**Lifecycle of Java Bytecode through C2:**

**Java Source:**

public int doSum1(int b, int c , int cond) {

int ret = 0;

if (cond == 1)

ret = b + c;

else

ret = 10;

return ret + 20;

}

**Java Bytecode:**

public int doSum1(int, int, int);

descriptor: (III)I

flags: (0x0001) ACC\_PUBLIC

Code:

stack=2, locals=5, args\_size=4

0: iconst\_0

1: istore 4

3: iload\_3

4: iconst\_1

5: if\_icmpne 16

8: iload\_1

9: iload\_2

10: iadd

11: istore 4

13: goto 20

16: bipush 10

18: istore 4

20: iload 4

22: bipush 20

24: iadd

25: ireturn

LineNumberTable:

line 6: 0

line 7: 3

line 8: 8

line 10: 16

line 11: 20

StackMapTable: number\_of\_entries = 2

frame\_type = 252 /\* append \*/

offset\_delta = 16

locals = [ int ]

frame\_type = 3 /\* same \*/

**Ideal Graph:**

(gdb) p root()->dump(100)

26 ConI === 0 [[ 27 ]] #int:1

13 Parm === 3 [[ 27 ]] Parm3: int !jvms: test::doSum1 @ bci:-1

27 CmpI === \_ 13 26 [[ 28 ]] !jvms: test::doSum1 @ bci:5

12 Parm === 3 [[ 35 ]] Parm2: int !jvms: test::doSum1 @ bci:-1

11 Parm === 3 [[ 35 ]] Parm1: int !jvms: test::doSum1 @ bci:-1

28 Bool === \_ 27 [[ 29 ]] [ne] !jvms: test::doSum1 @ bci:5

5 Parm === 3 [[ 29 ]] Control !jvms: test::doSum1 @ bci:-1

39 ConI === 0 [[ 45 ]] #int:20

35 AddI === \_ 11 12 [[ 45 ]] !jvms: test::doSum1 @ bci:10

29 If === 5 28 [[ 30 31 ]] P=0.967129, C=33069.000000 !jvms: test::doSum1 @ bci:5

45 AddI === \_ 35 39 [[ 42 ]] !orig=[40] !jvms: test::doSum1 @ bci:24

44 ConI === 0 [[ 42 ]] #int:30

3 Start === 3 0 [[ 3 5 6 7 8 9 13 11 12 ]] #{0:control, 1:abIO, 2:memory, 3:rawptr:BotPTR, 4:return\_address, 5:test:NotNull \*, 6:int, 7:int, 8:int}

31 IfFalse === 29 [[ 36 ]] #0 !jvms: test::doSum1 @ bci:5

30 IfTrue === 29 [[ 36 ]] #1 !jvms: test::doSum1 @ bci:5

42 Phi === 36 44 45 [[ 41 ]] #int !orig=[40] !jvms: test::doSum1 @ bci:24

9 Parm === 3 [[ 41 ]] ReturnAdr !jvms: test::doSum1 @ bci:-1

8 Parm === 3 [[ 41 ]] FramePtr !jvms: test::doSum1 @ bci:-1

7 Parm === 3 [[ 41 ]] Memory Memory: @BotPTR \*+bot, idx=Bot; !jvms: test::doSum1 @ bci:-1

6 Parm === 3 [[ 41 ]] I\_O !jvms: test::doSum1 @ bci:-1

36 Region === 36 30 31 [[ 36 41 42 ]] !jvms: test::doSum1 @ bci:20

41 Return === 36 6 7 8 9 returns 42 [[ 0 ]]

0 Root === 0 41 [[ 0 1 3 39 26 37 44 ]] inner

$2 = void

**Lowered Ideal Machine Graph:**

2585 TracePhase tp("matcher", &timers[\_t\_matcher]);

(gdb)

2586 matcher.match();

(gdb)

2587 if (failing()) {

(gdb) p root()->dump(100)

9 MachProj === 8 [[ 6 ]] #8 !jvms: test::doSum1 @ bci:-1

22 MachProj === 8 [[ 18 ]] #7 !jvms: test::doSum1 @ bci:-1

21 MachProj === 8 [[ 18 ]] #6 !jvms: test::doSum1 @ bci:-1

6 compI\_rReg\_imm === \_ 9 [[ 5 ]] #1/0x00000001

7 MachProj === 8 [[ 5 ]] #0/unmatched !jvms: test::doSum1 @ bci:-1

18 addI\_rReg === \_ 21 22 [[ 19 17 ]]

