 64
%2(_ , s64), %3(_ , s64) = G_EXTRACT %5, 0, 64
%7(_ , s64) = G_OR %0, %2
%8(_ , s64) = G_OR %1, %3
%6(_ , s128) = G_MERGE_VALUES %7, %8
```

Legalization Artifacts

Simpler More Regular Semantic

Same type



```
%0(_ , s64), %1(_ , s64) = G_UNMERGE_VALUES %4  
%2(_ , s64), %3(_ , s64) = G_EXTRACT %5, 0, 64  
%7(_ , s64) = G_OR %0, %2  
%8(_ , s64) = G_OR %1, %3  
%6(_ , s128) = G_MERGE_VALUES %7, %8
```

Legalization Artifacts

Simpler More Regular Semantic

```
%0(_ , s64) , %1(_ , s64) = G_UNMERGE_VALUES %4  
%2(_ , s64) , %3(_ , s64) = G_UNMERGE_VALUES %5  
%7(_ , s64) = G_OR %0 , %2  
%8(_ , s64) = G_OR %1 , %3  
%6(_ , s128) = G_MERGE_VALUES %7 , %8
```

Legalization Artifacts

Simpler More Regular Semantic

```
%0(_ , s64) , %1(_ , s64) = G_UNMERGE_VALUES %4  
%2(_ , s64) , %3(_ , s64) = G_UNMERGE_VALUES %5  
%7(_ , s64) = G_OR %0 , %2  
%8(_ , s64) = G_OR %1 , %3  
%6(_ , s128) = G_MERGE_VALUES %7 , %8
```

- **G_SEQUENCES** are gone

Legalization Artifacts

Simpler More Regular Semantic

```
%0(_ , s64), %1(_ , s64) = G_UNMERGE_VALUES %4  
%2(_ , s64), %3(_ , s64) = G_UNMERGE_VALUES %5  
%7(_ , s64) = G_OR %0, %2  
%8(_ , s64) = G_OR %1, %3  
%6(_ , s128) = G_MERGE_VALUES %7, %8
```

- **G_SEQUENCES** are **gone**
- **G_EXTRACTs** still exist but have only **one** result

Legalization Artifacts

Simpler More Regular Semantic

```
%0(_, s64), %1(_, s64) = G_UNMERGE_VALUES %4  
%2(_, s64), %3(_, s64) = G_UNMERGE_VALUES %5  
%7(_, s64) = G_OR %0, %2  
%8(_, s64) = G_OR %1, %3  
%6(_, s128) = G_MERGE_VALUES %7, %8
```

- **G_SEQUENCES** are **gone**
- **G_EXTRACTs** still exist but have only **one** result
- **G_[UN]MERGE_VALUES** is used instead for these uses cases

Legalization Algorithm

How Does It Work?

Legalization Algorithm

How Does It Work?

```
setAction({opcode, [OpIdx,] Type},  
          Action)
```

Legalization Algorithm

How Does It Work?

```
setAction({opcode, [OpIdx,] Type},  
          Action)
```

Legal

Legalization Algorithm

How Does It Work?

```
setAction({opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar

Legalization Algorithm

How Does It Work?

```
setAction({opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar
WidenScalar

Legalization Algorithm

How Does It Work?

```
setAction({opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%0(, s16) = G_UREM %1, %2
```


Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
setAction({G_UREM, 0, s16},  
          LegalizerInfo::WidenScalar)
```

```
%0(, s16) = G_UREM %1, %2
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

```
setAction({G_UREM, 0, s16},  
          LegalizerInfo::WidenScalar)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%5(_, s32) = G_UREM %3, %4  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%5(_, s32) = G_UREM %3, %4  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

```
setAction({G_UREM, 0, s32},  
          LegalizerInfo::Lower)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%5(_, s32) = G_UREM %3, %4  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

```
setAction({G_UREM, 0, s32},  
          LegalizerInfo::Lower)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%6(_, s32) = G_UDIV %3, %4  
%7(_, s32) = G_MUL %6, %4  
%5(_, s32) = G_SUB %3, %7  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%6(_, s32) = G_UDIV %3, %4  
%7(_, s32) = G_MUL %6, %4  
%5(_, s32) = G_SUB %3, %7  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

```
setAction({G_SUB, 0, s32},  
          LegalizerInfo::Legal)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%6(_, s32) = G_UDIV %3, %4  
%7(_, s32) = G_MUL %6, %4  
%5(_, s32) = G_SUB %3, %7  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%6(_, s32) = G_UDIV %3, %4  
%7(_, s32) = G_MUL %6, %4  
%5(_, s32) = G_SUB %3, %7  
%0(_, s16) = G_TRUNC %5
```


Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

```
setAction({G_MUL, 0, s32},  
          LegalizerInfo::Legal)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%6(_, s32) = G_UDIV %3, %4  
%7(_, s32) = G_MUL %6, %4  
%5(_, s32) = G_SUB %3, %7  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%6(_, s32) = G_UDIV %3, %4  
%7(_, s32) = G_MUL %6, %4  
%5(_, s32) = G_SUB %3, %7  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

```
setAction({G_UDIV, 0, s32},  
          LegalizerInfo::LibCall)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%6(_, s32) = G_UDIV %3, %4  
%7(_, s32) = G_MUL %6, %4  
%5(_, s32) = G_SUB %3, %7  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

```
setAction({G_UDIV, 0, s32},  
          LegalizerInfo::LibCall)
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

Lower
Libcall
Custom
Unsupported
NotFound

```
%3(_, s32) = G_ZEXT %1  
%4(_, s32) = G_ZEXT %2  
%r0 = COPY %3  
%r1 = COPY %4  
%r0 = BLX uidivmod %r0, %r1  
%6(_, s32) = COPY %r0  
%7(_, s32) = G_MUL %6, %4  
%5(_, s32) = G_SUB %3, %7  
%0(_, s16) = G_TRUNC %5
```

Legalization Algorithm

How Does It Work?

```
setAction({Opcode, [OpIdx,] Type},  
          Action)
```

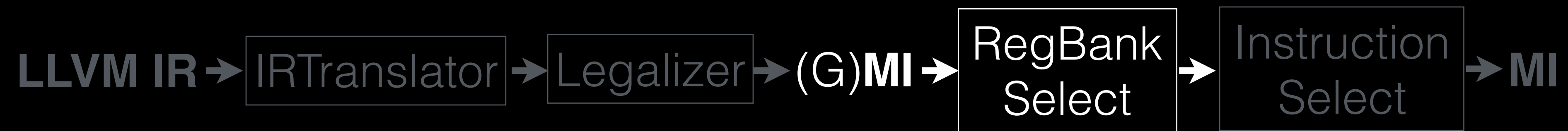
```
%0(, s16) = G_UREM %1, %2
```

Legal
NarrowScalar
WidenScalar
FewerElements
MoreElements

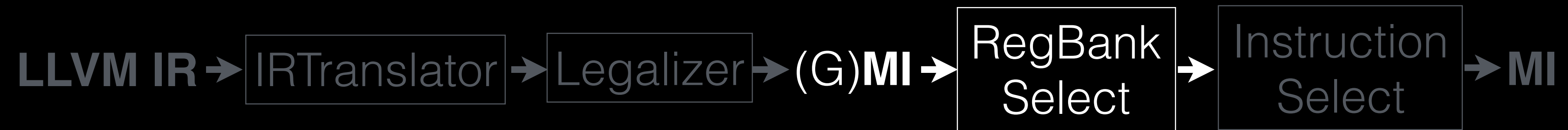
Lower
Libcall
Custom
Unsupported
NotFound

```
%3(, s32) = G_ZEXT %1  
%4(, s32) = G_ZEXT %2  
%r0 = COPY %3  
%r1 = COPY %4  
%r0 = BLX uidivmod %r0, %r1  
%6(, s32) = COPY %r0  
%7(, s32) = G_MUL %6, %4  
%5(, s32) = G_SUB %3, %7  
%0(, s16) = G_TRUNC %5
```

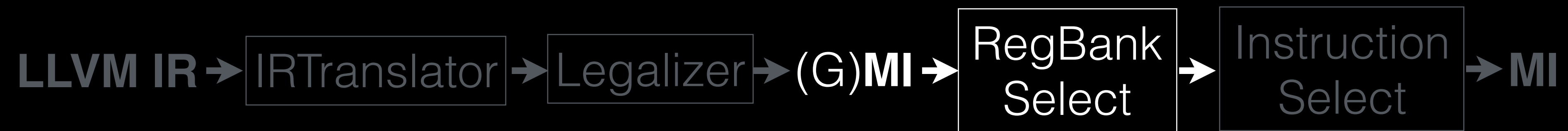




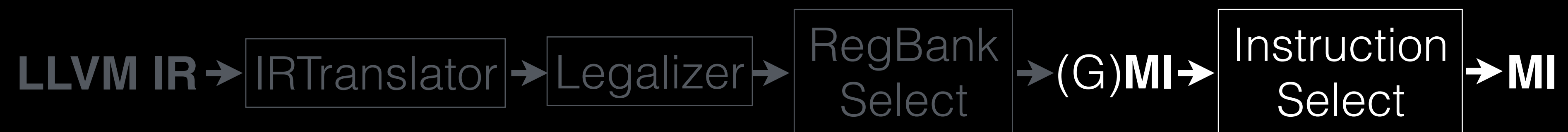
What Changed?



What Changed?



Partial TableGen support
Statically allocated structures



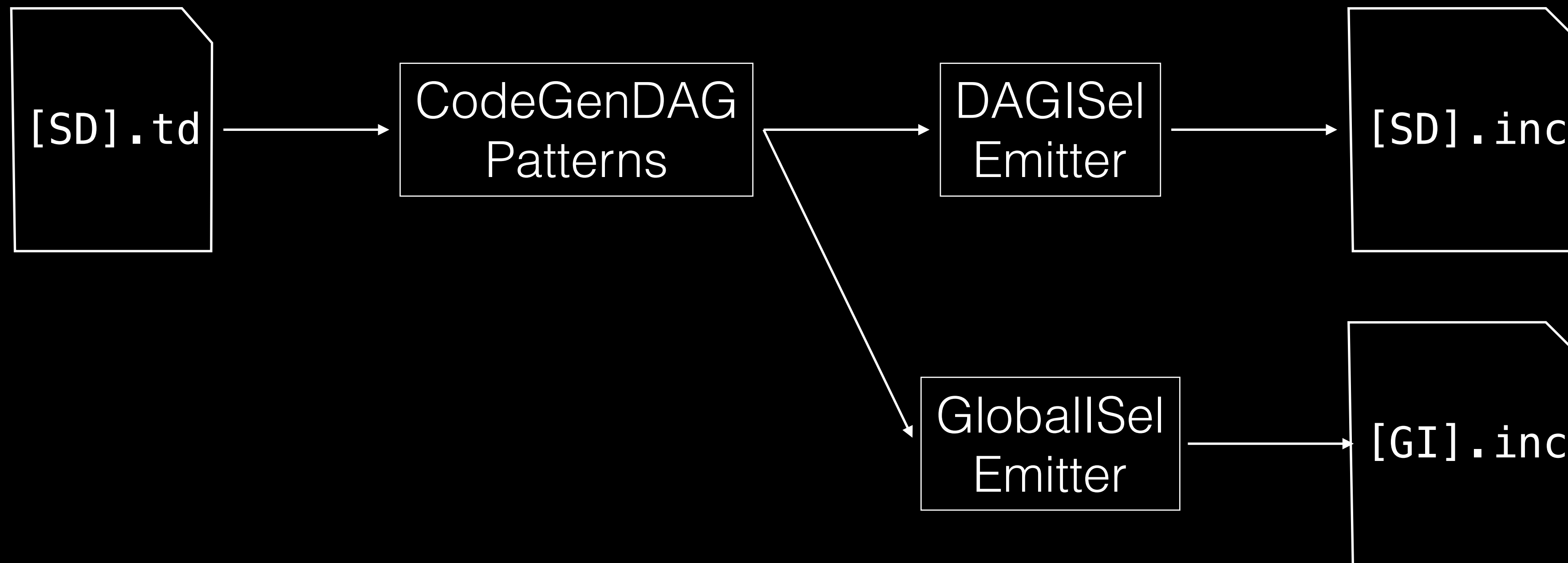
Instruction Select

TableGen Support



Instruction Select

TableGen Support



Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

Instruction Select

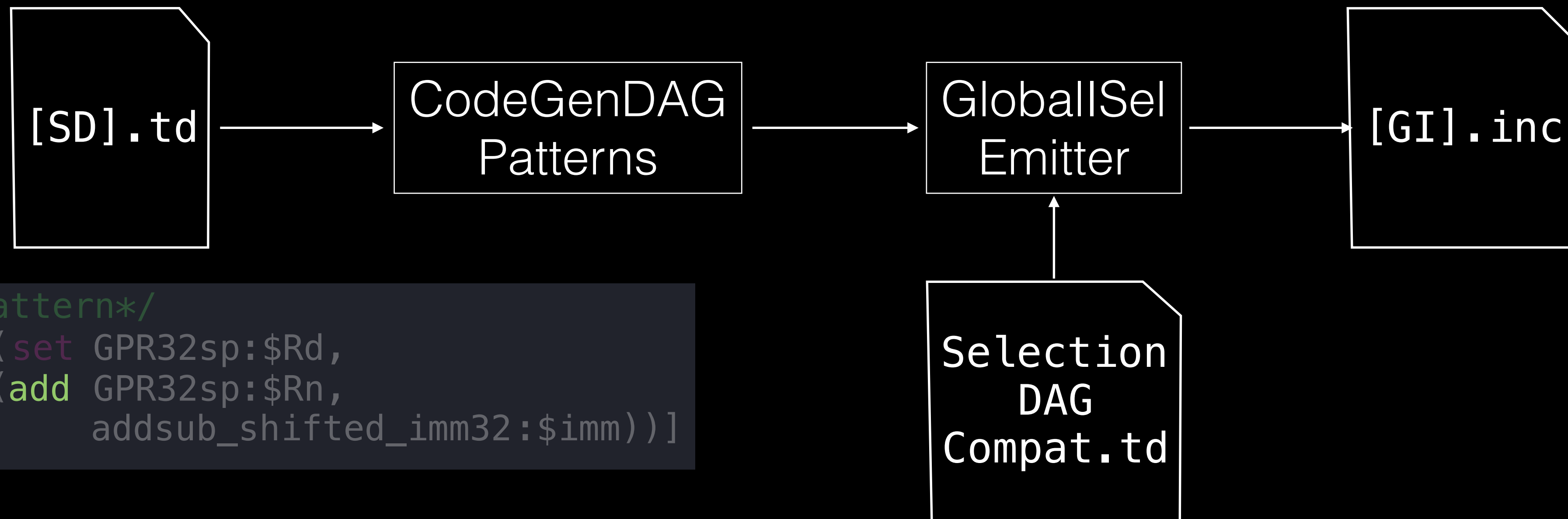
TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

Instruction Select

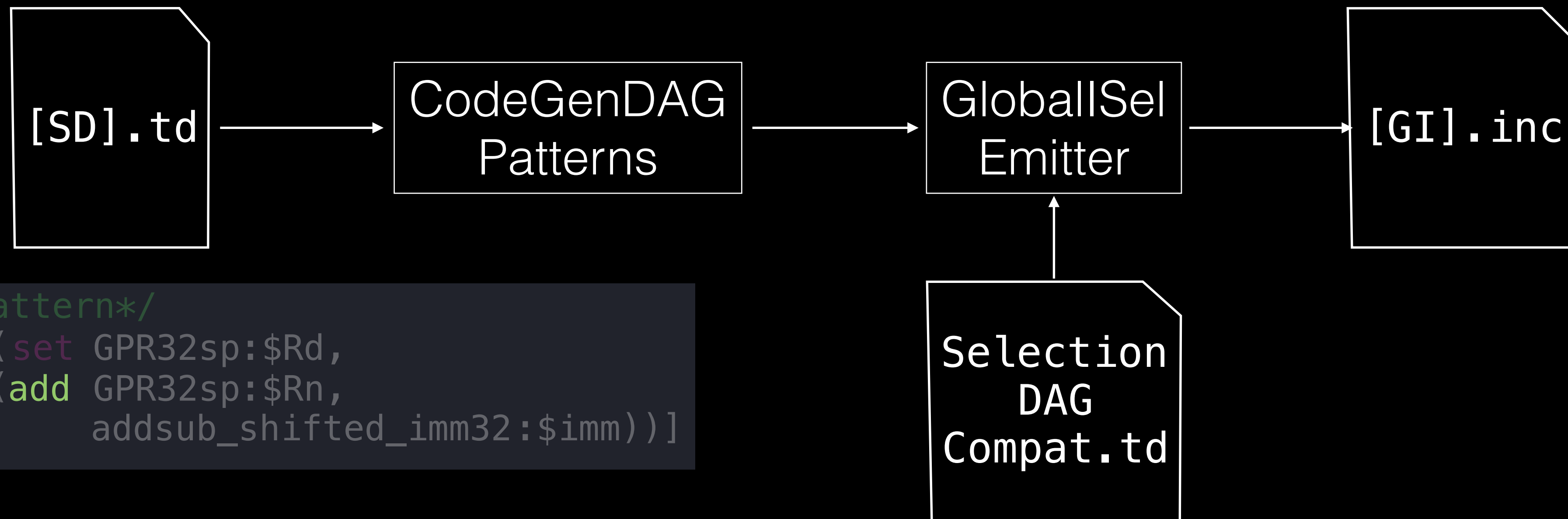
TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

Instruction Select

TableGen Support

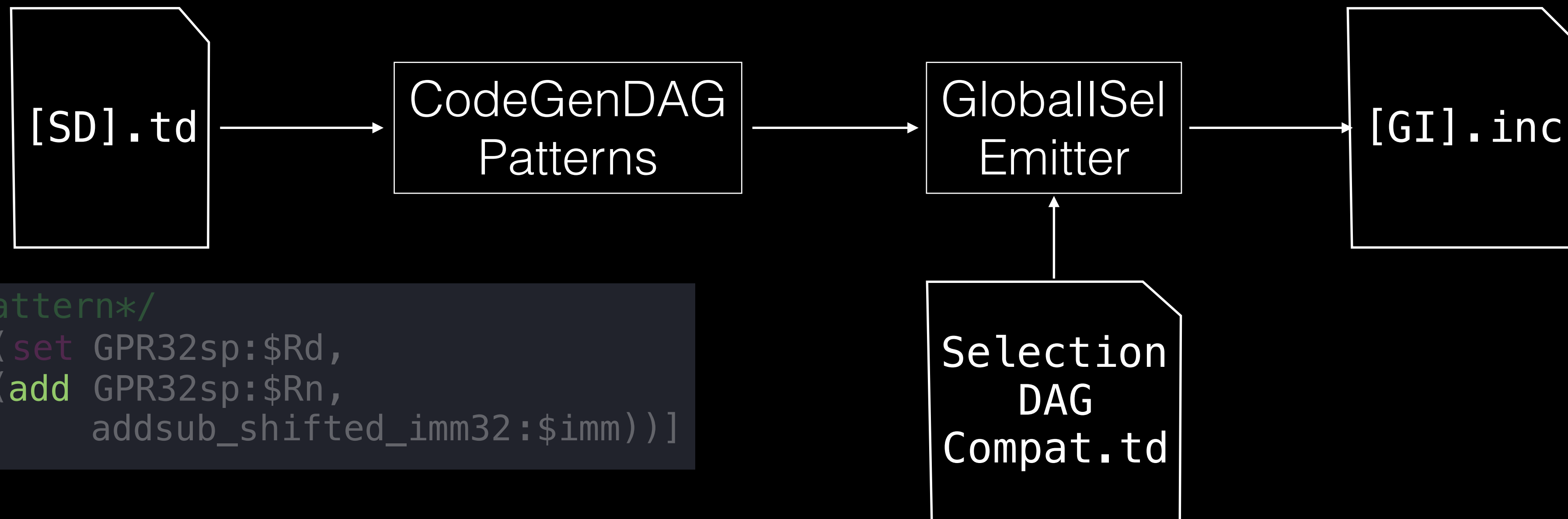