5 jmpCon === 7 6 [[ 4 10 ]] P=0.967129, C=33069.000000 !jvms: test::doSum1 @ bci:5

17 addI\_rReg\_imm === \_ 18 [[ 20 15 ]] #20/0x00000014 !jvms: test::doSum1 @ bci:24

16 loadConI === \_ [[ 15 ]] [3400016]

8 Start === 8 1 [[ 8 7 9 11 12 13 14 21 22 ]] #{0:control, 1:abIO, 2:memory, 3:rawptr:BotPTR, 4:return\_address, 5:test:NotNull \*, 6:int, 7:int, 8:int} !jvms: test::doSum1 @ bci:20

10 IfFalse === 5 [[ 3 ]] #0 !jvms: test::doSum1 @ bci:5

4 IfTrue === 5 [[ 3 ]] #1 !jvms: test::doSum1 @ bci:5

15 Phi === 3 16 17 [[ 2 ]] #int !jvms: test::doSum1 @ bci:24

14 MachProj === 8 [[ 2 ]] #4 !jvms: test::doSum1 @ bci:-1

13 MachProj === 8 [[ 2 ]] #3 !jvms: test::doSum1 @ bci:-1

12 MachProj === 8 [[ 2 ]] #2/unmatched Memory: @BotPTR \*+bot, idx=Bot; !jvms: test::doSum1 @ bci:-1

11 MachProj === 8 [[ 2 ]] #1/unmatched !jvms: test::doSum1 @ bci:-1

3 Region === 3 4 10 [[ 3 2 15 ]] !jvms: test::doSum1 @ bci:20

2 Ret === 3 11 12 13 14 15 [[ 1 ]]

1 Root === 1 2 [[ 1 8 0 ]] inner

$3 = void

2597 check\_node\_count(0, "out of nodes matching instructions");

(gdb)

2598 if (failing()) {

(gdb)

2602 print\_method(PHASE\_MATCHING, 2);

(gdb)

2605 PhaseCFG cfg(node\_arena(), root(), matcher);

(gdb)

2606 \_cfg = &cfg;

(gdb)

2608 TracePhase tp("scheduler", &timers[\_t\_scheduler]);

(gdb)

2609 bool success = cfg.do\_global\_code\_motion();

**Machine CFG:**

(gdb) p cfg.dump()

--- CFG --- 5 BBs

B1: # out( B2 ) <- in( B4 ) Freq: 1

1 Root === 1 2 [[ 1 8 16 ]] inner

B2: # out( B5 B3 ) <- in( B1 ) Freq: 1

8 Start === 8 1 [[ 8 7 9 11 12 13 14 21 22 0 ]] #{0:control, 1:abIO, 2:memory, 3:rawptr:BotPTR, 4:return\_address, 5:test:NotNull \*, 6:int, 7:int, 8:int} !jvms: test::doSum1 @ bci:20

7 MachProj === 8 [[ 5 ]] #0/unmatched !jvms: test::doSum1 @ bci:-1

9 MachProj === 8 [[ 6 ]] #8 !jvms: test::doSum1 @ bci:-1

11 MachProj === 8 [[ 2 ]] #1/unmatched !jvms: test::doSum1 @ bci:-1

12 MachProj === 8 [[ 2 ]] #2/unmatched Memory: @BotPTR \*+bot, idx=Bot; !jvms: test::doSum1 @ bci:-1

13 MachProj === 8 [[ 2 ]] #3 !jvms: test::doSum1 @ bci:-1

14 MachProj === 8 [[ 2 ]] #4 !jvms: test::doSum1 @ bci:-1

21 MachProj === 8 [[ 18 ]] #6 !jvms: test::doSum1 @ bci:-1

22 MachProj === 8 [[ 18 ]] #7 !jvms: test::doSum1 @ bci:-1

0 Con === 8 [[]] #top

6 compI\_rReg\_imm === \_ 9 [[ 5 ]] #1/0x00000001

5 jmpCon === 7 6 [[ 4 10 ]] P=0.967129, C=33069.000000 !jvms: test::doSum1 @ bci:5

4 IfTrue === 5 [[ 29 ]] #1 !jvms: test::doSum1 @ bci:5

10 IfFalse === 5 [[ 30 ]] #0 !jvms: test::doSum1 @ bci:5

B5: # out( B4 ) <- in( B2 ) Freq: 0.967129

29 Region === 29 4 [[ 29 28 ]]

16 loadConI === 1 [[ 15 ]] #30/0x0000001e

28 jmpDir === 29 [[ 3 ]] !orig=26

B3: # out( B4 ) <- in( B2 ) Freq: 0.0328707

30 Region === 30 10 [[ 30 27 ]]

18 addI\_rReg === \_ 21 22 [[ 19 17 ]]

19 MachProj === 18 [[]] #1

17 addI\_rReg\_imm === \_ 18 [[ 20 15 ]] #20/0x00000014 !jvms: test::doSum1 @ bci:24

20 MachProj === 17 [[]] #1

27 jmpDir === 30 [[ 3 ]] !orig=26

B4: # out( B1 ) <- in( B5 B3 ) Freq: 1

3 Region === 3 28 27 [[ 3 2 15 ]] !jvms: test::doSum1 @ bci:20

15 Phi === 3 16 17 [[ 2 ]] #int !jvms: test::doSum1 @ bci:24

2 Ret === 3 11 12 13 14 15 [[ 1 ]]

$7 = void

(gdb) n

2620 \_regalloc = &regalloc;

(gdb)

2622 TracePhase tp("regalloc", &timers[\_t\_registerAllocation]);

(gdb)

2626 \_regalloc->Register\_Allocate();

**RA Live Ranges and allocation**

(gdb)

L3 selecting degree 0 degrees\_of\_freedom 0

2 [rS582,rS583], #2 EffDeg: 0 Def: N14 Cost: 3 Area: 6.1 Score: 1.5 Lo Bound Trivial

L3 selected [rS582,rS583] from [rS582,rS583]

L8 selecting degree 0 degrees\_of\_freedom 1048575

0 [R10-rS703], #1048575 EffDeg: 0 Dead Cost: 0 Area: 0 Score:1e+35 Lo Trivial

L10 selecting degree 0 degrees\_of\_freedom 1048575

0 [R10-rS703], #1048575 EffDeg: 0 Dead Cost: 0 Area: 0 Score:1e+35 Lo Trivial

L12 selecting degree 0 degrees\_of\_freedom 1048575

0 [R10-rS703], #1048575 EffDeg: 0 Dead Cost: 0 Area: 0 Score:1e+35 Lo Trivial

L1 selecting degree 0 degrees\_of\_freedom 0

1 [R8], #1 EffDeg: 0 Def: N9 Cost: 3 Area: 7 Score: 1.3 Lo Bound Trivial

L1 selected [R8] from [R8]

L2 selecting degree 0 degrees\_of\_freedom 0

2 [RSP,RSP\_H], #2 EffDeg: 0 Def: N13 Cost:1e+12 Area: 7.1 Score:1e+17 Lo Bound Trivial

L2 selected [RSP,RSP\_H] from [RSP,RSP\_H]

L4 selecting degree 0 degrees\_of\_freedom 0

1 [RDX], #1 EffDeg: 0 MultiDef (N21 N18 ) Cost: 1.2 Area: 4 Score:0.18 Lo Copy Bound Trivial

L4 selected [RDX] from [RDX]

L5 selecting degree 0 degrees\_of\_freedom 0

1 [RCX], #1 EffDeg: 0 Def: N22 Cost: 1.1 Area: 3 Score:0.33 Lo Bound Trivial

L5 selected [RCX] from [RCX]

L6 selecting degree 0 degrees\_of\_freedom 0

1 [RFLAGS], #1 EffDeg: 0 Def: N6 Cost: 1 Area: 0 Score:1e+35 Lo Bound Trivial

L6 selected [RFLAGS] from [RFLAGS]

L7 selecting degree 0 degrees\_of\_freedom 0

1 [RAX], #1 EffDeg: 0 MultiDef (N16 N31 N17 N15 ) Cost: 3.1 Area: 1 Score:1e+17 Lo Copy Bound Trivial

L7 selected [RAX] from [RAX]

L9 selecting degree 0 degrees\_of\_freedom 0

1 [RFLAGS], #1 EffDeg: 0 Def: N19 Cost:0.033 Area: 0 Score:1e+35 Lo Bound Trivial