```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
def : GINodeEquiv<G_ADD, add>;
```


Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
def : GINodeEquiv<G_ADD, add>;
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
               addsub_shifted_imm32:$imm))]
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
               addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

Instruction Select

TableGen Support

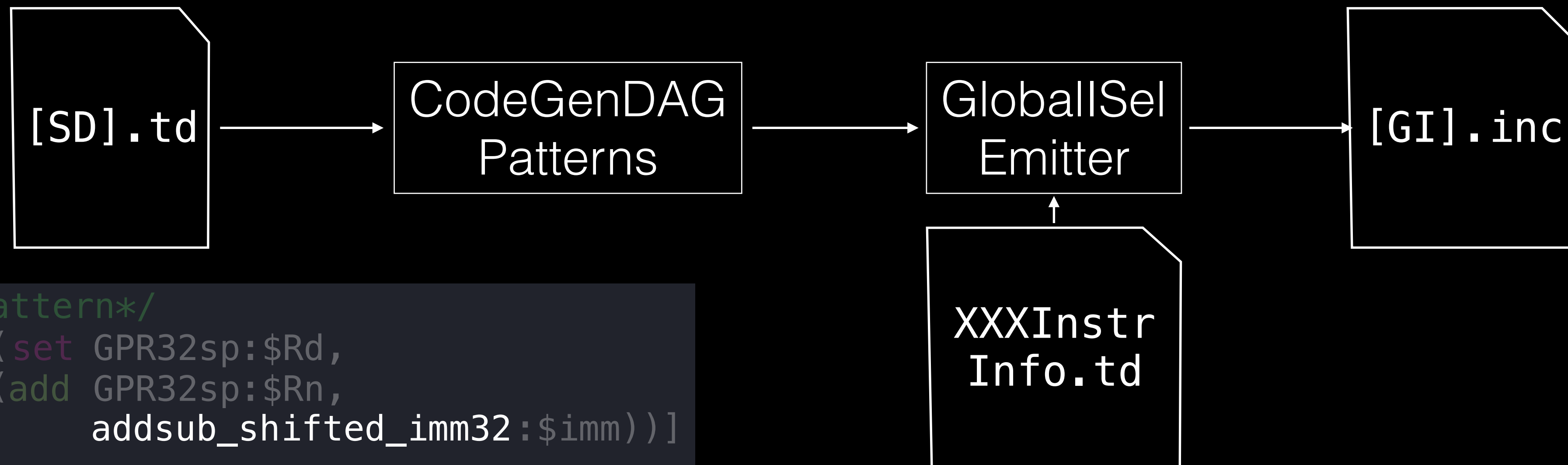


```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
            (add GPR32sp:$Rn,  
              addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

Instruction Select

TableGen Support



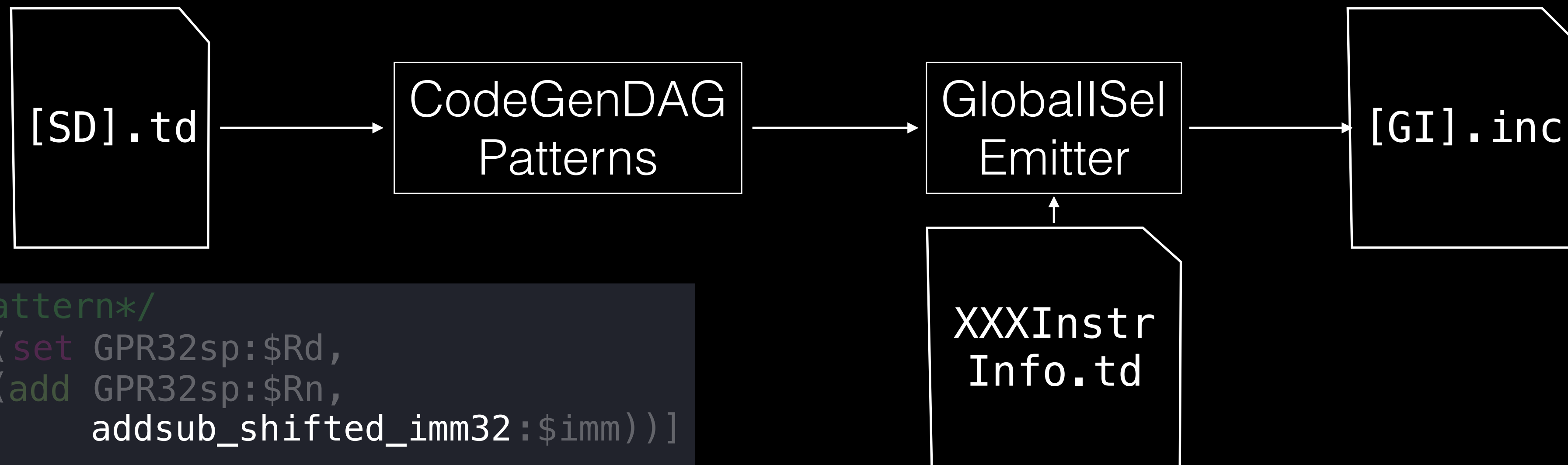
```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

```
def gi_addsub_shifted_imm32:
```

Instruction Select

TableGen Support



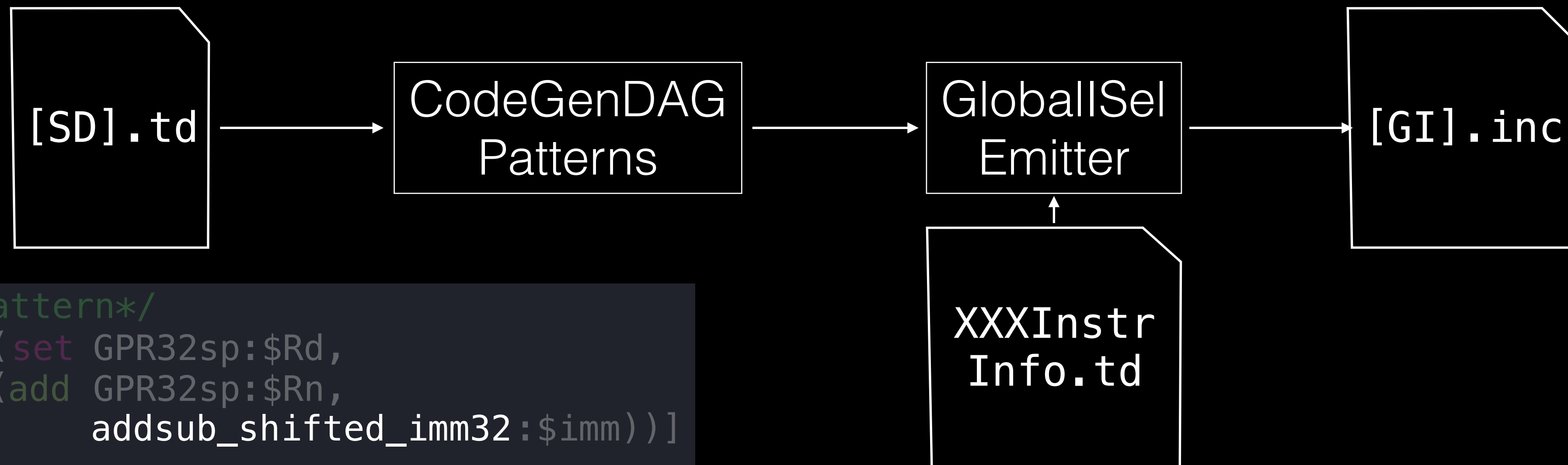
```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
               addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

```
def gi_addsub_shifted_imm32:  
  GIComplexOperandMatcher  
    <s32, "selectArithImmed">,
```

Instruction Select

TableGen Support



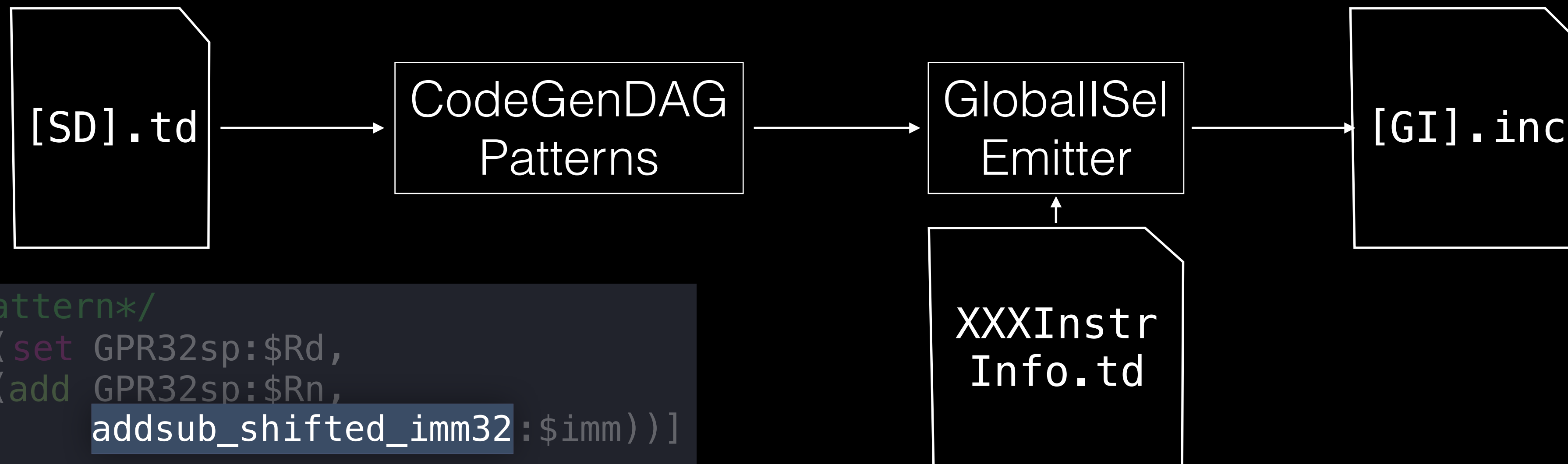
```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
            (add GPR32sp:$Rn,  
              addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