L9 selected [RFLAGS] from [RFLAGS]

L11 selecting degree 0 degrees\_of\_freedom 0

1 [RFLAGS], #1 EffDeg: 0 Def: N20 Cost:0.033 Area: 0 Score:1e+35 Lo Bound Trivial

L11 selected [RFLAGS] from [RFLAGS]

2629 if (failing()) {

(gdb) p cfg.dump()

**Post Allocation CFG:**

--- CFG --- 5 BBs

B1: # out( B2 ) <- in( B4 ) Freq: 1

1 Root === 1 2 [[ 1 8 16 ]] inner

B2: # out( B5 B3 ) <- in( B1 ) Freq: 1

8 Start === 8 1 [[ 8 7 9 11 12 13 14 21 22 0 ]] #{0:control, 1:abIO, 2:memory, 3:rawptr:BotPTR, 4:return\_address, 5:test:NotNull \*, 6:int, 7:int, 8:int} !jvms: test::doSum1 @ bci:20

7 MachProj === 8 [[ 5 ]] #0/unmatched !jvms: test::doSum1 @ bci:-1

9 MachProj === 8 [[ 6 ]] #8 !jvms: test::doSum1 @ bci:-1

11 MachProj === 8 [[ 2 ]] #1/unmatched !jvms: test::doSum1 @ bci:-1

12 MachProj === 8 [[ 2 ]] #2/unmatched Memory: @BotPTR \*+bot, idx=Bot; !jvms: test::doSum1 @ bci:-1

13 MachProj === 8 [[ 2 ]] #3 !jvms: test::doSum1 @ bci:-1

14 MachProj === 8 [[ 2 ]] #4 !jvms: test::doSum1 @ bci:-1

21 MachProj === 8 [[ 18 ]] #6 !jvms: test::doSum1 @ bci:-1

22 MachProj === 8 [[ 18 ]] #7 !jvms: test::doSum1 @ bci:-1

0 Con === 8 [[]] #top

6 compI\_rReg\_imm === \_ 9 [[ 5 ]] #1/0x00000001

5 jmpCon === 7 6 [[ 4 10 ]] P=0.967129, C=33069.000000 !jvms: test::doSum1 @ bci:5

4 IfTrue === 5 [[ 29 ]] #1 !jvms: test::doSum1 @ bci:5

10 IfFalse === 5 [[ 30 ]] #0 !jvms: test::doSum1 @ bci:5

B5: # out( B4 ) <- in( B2 ) Freq: 0.967129

29 Region === 29 4 [[ 29 28 ]]

16 loadConI === 1 [[ 15 ]] #30/0x0000001e

28 jmpDir === 29 [[ 3 ]] !orig=26

B3: # out( B4 ) <- in( B2 ) Freq: 0.0328707

30 Region === 30 10 [[ 30 27 ]]

18 addI\_rReg === \_ 21 22 [[ 19 31 ]]

19 MachProj === 18 [[]] #1

31 TwoAddressSpillCopy === \_ 18 [[ 17 ]]

17 addI\_rReg\_imm === \_ 31 [[ 20 15 ]] #20/0x00000014 !jvms: test::doSum1 @ bci:24

20 MachProj === 17 [[]] #1

27 jmpDir === 30 [[ 3 ]] !orig=26

B4: # out( B1 ) <- in( B5 B3 ) Freq: 1

3 Region === 3 28 27 [[ 3 2 15 ]] !jvms: test::doSum1 @ bci:20

15 Phi === 3 16 17 [[ 2 ]] #int !jvms: test::doSum1 @ bci:24

2 Ret === 3 11 12 13 14 15 [[ 1 ]]

$8 = void

(gdb) n

2622 TracePhase tp("regalloc", &timers[\_t\_registerAllocation]);

(gdb)

2639 TracePhase tp("blockOrdering", &timers[\_t\_blockOrdering]);

(gdb)

2640 cfg.remove\_empty\_blocks();

(gdb)

2641 if (do\_freq\_based\_layout()) {

(gdb) p cfg.dump()

--- CFG --- 5 BBs

B1: # out( B2 ) <- in( B4 ) Freq: 1

1 Root === 1 2 [[ 1 8 16 ]] inner

B2: # out( B5 B3 ) <- in( B1 ) Freq: 1

8 Start === 8 1 [[ 8 7 9 11 12 13 14 21 22 0 ]] #{0:control, 1:abIO, 2:memory, 3:rawptr:BotPTR, 4:return\_address, 5:test:NotNull \*, 6:int, 7:int, 8:int} !jvms: test::doSum1 @ bci:20

7 MachProj === 8 [[ 5 ]] #0/unmatched !jvms: test::doSum1 @ bci:-1

9 MachProj === 8 [[ 6 ]] #8 !jvms: test::doSum1 @ bci:-1

11 MachProj === 8 [[ 2 ]] #1/unmatched !jvms: test::doSum1 @ bci:-1

12 MachProj === 8 [[ 2 ]] #2/unmatched Memory: @BotPTR \*+bot, idx=Bot; !jvms: test::doSum1 @ bci:-1

13 MachProj === 8 [[ 2 ]] #3 !jvms: test::doSum1 @ bci:-1

14 MachProj === 8 [[ 2 ]] #4 !jvms: test::doSum1 @ bci:-1

21 MachProj === 8 [[ 18 ]] #6 !jvms: test::doSum1 @ bci:-1

22 MachProj === 8 [[ 18 ]] #7 !jvms: test::doSum1 @ bci:-1

0 Con === 8 [[]] #top

6 compI\_rReg\_imm === \_ 9 [[ 5 ]] #1/0x00000001

5 jmpCon === 7 6 [[ 4 10 ]] P=0.967129, C=33069.000000 !jvms: test::doSum1 @ bci:5

4 IfTrue === 5 [[ 29 ]] #1 !jvms: test::doSum1 @ bci:5

10 IfFalse === 5 [[ 30 ]] #0 !jvms: test::doSum1 @ bci:5

B5: # out( B4 ) <- in( B2 ) Freq: 0.967129

29 Region === 29 4 [[ 29 28 ]]

16 loadConI === 1 [[ 15 ]] #30/0x0000001e

28 jmpDir === 29 [[ 3 ]] !orig=26

B3: # out( B4 ) <- in( B2 ) Freq: 0.0328707

30 Region === 30 10 [[ 30 27 ]]

18 addI\_rReg === \_ 21 22 [[ 19 31 ]]

19 MachProj === 18 [[]] #1

31 TwoAddressSpillCopy === \_ 18 [[ 17 ]]

17 addI\_rReg\_imm === \_ 31 [[ 20 15 ]] #20/0x00000014 !jvms: test::doSum1 @ bci:24

20 MachProj === 17 [[]] #1

27 jmpDir === 30 [[ 3 ]] !orig=26

B4: # out( B1 ) <- in( B5 B3 ) Freq: 1

3 Region === 3 28 27 [[ 3 2 15 ]] !jvms: test::doSum1 @ bci:20

15 Phi === 3 16 17 [[ 2 ]] #int !jvms: test::doSum1 @ bci:24

2 Ret === 3 11 12 13 14 15 [[ 1 ]]

$9 = void

(gdb) n

2642 PhaseBlockLayout layout(cfg);

(gdb)

2646 cfg.fixup\_flow();

(gdb)

2639 TracePhase tp("blockOrdering", &timers[\_t\_blockOrdering]);

(gdb) p cfg.dump()

--- CFG --- 5 BBs

N1: # out( B1 ) <- in( B3 ) Freq: 1

B1: # out( B4 B2 ) <- in( N1 ) Freq: 1

8 Start === 8 1 [[ 8 7 9 11 12 13 14 21 22 0 ]] #{0:control, 1:abIO, 2:memory, 3:rawptr:BotPTR, 4:return\_address, 5:test:NotNull \*, 6:int, 7:int, 8:int} !jvms: test::doSum1 @ bci:20

7 MachProj === 8 [[ 5 ]] #0/unmatched !jvms: test::doSum1 @ bci:-1

9 MachProj === 8 [[ 6 ]] #8 !jvms: test::doSum1 @ bci:-1

11 MachProj === 8 [[ 2 ]] #1/unmatched !jvms: test::doSum1 @ bci:-1

12 MachProj === 8 [[ 2 ]] #2/unmatched Memory: @BotPTR \*+bot, idx=Bot; !jvms: test::doSum1 @ bci:-1

13 MachProj === 8 [[ 2 ]] #3 !jvms: test::doSum1 @ bci:-1

14 MachProj === 8 [[ 2 ]] #4 !jvms: test::doSum1 @ bci:-1

21 MachProj === 8 [[ 18 ]] #6 !jvms: test::doSum1 @ bci:-1

22 MachProj === 8 [[ 18 ]] #7 !jvms: test::doSum1 @ bci:-1

0 Con === 8 [[]] #top

6 compI\_rReg\_imm === \_ 9 [[ 5 ]] #1/0x00000001

5 jmpCon === 7 6 [[ 4 10 ]] P=0.032871, C=33069.000000 !jvms: test::doSum1 @ bci:5