```
def gi_addsub_shifted_imm32:  
  GIComplexOperandMatcher  
    <s32, "selectArithImmed">,  
  GIComplexPatternEquiv  
    <addsub_shifted_imm32>;
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

```
def gi_addsub_shifted_imm32:  
  GIComplexOperandMatcher  
    <s32, "selectArithImmed">,  
  GIComplexPatternEquiv  
    <addsub_shifted_imm32>;
```


Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
               addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

```
bool AArch64DAGToDAGISel::  
  SelectArithImmed(  
    SDValue N, [...])
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
                  "SelectArithImmed",  
                  [imm]>
```

```
bool AArch64DAGToDAGISel::  
  SelectArithImmed(  
    SDValue N, [...])
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
               addsub_shifted_imm32:$imm))]
```

```
class addsub_shifted_imm<i32>  
  : Operand<i32>,  
    ComplexPattern<i32, 2,  
      "SelectArithImmed",  
      [imm]>
```

```
bool AArch64DAGToDAGISel::  
  SelectArithImmed(  
    SDValue N, [...])
```

```
InstructionSelector::ComplexRendererFn  
AArch64InstructionSelector::  
  selectArithImmed(  
    MachineOperand &Root) const
```

Instruction Select

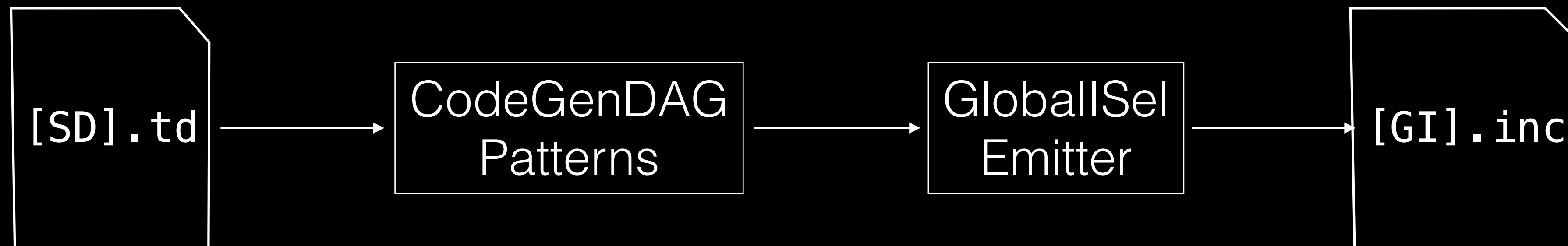
TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
    /*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
    /*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
    /*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```


Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
    /*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
    /*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
/*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```

Instruction Select

TableGen Support

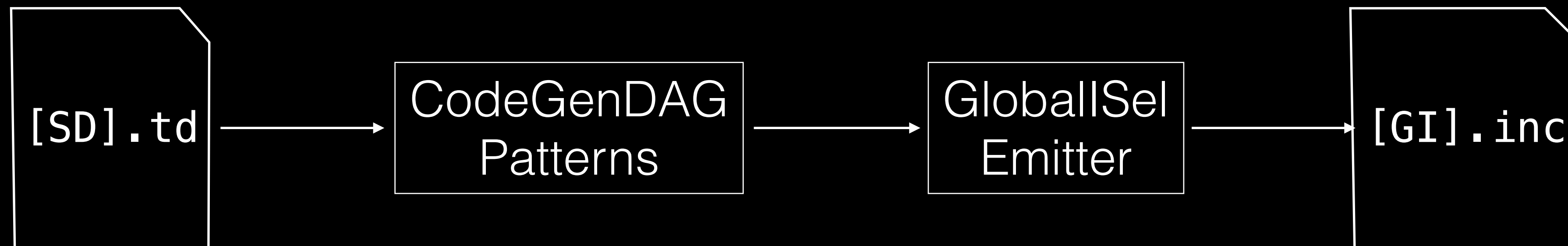


```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
/*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```

Instruction Select

TableGen Support



```
/*ADDWri Pattern*/  
ADDWri : [(set GPR32sp:$Rd,  
             (add GPR32sp:$Rn,  
                 addsub_shifted_imm32:$imm))]
```

```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
    /*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```

Instruction Select

TableGen Support



```
GIM_Try, /*On fail goto*//*Label 478*/ 30417,  
GIM_CheckNumOperands, /*MI*/0, /*Expected*/3,  
GIM_CheckOpcode, /*MI*/0, TargetOpcode::G_ADD,  
GIM_CheckRegBankForClass, [...]GPR32spRegClassID,  
[...]  
GIM_CheckComplexPattern, /*MI*/0, /*Op*/1,  
    /*Renderer*/0, GICP_gi_addsub_shifted_imm32,  
[...]  
GIR_BuildMI, [...] /*Opcode*/AArch64::ADDWri,  
[...]  
GIR_ComplexRenderer, [...]  
GIR_EraseFromParent, /*InsnID*/0,  
[...]  
GIR_Done
```

Present

Present
Status

Present

Status

- Non-optional library

Present

Status

- Non-optional library
- Ports in progress:

Present

Status

- Non-optional library
- Ports in progress:

X86 AMDGPU AArch64 ARM Hexagon PowerPC
Mips SystemZ WebAssembly AVR NVPTX Sparc
BPF XCore Lanai RISCV MSP430 Nios2 ARC

Present

Status

- Non-optional library
- Ports in progress:

X86 AMDGPU AArch64 ARM Hexagon PowerPC
Mips SystemZ WebAssembly AVR NVPTX Sparc
BPF XCore Lanai RISCV MSP430 Nios2 ARC

Present

Status

- Non-optional library
- Ports in progress:

X86 AMDGPU **AArch64** ARM Hexagon PowerPC
Mips SystemZ WebAssembly AVR NVPTX Sparc
BPF XCore Lanai RISCV MSP430 Nios2 ARC

Present

Status: AArch64 -O0

Present

Status: AArch64 -O0

- Test Suite + SPEC

Present

Status: AArch64 -O0

- Test Suite + SPEC



■ % Instructions handled without fallback

Present

Status: AArch64 -O0

- Test Suite + SPEC
- Clang self-host



■ % Instructions handled without fallback

Present

Status: AArch64 -O0

- Test Suite + SPEC
- Clang self-host



■ % Instructions handled without fallback

Present

Status: AArch64 -O0

- Test Suite + SPEC
- Clang self-host
- Our internal software

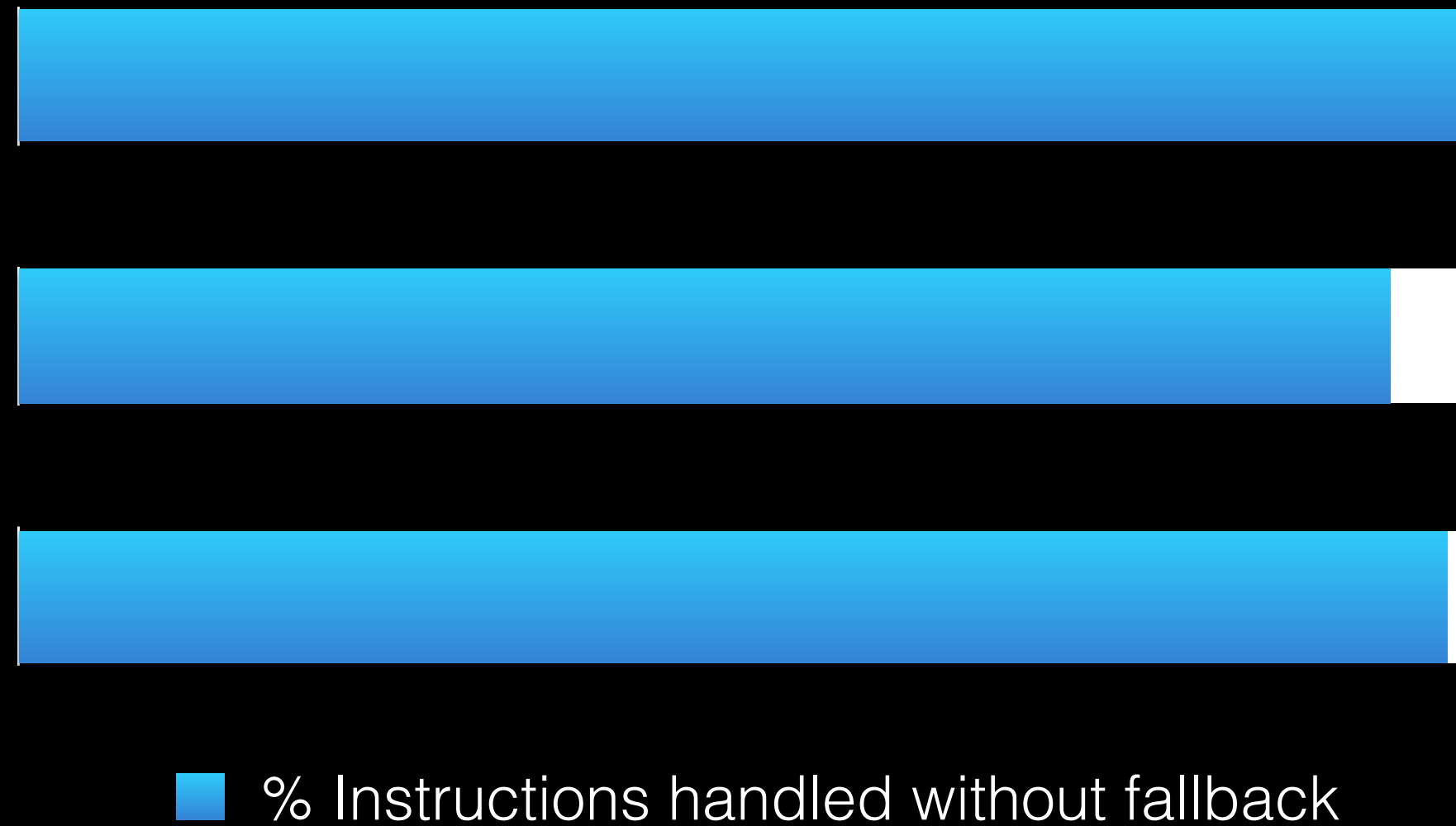


■ % Instructions handled without fallback

Present

Status: AArch64 -O0

- Test Suite + SPEC
- Clang self-host
- Our internal software



Present

Status: Select TableGen

Present

Status: Select TableGen

- AArch64

Present

Status: Select TableGen

- AArch64



■ % Imported patterns

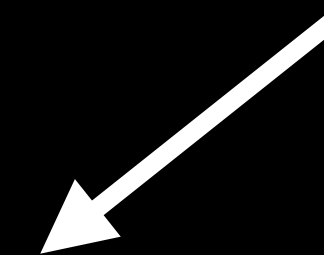
Present

Status: Select TableGen

- AArch64



Lots of vectors

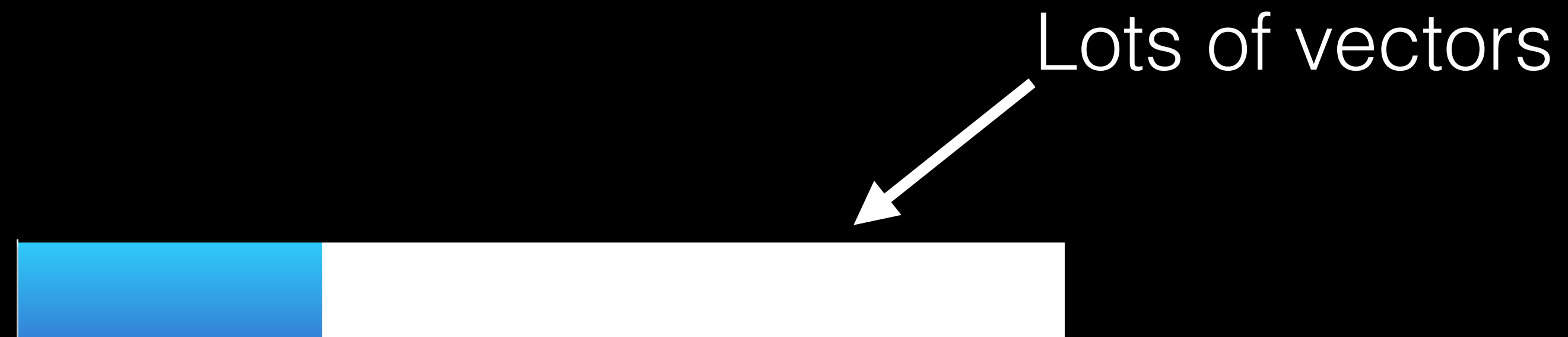


■ % Imported patterns

Present

Status: Select TableGen

- AArch64
- ARM



■ % Imported patterns

Present

Status: Select TableGen

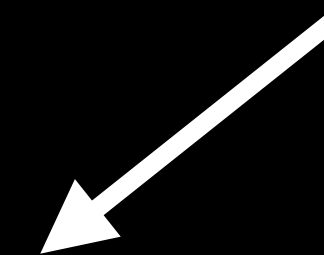
- AArch64



- ARM



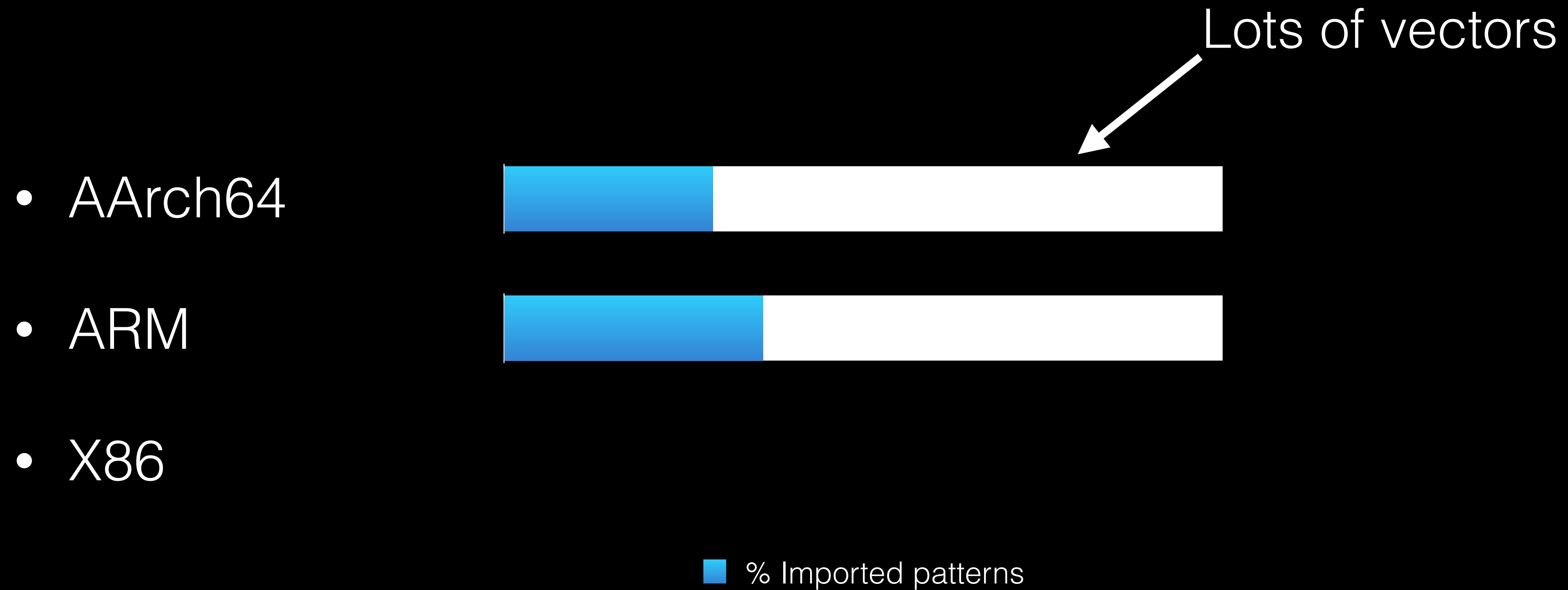
Lots of vectors