B2: # out( B3 ) <- in( B1 ) Freq: 0.967129

29 Region === 29 4 [[ 29 28 ]]

16 loadConI === 1 [[ 15 ]] #30/0x0000001e

B3: # out( N1 ) <- in( B2 B4 ) Freq: 1

3 Region === 3 28 27 [[ 3 2 15 ]] !jvms: test::doSum1 @ bci:20

15 Phi === 3 16 17 [[ 2 ]] #int !jvms: test::doSum1 @ bci:24

2 Ret === 3 11 12 13 14 15 [[ 1 ]]

B4: # out( B3 ) <- in( B1 ) Freq: 0.0328707

30 Region === 30 10 [[ 30 27 ]]

18 addI\_rReg === \_ 21 22 [[ 19 31 ]]

19 MachProj === 18 [[]] #1

31 TwoAddressSpillCopy === \_ 18 [[ 17 ]]

17 addI\_rReg\_imm === \_ 31 [[ 20 15 ]] #20/0x00000014 !jvms: test::doSum1 @ bci:24

20 MachProj === 17 [[]] #1

27 jmpDir === 30 [[ 3 ]] !orig=26

$10 = void

(gdb) n

2650 if( OptoPeephole ) {

(gdb)

2651 TracePhase tp("peephole", &timers[\_t\_peephole]);

(gdb)

2652 PhasePeephole peep( \_regalloc, cfg);

(gdb)

2653 peep.do\_transform();

(gdb)

2652 PhasePeephole peep( \_regalloc, cfg);

(gdb) n

2651 TracePhase tp("peephole", &timers[\_t\_peephole]);

(gdb)

2657 if (Matcher::require\_postalloc\_expand) {

(gdb) p cfg.dump()

--- CFG --- 5 BBs

N1: # out( B1 ) <- in( B3 ) Freq: 1

B1: # out( B4 B2 ) <- in( N1 ) Freq: 1

8 Start === 8 1 [[ 8 7 9 11 12 13 14 21 22 0 ]] #{0:control, 1:abIO, 2:memory, 3:rawptr:BotPTR, 4:return\_address, 5:test:NotNull \*, 6:int, 7:int, 8:int} !jvms: test::doSum1 @ bci:20

7 MachProj === 8 [[ 5 ]] #0/unmatched !jvms: test::doSum1 @ bci:-1

9 MachProj === 8 [[ 6 ]] #8 !jvms: test::doSum1 @ bci:-1

11 MachProj === 8 [[ 2 ]] #1/unmatched !jvms: test::doSum1 @ bci:-1

12 MachProj === 8 [[ 2 ]] #2/unmatched Memory: @BotPTR \*+bot, idx=Bot; !jvms: test::doSum1 @ bci:-1

13 MachProj === 8 [[ 2 ]] #3 !jvms: test::doSum1 @ bci:-1

14 MachProj === 8 [[ 2 ]] #4 !jvms: test::doSum1 @ bci:-1

21 MachProj === 8 [[ 18 ]] #6 !jvms: test::doSum1 @ bci:-1

22 MachProj === 8 [[ 18 ]] #7 !jvms: test::doSum1 @ bci:-1

0 Con === 8 [[]] #top

6 compI\_rReg\_imm === \_ 9 [[ 5 ]] #1/0x00000001

5 jmpCon === 7 6 [[ 4 10 ]] P=0.032871, C=33069.000000 !jvms: test::doSum1 @ bci:5