■ % Imported patterns

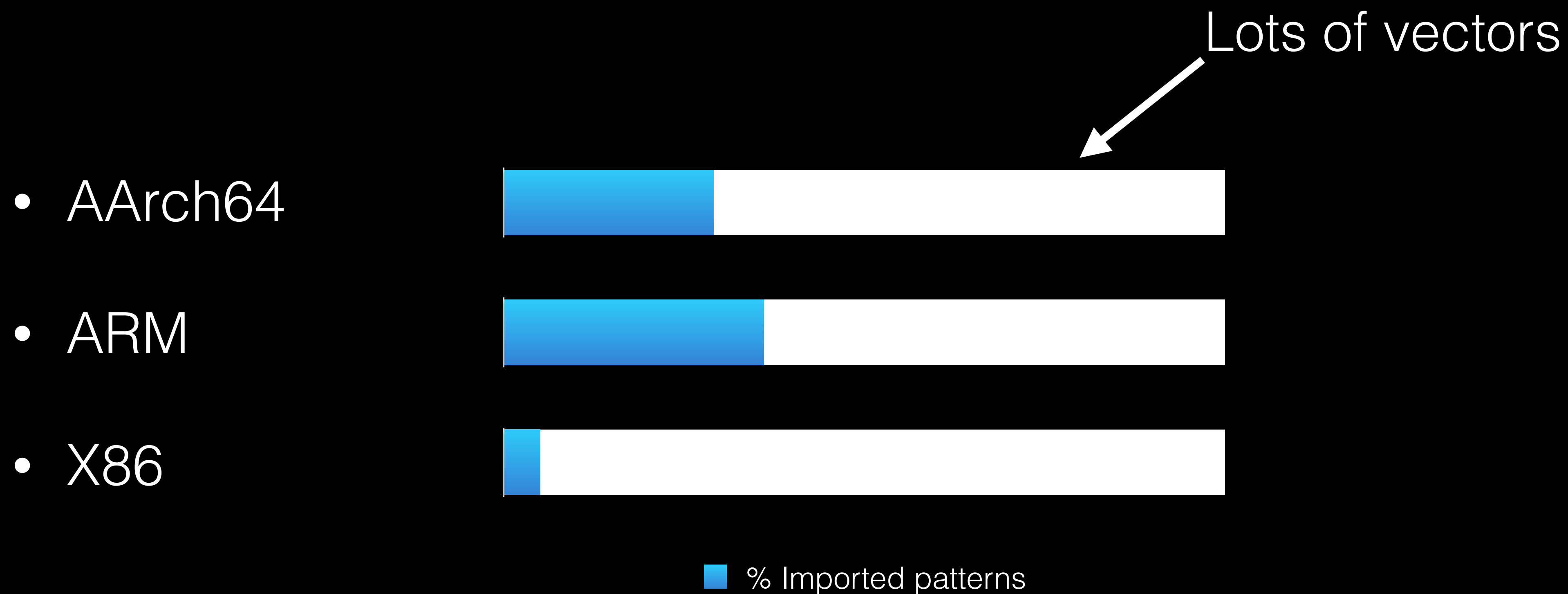
Present

Status: Select TableGen



Present

Status: Select TableGen



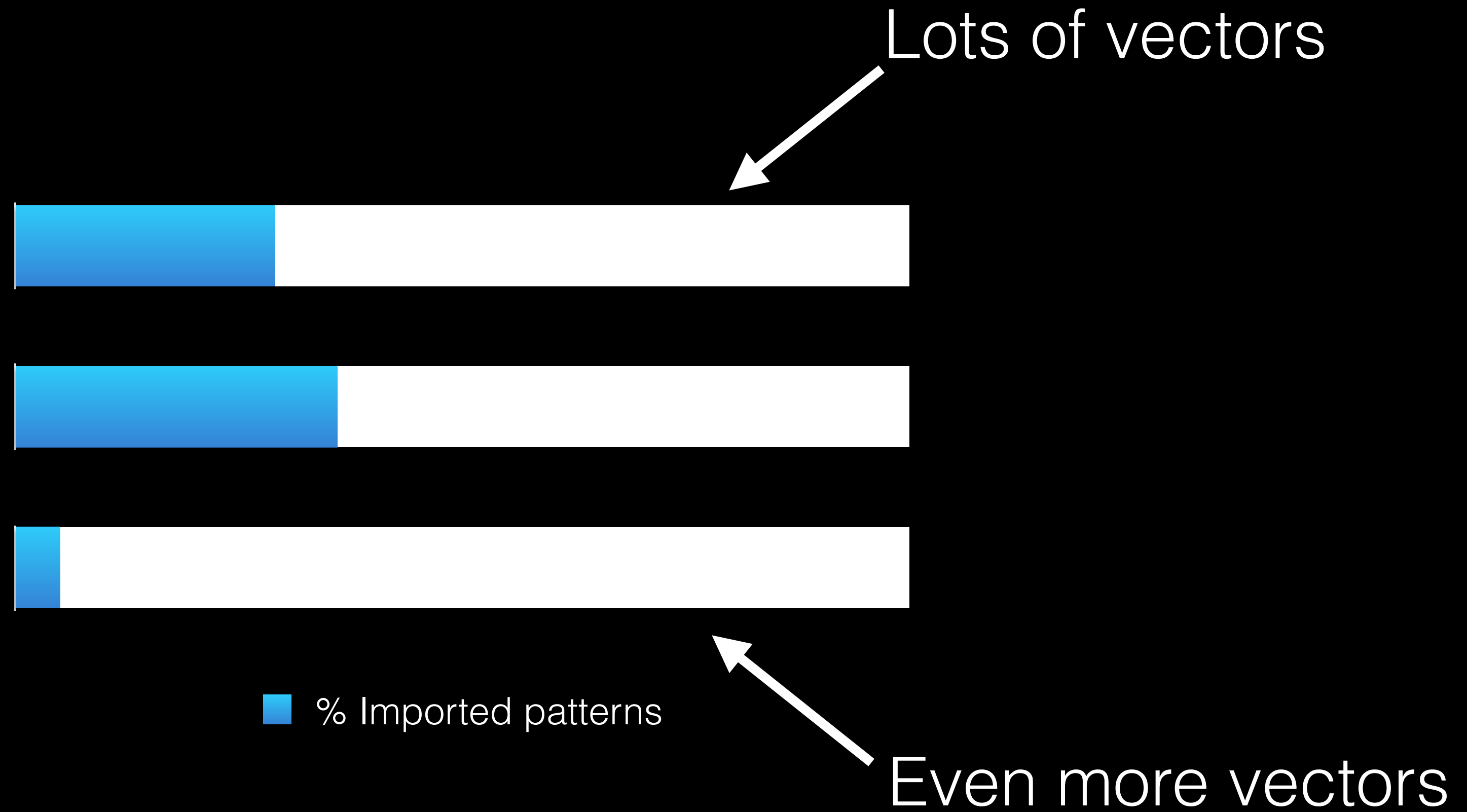
Present

Status: Select TableGen

- AArch64

- ARM

- X86



Present

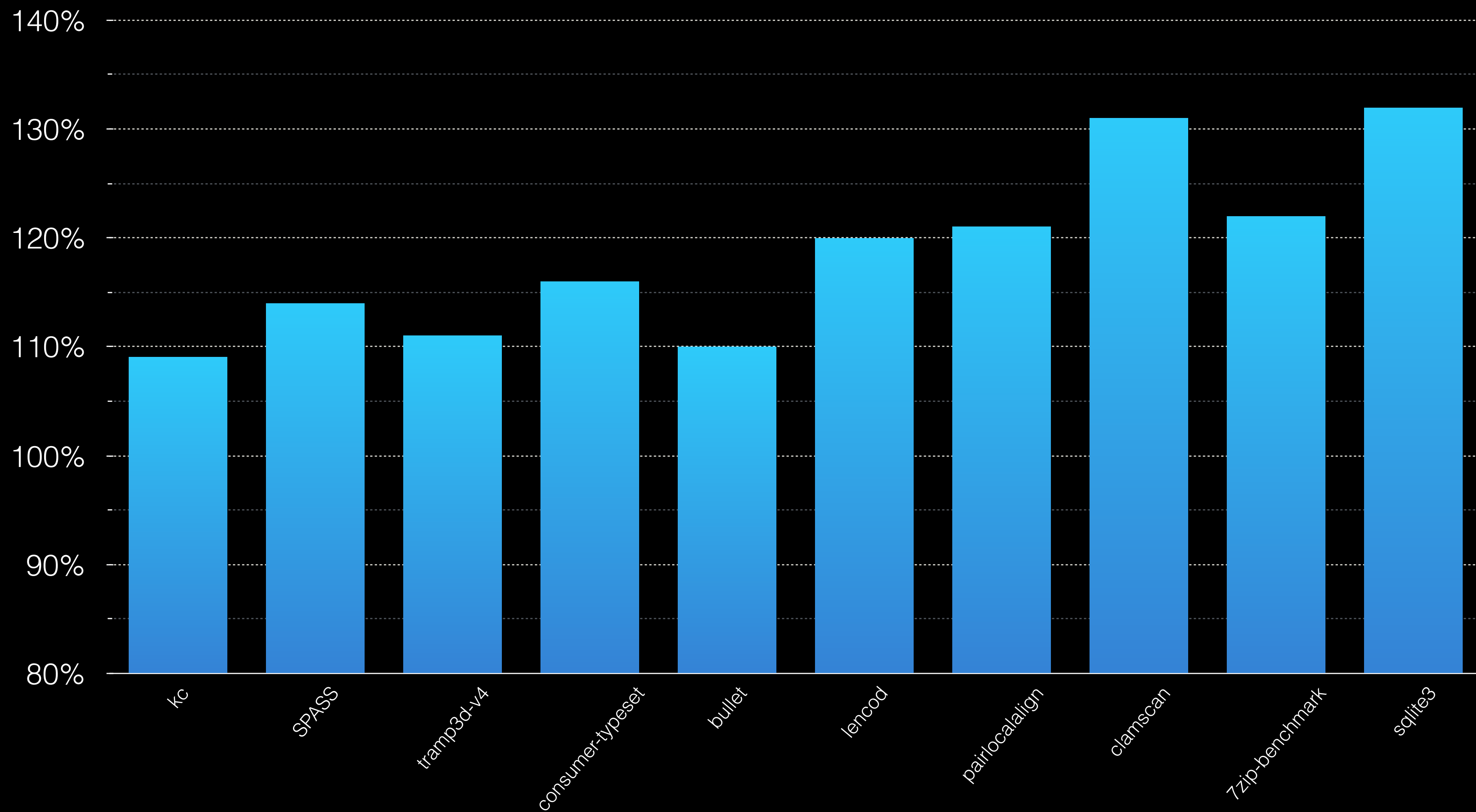
Performance

Performance

Code size

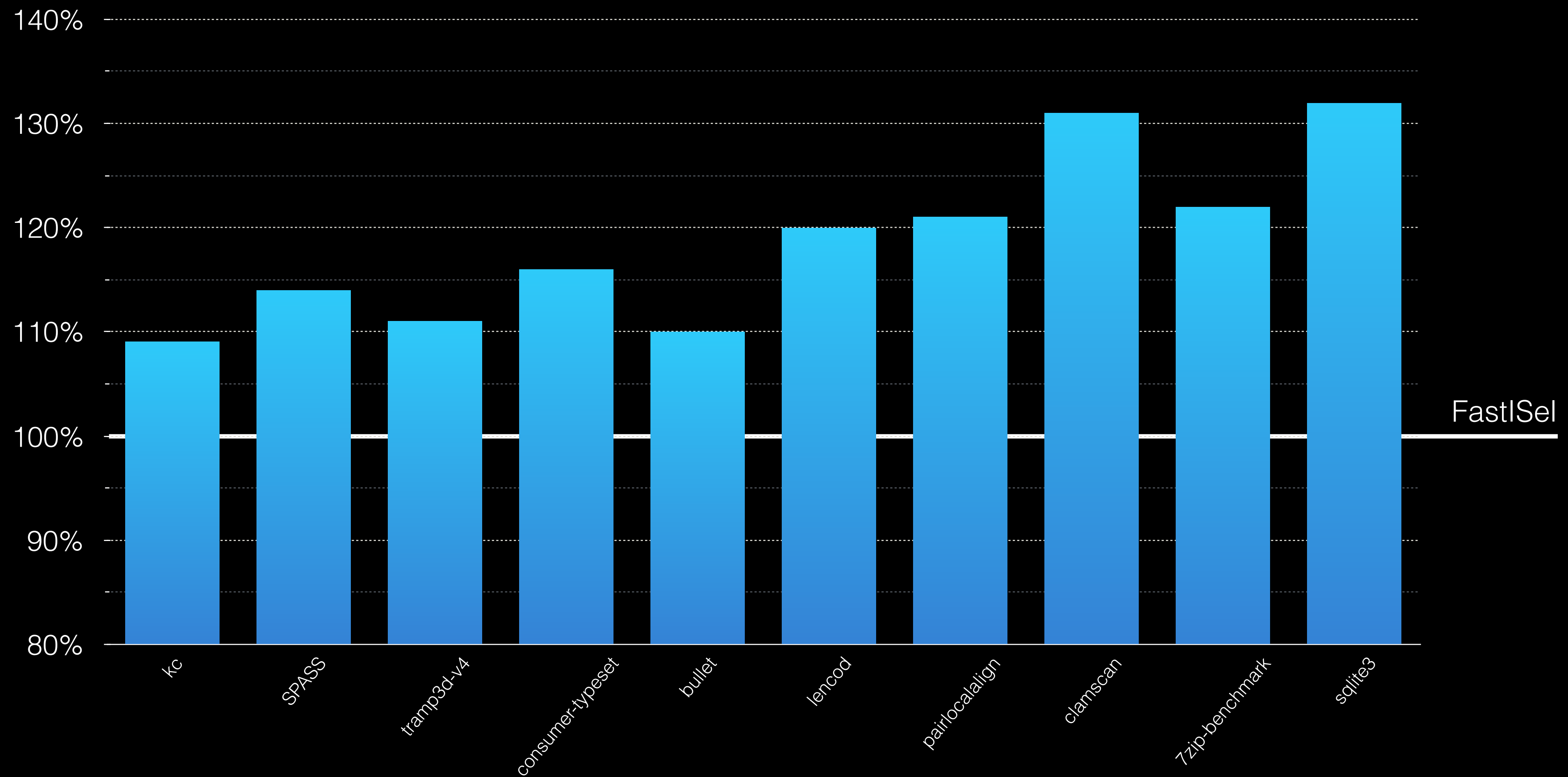
Performance

Code size



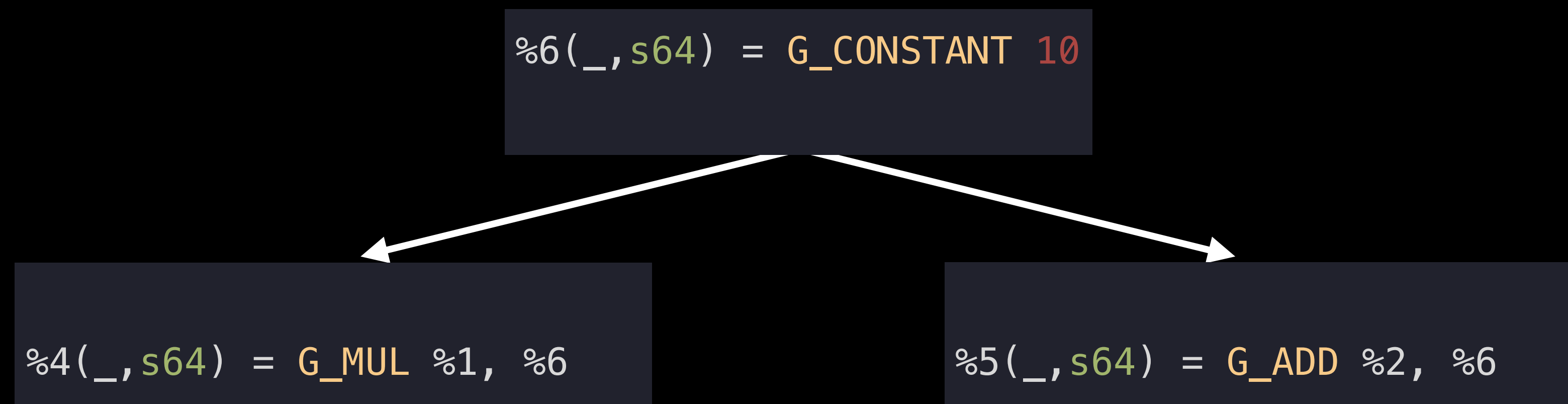
Performance

Code size



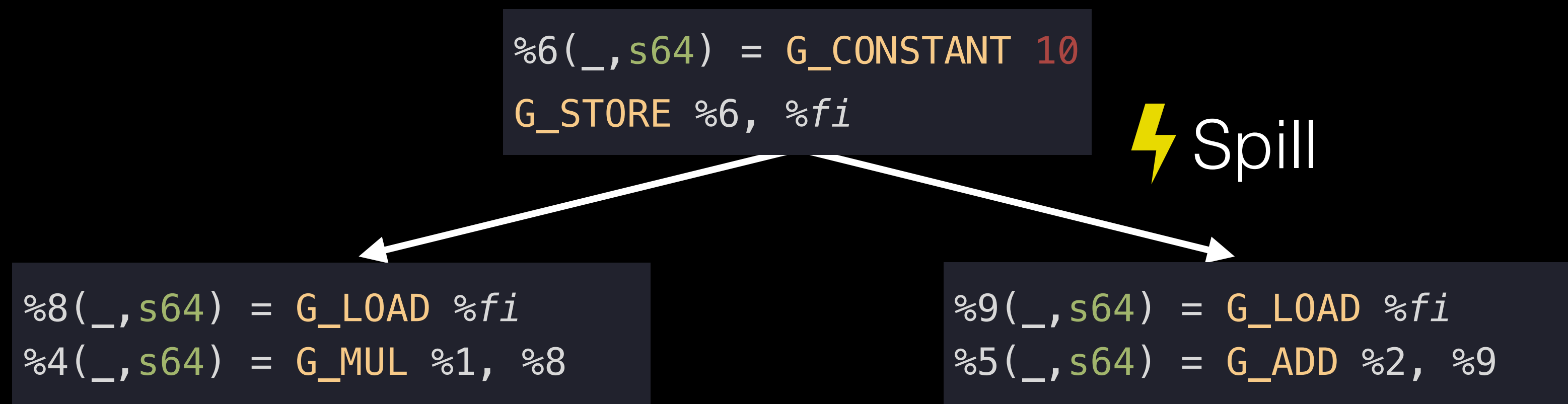
Performance

Constant placement



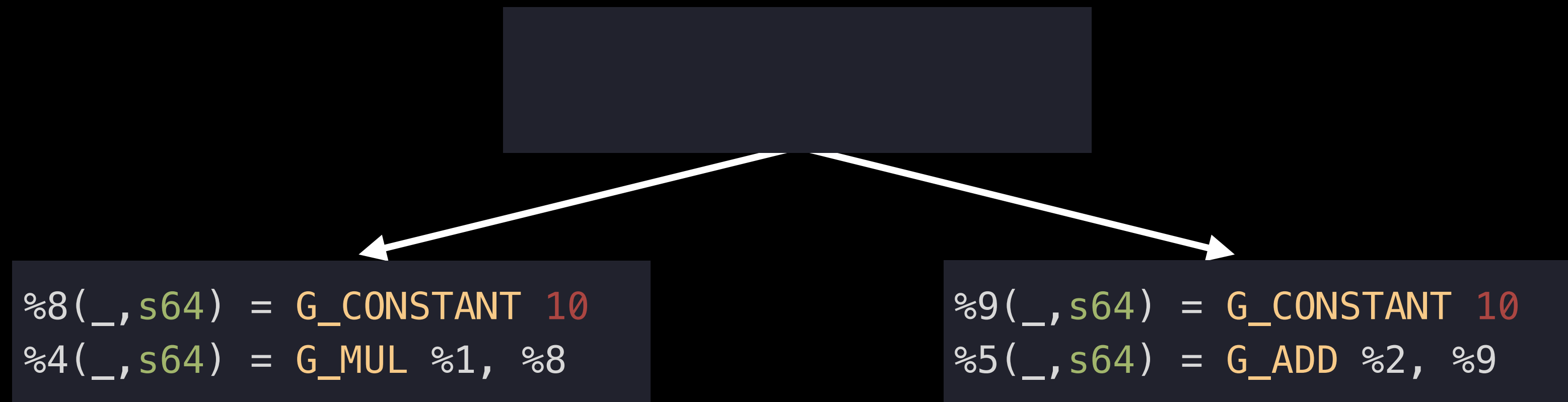
Performance

Constant placement: -regalloc=fast



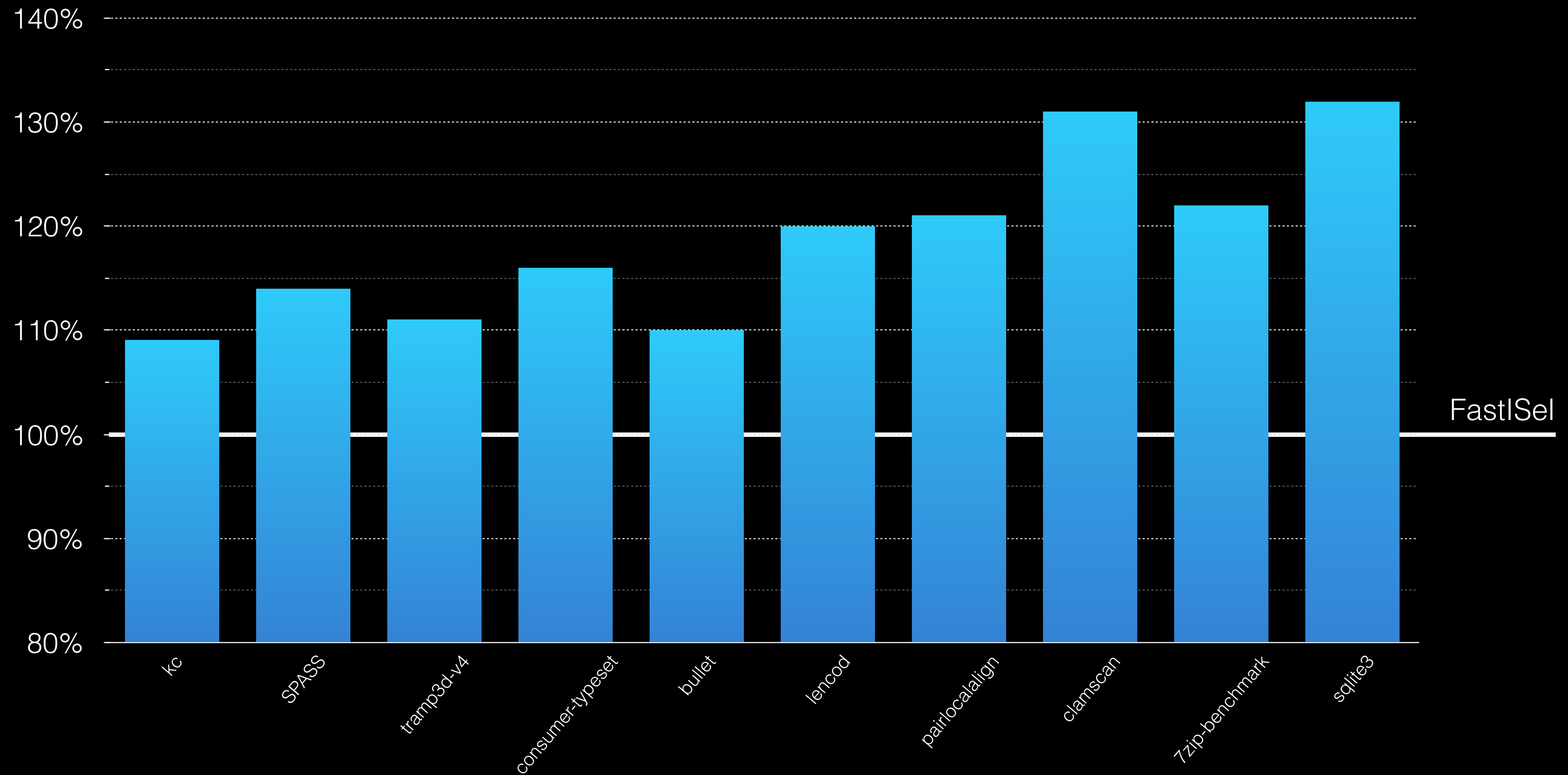
Performance

Constant placement: Localizer



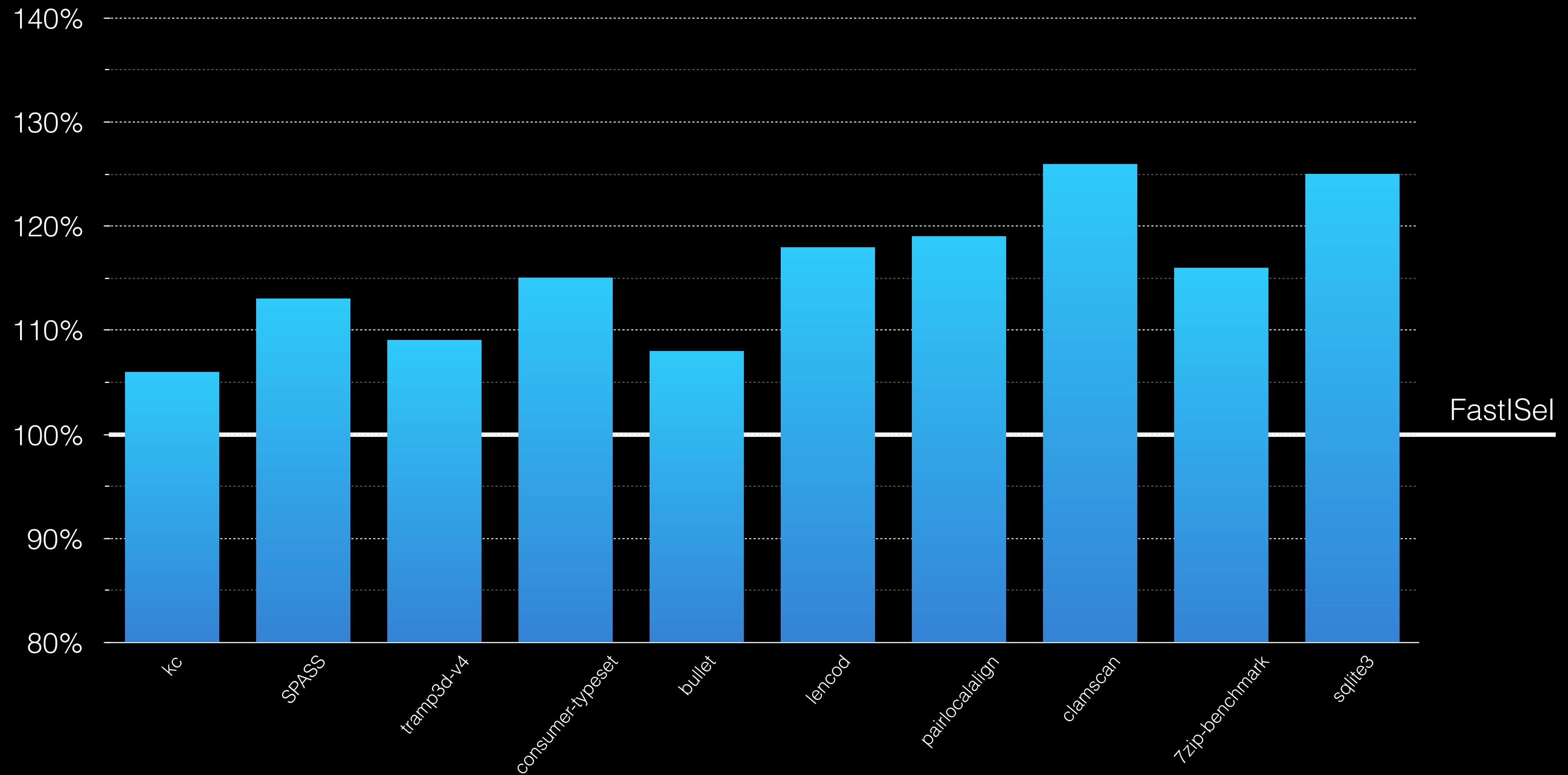
Performance

Code size: before



Performance

Code size: with localizer

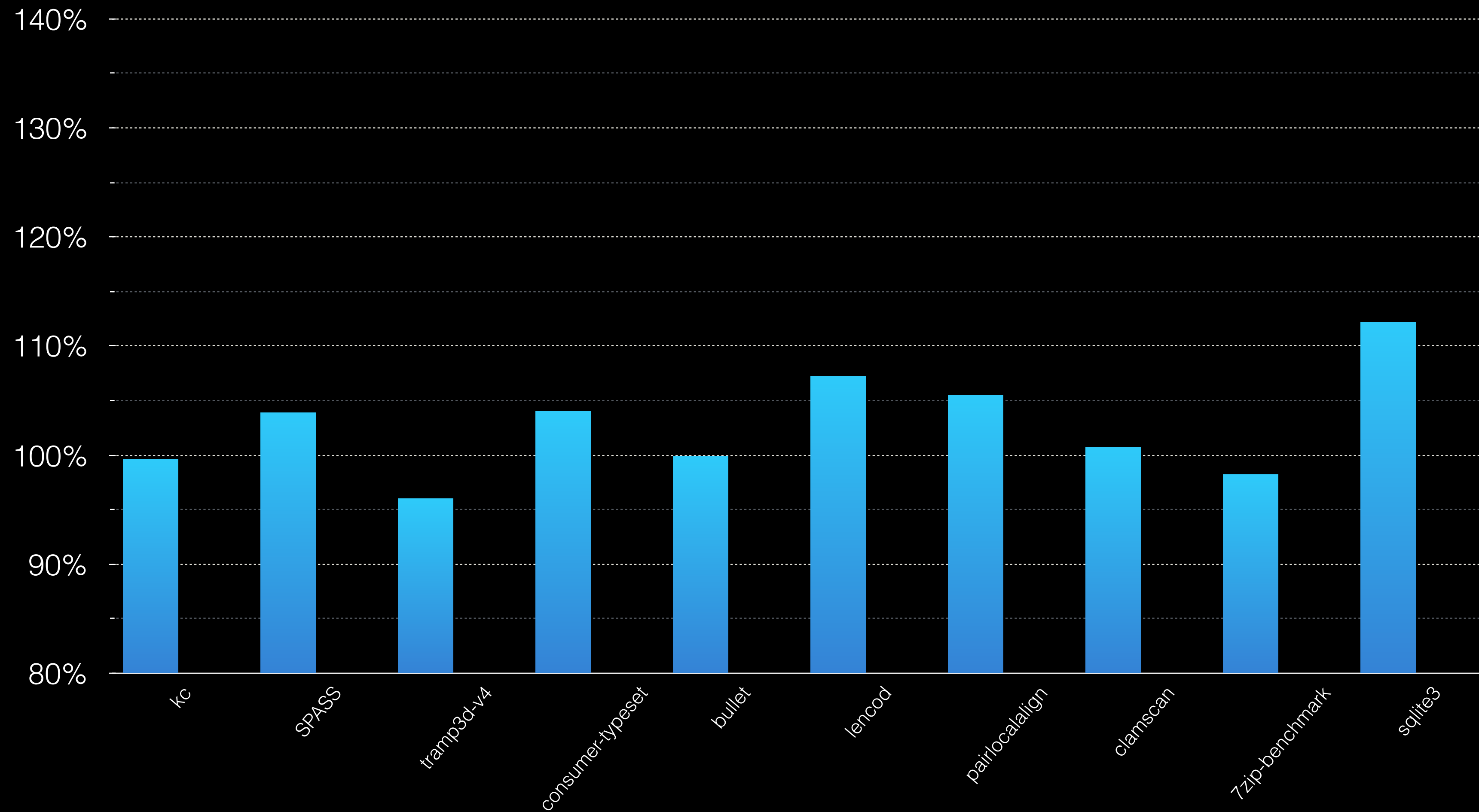


Performance

Compile time

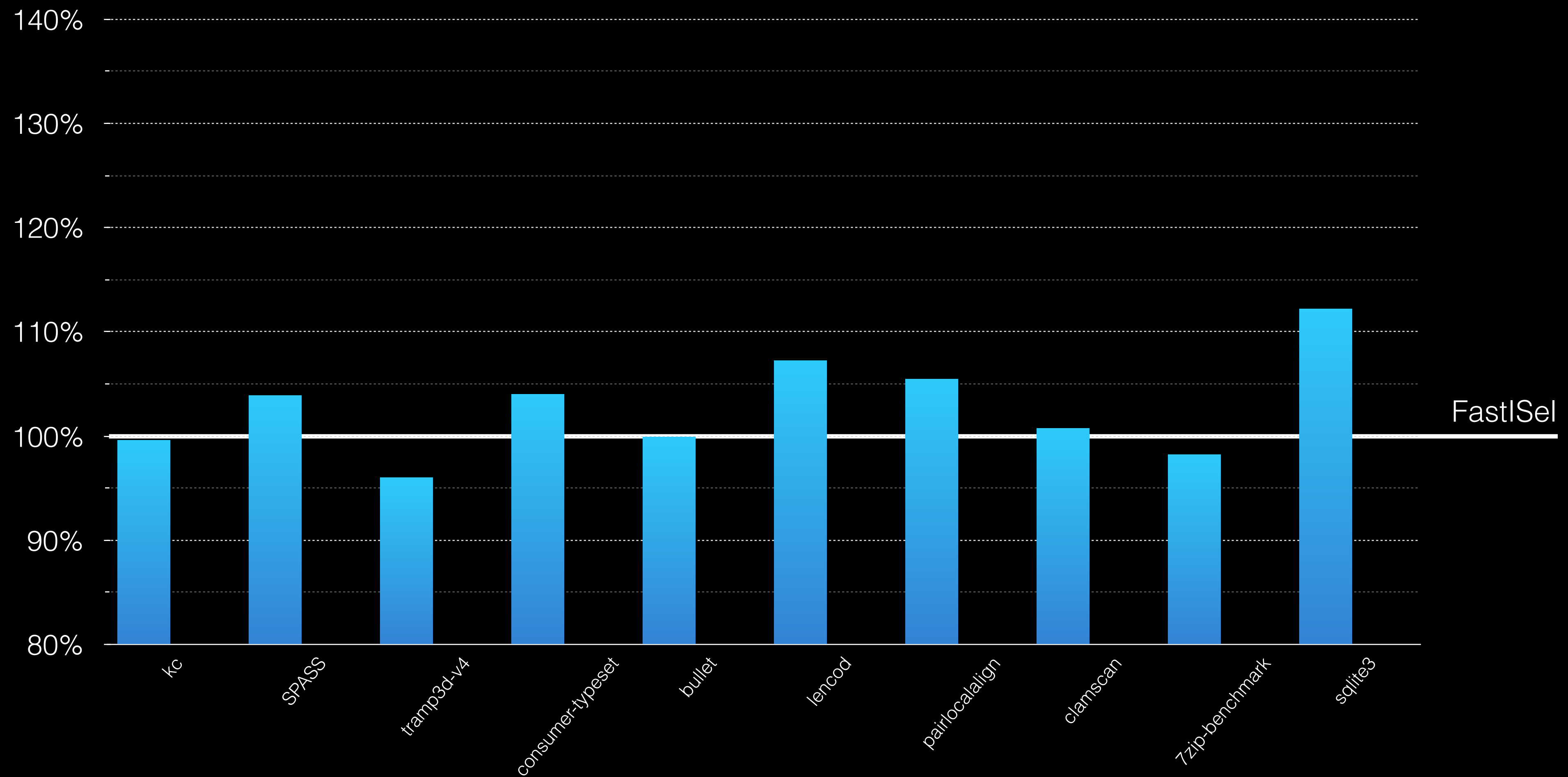
Performance

Compile time



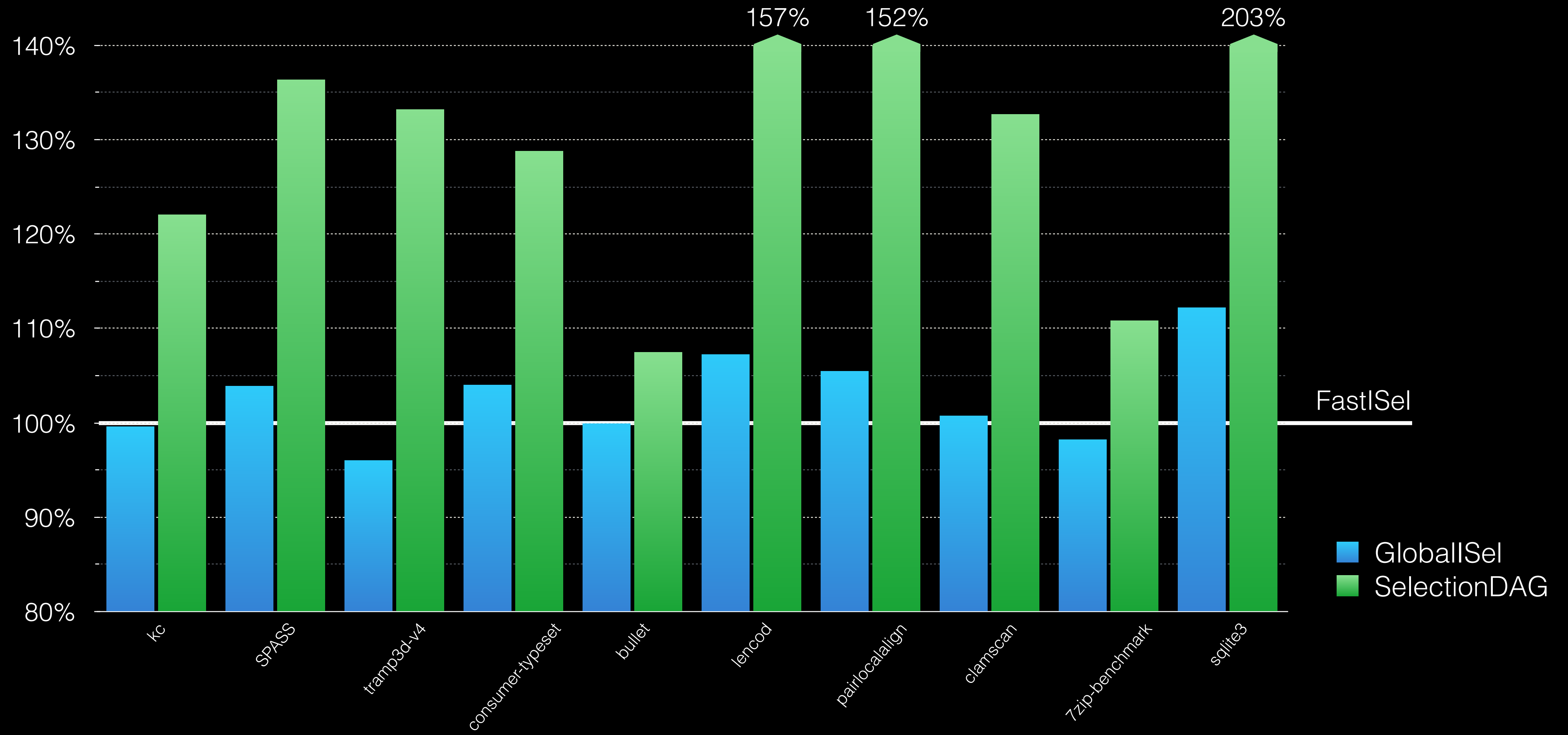
Performance

Compile time



Performance

Compile time

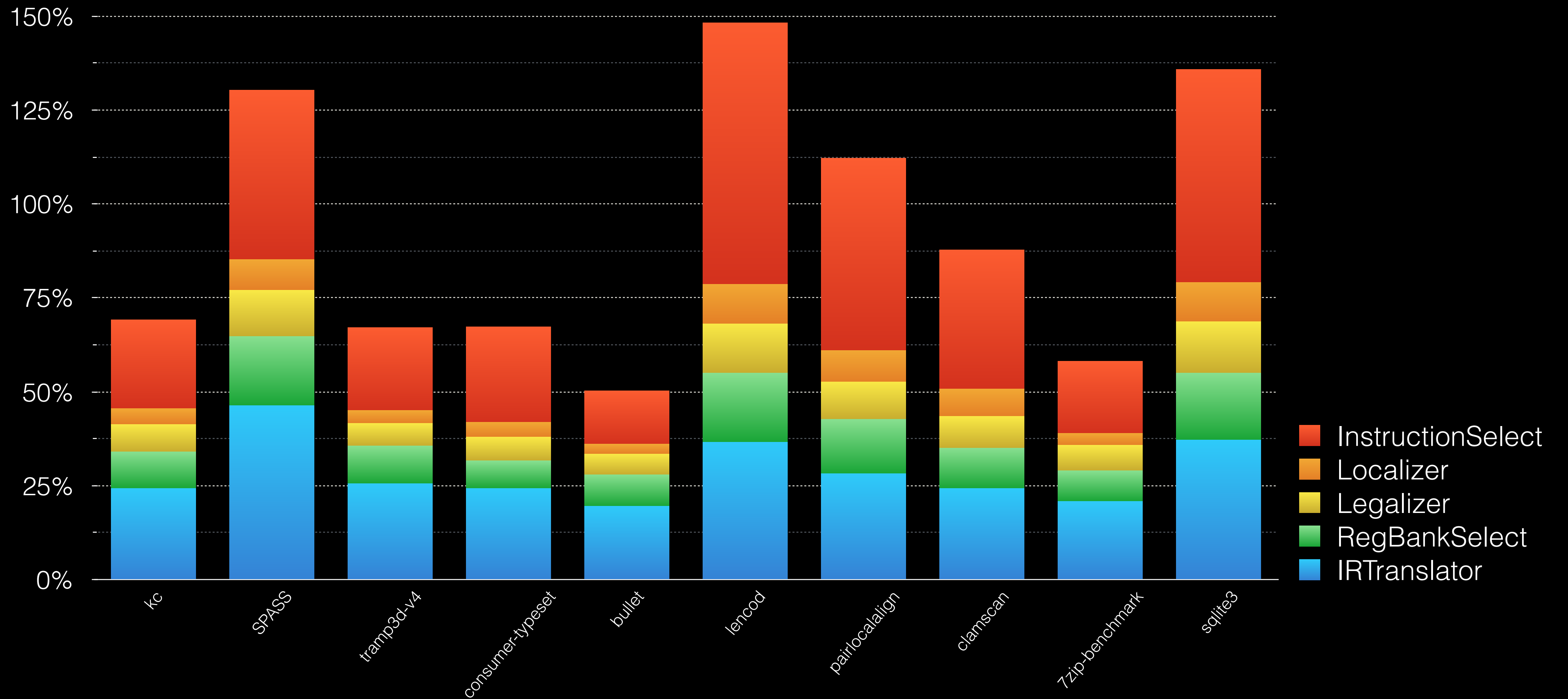


Performance

Compile time: per pass

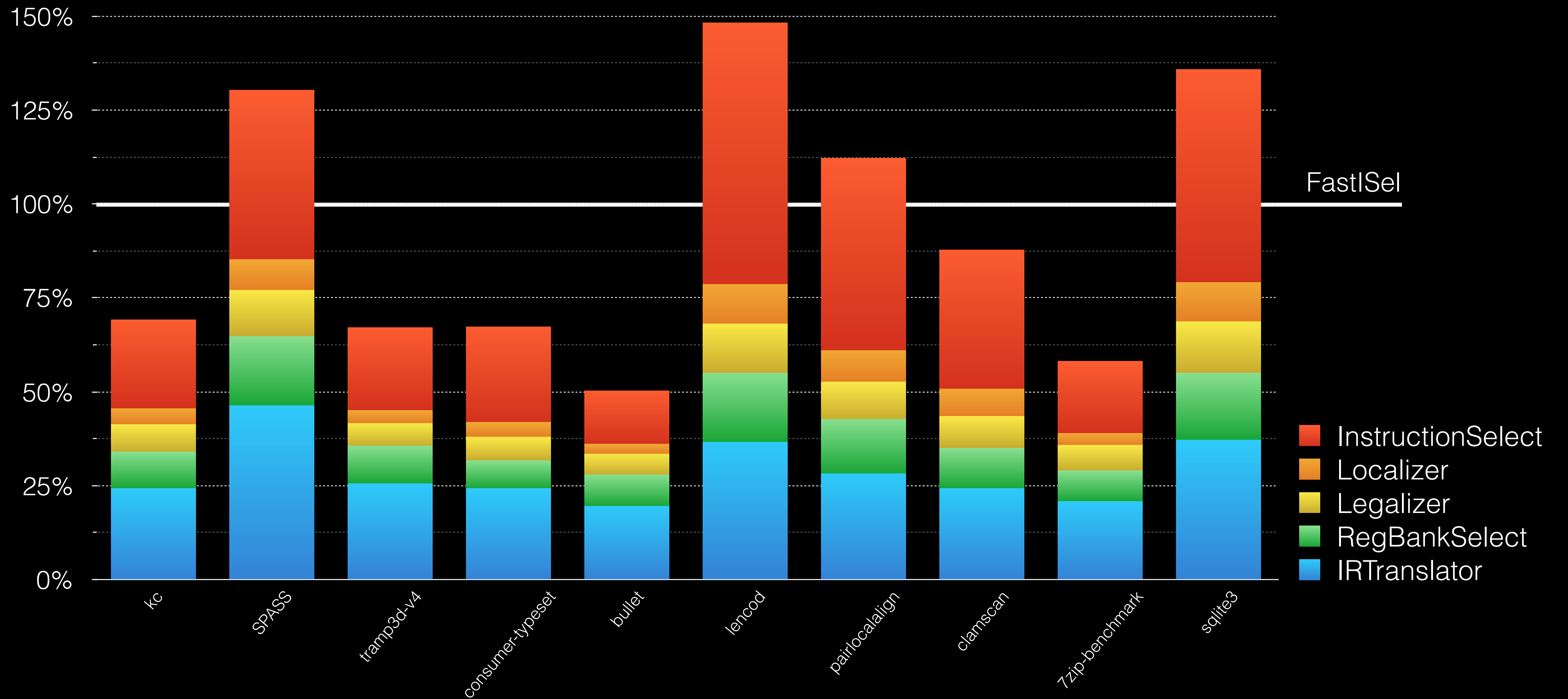
Performance

Compile time: per pass



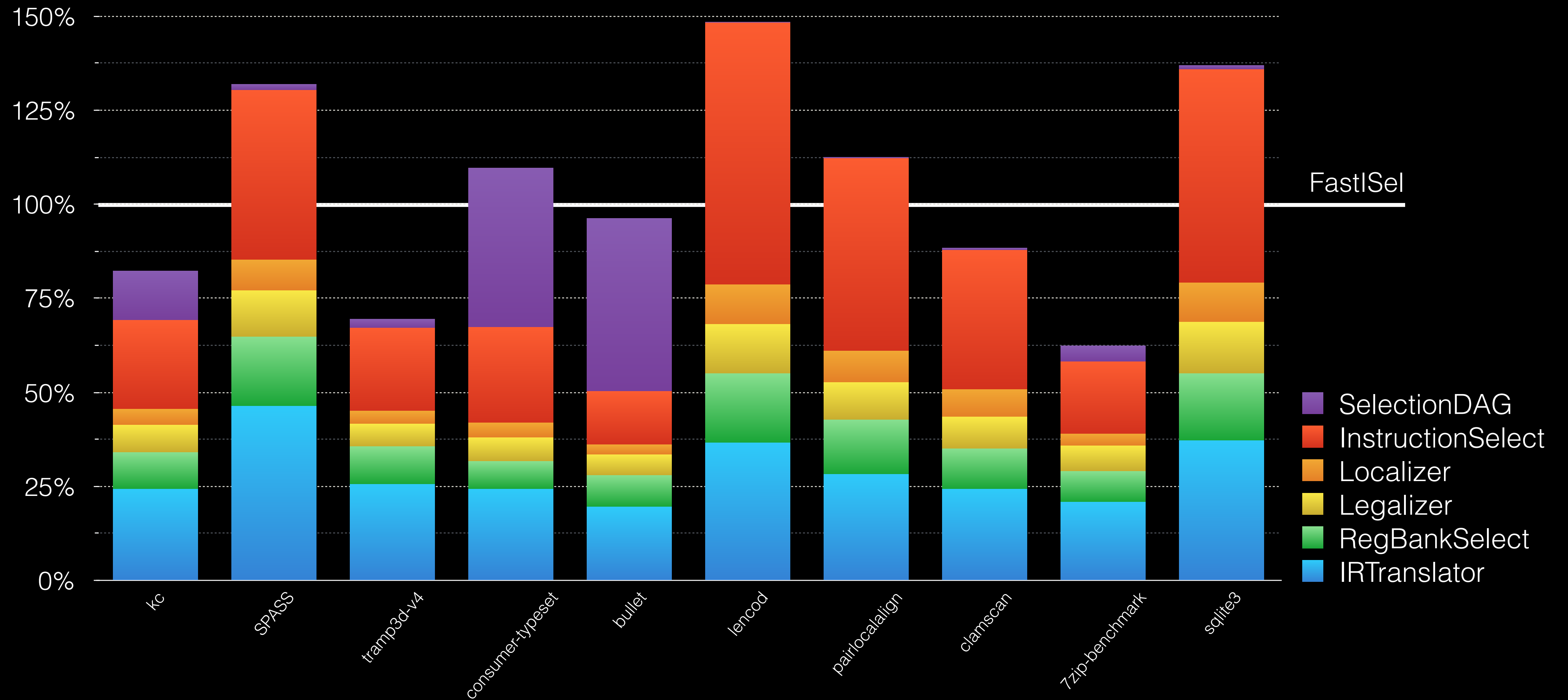
Performance

Compile time: per pass



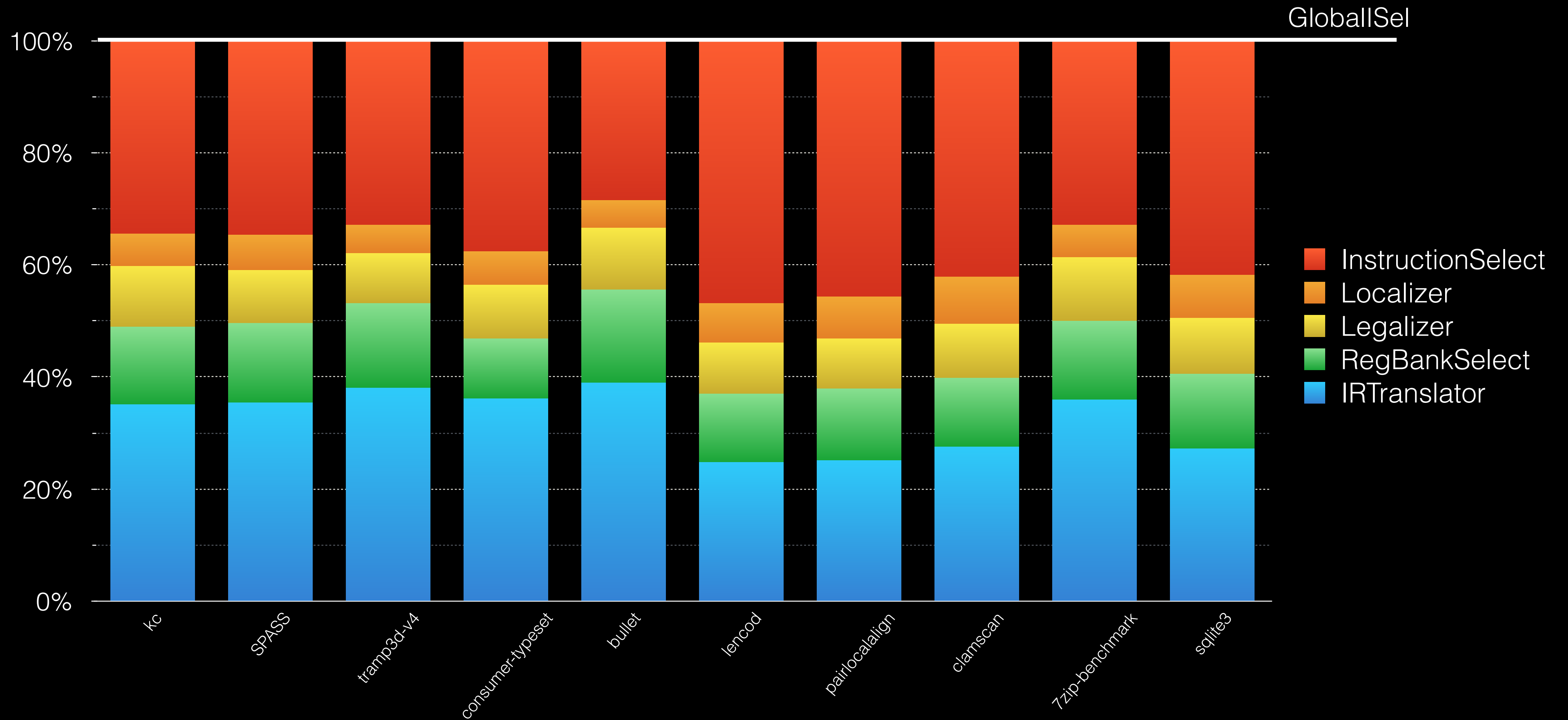
Performance

Compile time: per pass



Performance

Compile time: per pass, normalized

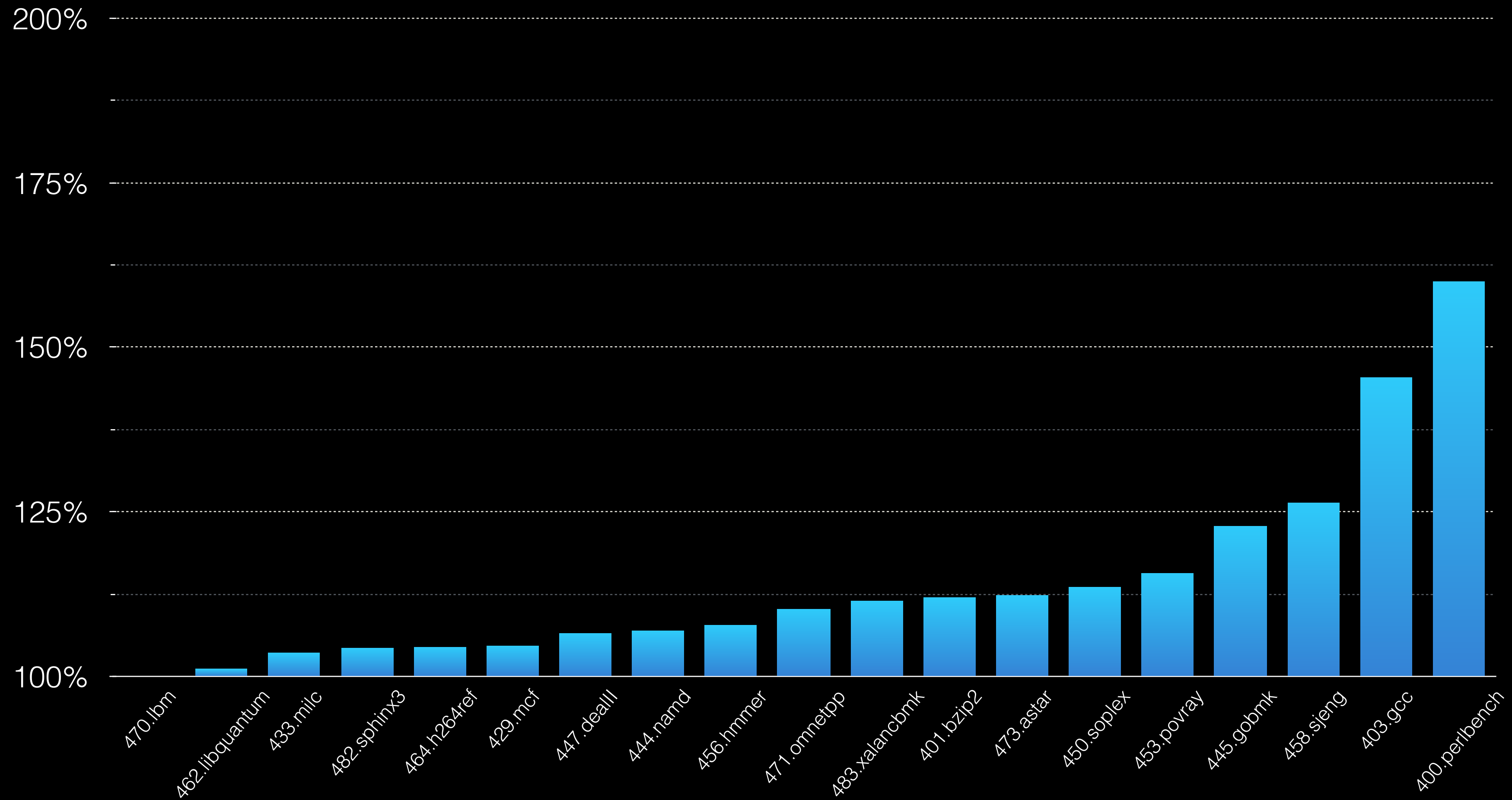


Performance

Runtime: SPEC2006

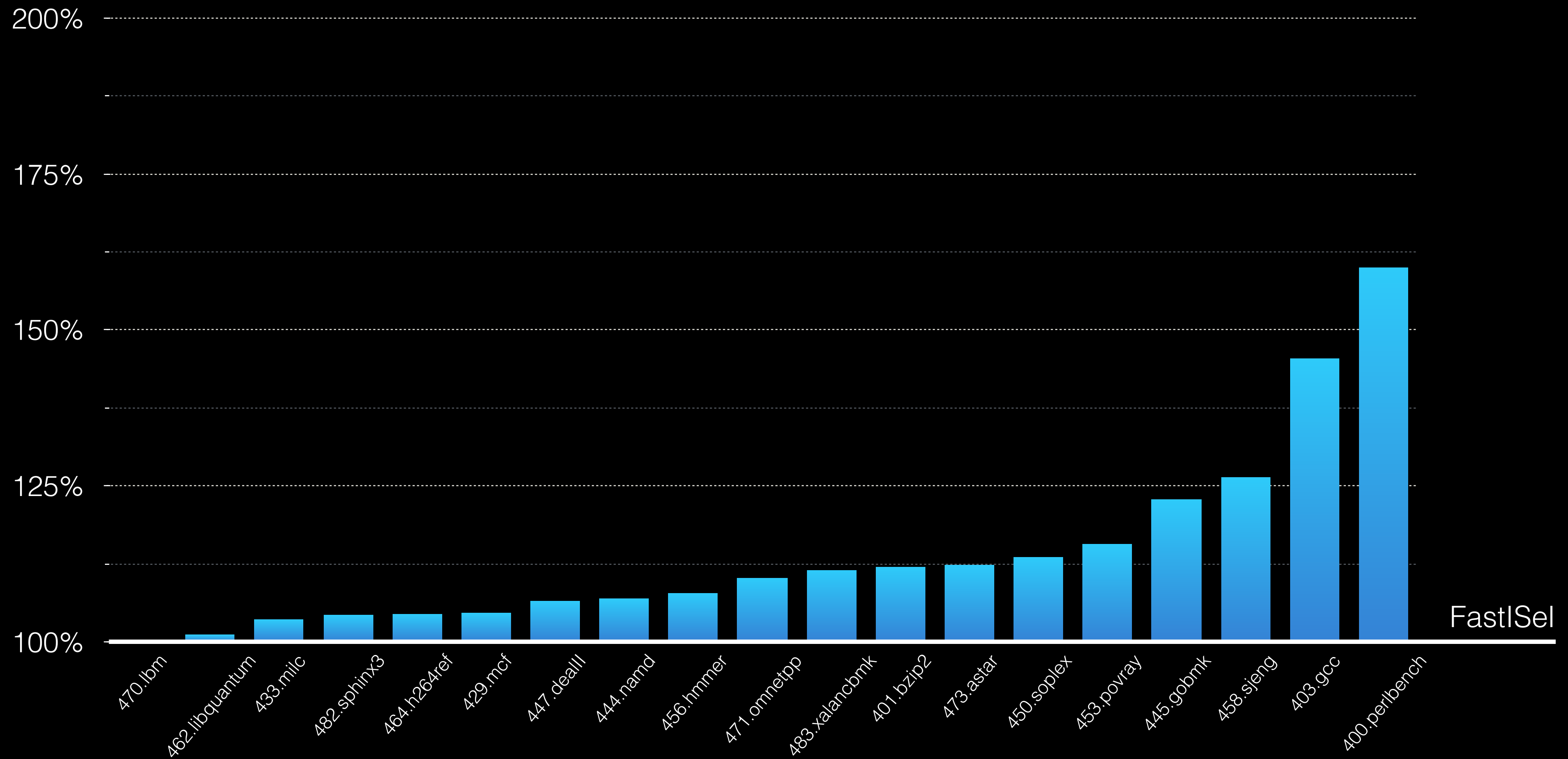
Performance

Runtime: SPEC2006



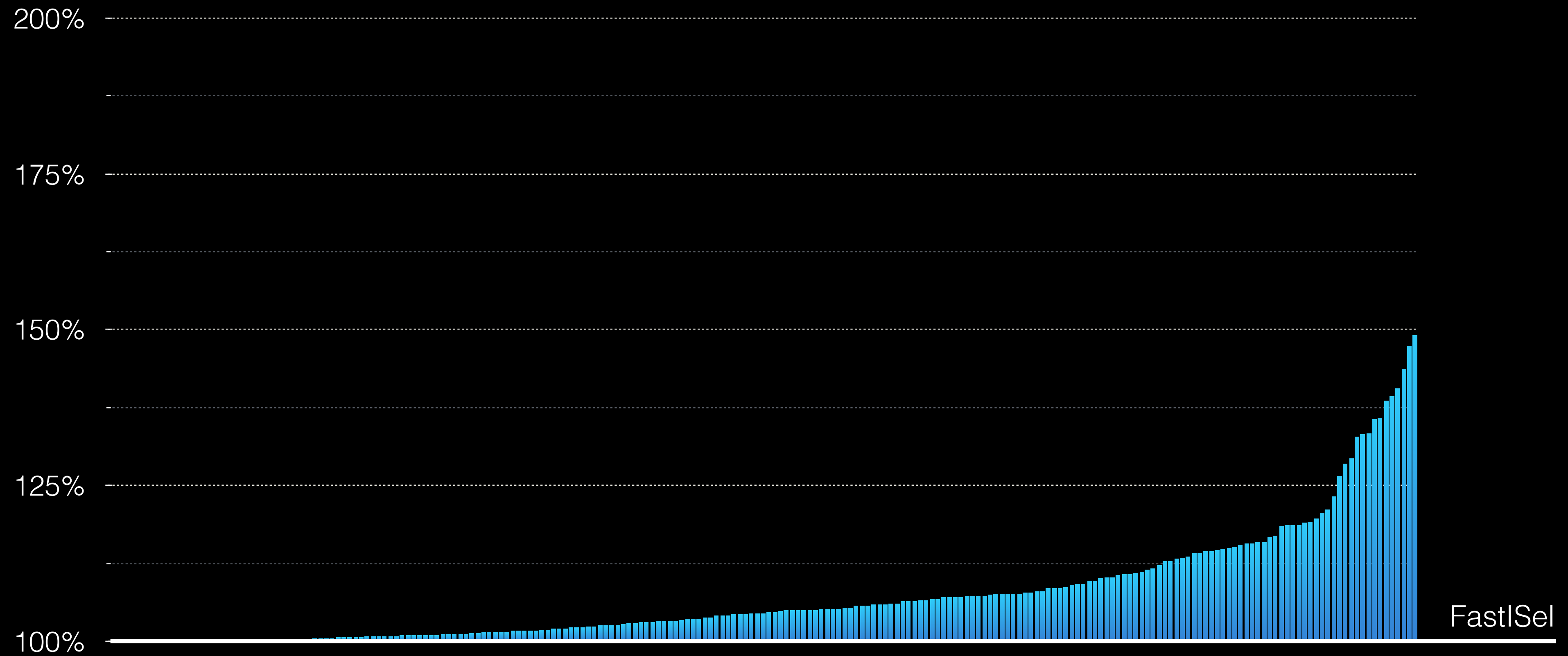
Performance

Runtime: SPEC2006



Performance

Runtime: Test Suite



Future

Future

Beyond -00

Future

Beyond -O0

- Flesh out the pass pipeline

Future

Beyond -O0

- Flesh out the pass pipeline
- Kill Localizer

Future

Beyond -O0

- Flesh out the pass pipeline
- Kill Localizer
- Kill CodeGenPrepare

Future

Beyond the SelectionDAG GlobalSelEmitter

Future

Beyond the SelectionDAG GlobalSelEmitter

- Emit legality info

Future

Beyond the SelectionDAG GlobalSelEmitter

- Emit legality info

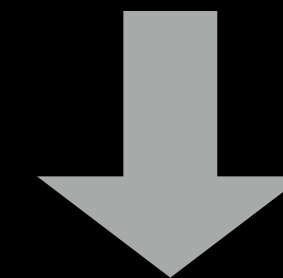
```
def : Pat<  
    (i32 (mul (GPR32:$Rn, GPR32:$Rm))),  
    (...)  
>;
```

Future

Beyond the SelectionDAG GlobalSelEmitter

- Emit legality info

```
def : Pat<  
  (i32 (mul (GPR32:$Rn, GPR32:$Rm))),  
  (...)  
>;
```



```
setAction({G_MUL, 0, s32}, Legal)
```

Future

Beyond the SelectionDAG GlobalSelEmitter