B2: # out( B3 ) <- in( B1 ) Freq: 0.967129

29 Region === 29 4 [[ 29 28 ]]

16 loadConI === 1 [[ 15 ]] #30/0x0000001e

B3: # out( N1 ) <- in( B2 B4 ) Freq: 1

3 Region === 3 28 27 [[ 3 2 15 ]] !jvms: test::doSum1 @ bci:20

15 Phi === 3 16 17 [[ 2 ]] #int !jvms: test::doSum1 @ bci:24

2 Ret === 3 11 12 13 14 15 [[ 1 ]]

B4: # out( B3 ) <- in( B1 ) Freq: 0.0328707

30 Region === 30 10 [[ 30 27 ]]

18 addI\_rReg === \_ 21 22 [[ 19 31 ]]

19 MachProj === 18 [[]] #1

31 TwoAddressSpillCopy === \_ 18 [[ 17 ]]

17 addI\_rReg\_imm === \_ 31 [[ 20 15 ]] #20/0x00000014 !jvms: test::doSum1 @ bci:24

20 MachProj === 17 [[]] #1

27 jmpDir === 30 [[ 3 ]] !orig=26

$11 = void

(gdb) n

2664 TracePhase tp("output", &timers[\_t\_output]);

(gdb)

2665 PhaseOutput output;

(gdb)

2666 output.Output();

(gdb)

============================= C2-compiled nmethod ==============================

----------------------- MetaData before Compile\_id = 29 ------------------------

{method}

- this oop: 0x00007efed594a460

- method holder: 'test'

- constants: 0x00007efed594a088 constant pool [64]/operands[5] {0x00007efed594a088} for 'test' cache=0x00007efed594a778

- access: 0x81000001 public

- name: 'doSum1'

- signature: '(III)I'

- max stack: 3

- max locals: 5

- size of params: 4

- method size: 13

- highest level: 4

- vtable index: 5

- i2i entry: 0x00007efeed010de0

- adapters: AHE@0x00007efef0362e70: 0xbaaa0000 i2c: 0x00007efeed0439e0 c2i: 0x00007efeed043acd c2iUV: 0x00007efeed043a94 c2iNCI: 0x00007efeed043b0a

- compiled entry 0x00007efeed043acd

- code size: 26

- code start: 0x00007efed594a440

- code end (excl): 0x00007efed594a45a

- method data: 0x00007efed594ac88

- checked ex length: 0

- linenumber start: 0x00007efed594a45a

- localvar length: 0

------------------------ OptoAssembly for Compile\_id = 29 -----------------------

#

# int ( test:NotNull \*, int, int, int )

#

#r018 rsi:rsi : parm 0: test:NotNull \*

#r016 rdx : parm 1: int

#r010 rcx : parm 2: int

#r004 r8 : parm 3: int

# -- Old rsp -- Framesize: 32 --

#r583 rsp+28: in\_preserve

#r582 rsp+24: return address

#r581 rsp+20: in\_preserve

#r580 rsp+16: saved fp register

#r579 rsp+12: pad2, stack alignment

#r578 rsp+ 8: pad2, stack alignment

#r577 rsp+ 4: Fixed slot 1

#r576 rsp+ 0: Fixed slot 0

#

000 N33: # out( B1 ) <- BLOCK HEAD IS JUNK Freq: 1

000 movl rscratch1, [j\_rarg0 + oopDesc::klass\_offset\_in\_bytes()] # compressed klass

decode\_klass\_not\_null rscratch1, rscratch1

cmpq rax, rscratch1 # Inline cache check

jne SharedRuntime::\_ic\_miss\_stub

nop # nops to align entry point

020 B1: # out( B4 B2 ) <- BLOCK HEAD IS JUNK Freq: 1

020 # stack bang (96 bytes)

pushq rbp # Save rbp

subq rsp, #16 # Create frame

02c cmpl R8, #1

030 je,s B4 P=0.032871 C=33069.000000

032 B2: # out( B3 ) <- in( B1 ) Freq: 0.967129

032 movl RAX, #30 # int

037 B3: # out( N33 ) <- in( B2 B4 ) Freq: 1

037 addq rsp, 16 # Destroy frame

popq rbp

movq rscratch1, poll\_offset[r15\_thread] #polling\_page\_address

testl rax, [rscratch1] # Safepoint: poll for GC

046 ret

047 B4: # out( B3 ) <- in( B1 ) Freq: 0.0328707

047 addl RDX, RCX # int

049 movl RAX, RDX # spill

04b addl RAX, #20 # int

04e jmp,s B3

--------------------------------------------------------------------------------

2667 if (failing()) return;