- Emit legality info
- Emit register bank mapping

Future

Beyond the SelectionDAG GlobalSelEmitter

- Emit legality info
- Emit register bank mapping
- Support pure GlobalSel patterns

Future

Beyond SelectionDAG

Future

Beyond SelectionDAG

- Feature parity (remember vectors?)

Future

Beyond SelectionDAG

- Feature parity (remember vectors?)
- Improved performance

Future

Beyond SelectionDAG

- Feature parity (remember vectors?)
- Improved performance
- More testing!

Future

Combines: Leveraging Patterns

Future

Combines: Leveraging Patterns

```
def : Pat<  
  (f32 (fma (fneg (FPR32:$Rn), FPR32:$Rm, FPR32:$Ra)),  
   (FMSUBSrrr FPR32:$Rn, FPR32:$Rm, FPR32:$Ra)  
>;
```

Future

Combines: Leveraging Patterns

```
def Mul2ToShl1 : Combine<  
  (mul type0:$src0, 2),  
  (shl type0:$src0, 1),  
>;
```

Future

Combines: Leveraging Patterns

```
def Mul2ToShl1 : Combine<  
  (mul type0:$src0, 2),  
  (shl type0:$src0, 1),  
>;
```



TableGen
Backend

Future

Combines: Leveraging Patterns

```
def Mul2ToShl1 : Combine<  
  (mul type0:$src0, 2),  
  (shl type0:$src0, 1),  
>;
```

TableGen
Backend

Inst
Combine
.inc

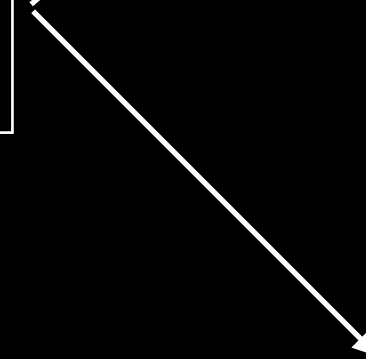
Future

Combines: Leveraging Patterns

```
def Mul2ToShl1 : Combine<  
  (mul type0:$src0, 2),  
  (shl type0:$src0, 1),  
>;
```



TableGen
Backend



Inst
Combine
.inc

GISEL
Combine
.inc

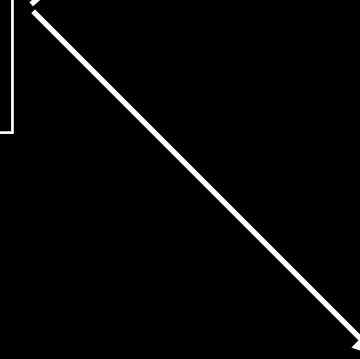
Future

Combines: Leveraging Patterns

```
def Mul2ToShl1 : Combine<  
  (mul type0:$src0, 2),  
  (shl type0:$src0, 1),  
>;
```



TableGen
Backend



Inst
Combine
.inc

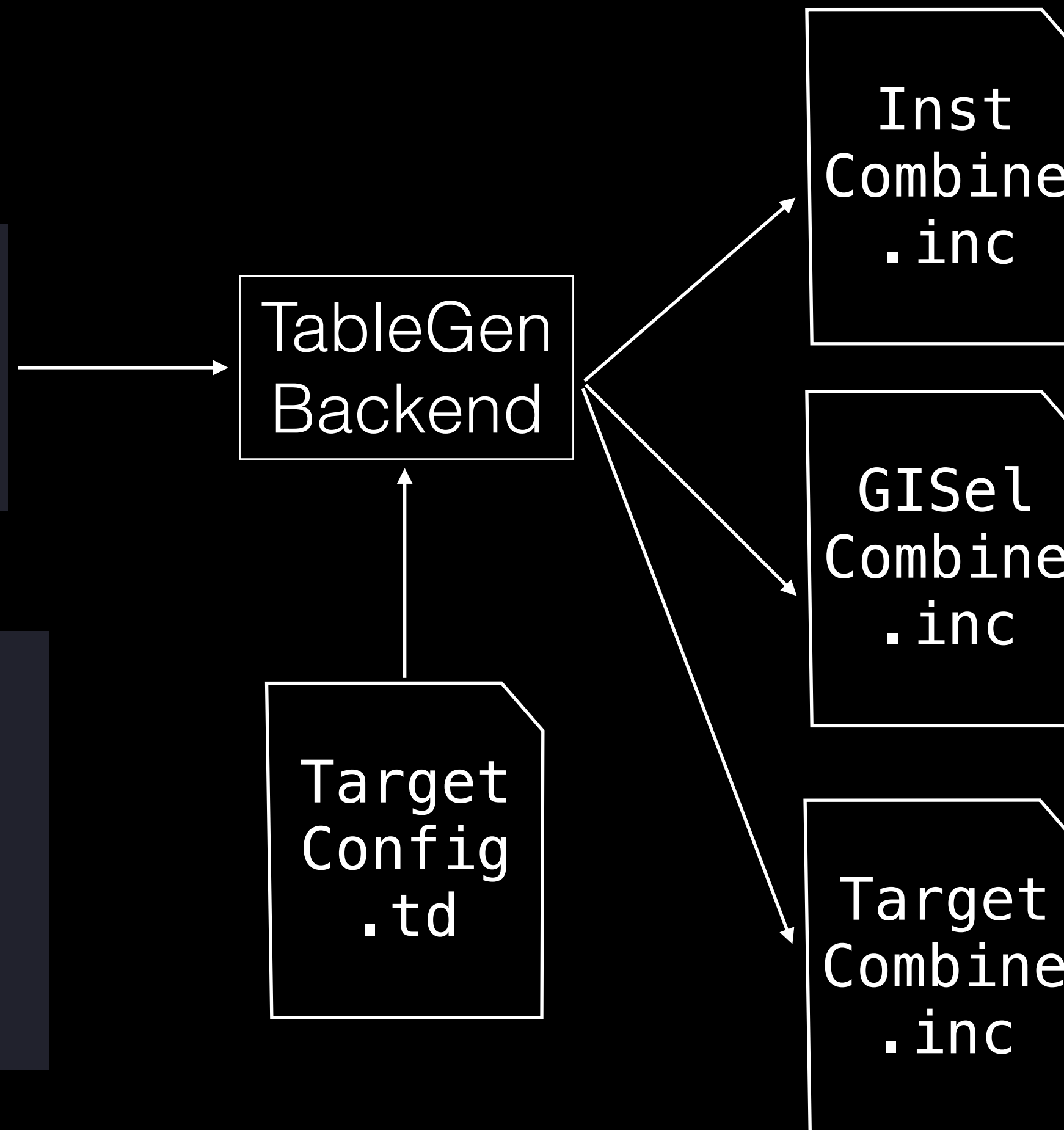
GISEL
Combine
.inc

Future

Combines: Leveraging Patterns

```
def Mul2ToShl1 : Combine<  
  (mul type0:$src0, 2),  
  (shl type0:$src0, 1),  
>;
```

```
def : RunWhen<Mul2ToShl1,  
      [BeforeLegalized,  
       AfterLegalized]>  
  
def MyPattern : (MyOp(...));  
def : RunWhen<MyPattern,  
      [AfterRegBankSelect]>
```



Future

Tools to Help the Transition

Future

Tools to Help the Transition

- Better `.mir` test format

Future

Tools to Help the Transition

- Better `.mir` test format
- Better MachineInstr API

Future

Tools to Help the Transition

- Better `.mir` test format
- Better MachineInstr API
- Regression tests generator

Future

Tools to Help the Transition

- Better `.mir` test format
- Better MachineInstr API
- Regression tests generator
- SDNode to GMIR migrator

Contributions Welcome!

Questions?