

%10:
%11 = sext i32 %4 to i64
%12 = icmp slt i64 %11, 256
%13 = select i1 %12, i64 %11, i64 256
%mul.i.i = shl i64 %7, 8
%mul2.i = mul nsw i32 %5, %3
%add3.i = add nsw i32 %mul2.i, %3
%idxprom4.i = sext i32 %add3.i to i64
%arrayidx5.i = getelementptr inbounds float, float* %1, i64 %idxprom4.i
%14 = icmp ugt i64 %13, 1
%umax = select i1 %14, i64 %13, i64 1
%min.iters.check = icmp ult i64 %umax, 8
br i1 %min.iters.check, label %pregion_for_entry.entry.i.preheader, label ... %vector.scevcheck

T

F

vector.scevcheck:
%ident.check = icmp ne i32 %5, 1
%15 = add nsw i64 %umax, -1
%16 = trunc i64 %7 to i32
%17 = shl i32 %16, 8
%18 = add i32 %17, %3
%19 = trunc i64 %15 to i32
%20 = add i32 %18, %19
%21 = icmp slt i32 %20, %18
%22 = icmp ugt i64 %15, 4294967295
%23 = or i1 %21, %22
%24 = or i1 %ident.check, %23
br i1 %24, label %pregion_for_entry.entry.i.preheader, label %vector.memcheck

T

F

vector.memcheck:
%scevgep = getelementptr float, float* %1, i64 %idxprom4.i
%scevgep1 = bitcast float* %scevgep to i8*
%uglygep = getelementptr i8, i8* %scevgep1, i64 1
%25 = trunc i64 %7 to i32
%26 = shl i32 %25, 8
%27 = add i32 %26, %3
%28 = sext i32 %27 to i64
%scevgep2 = getelementptr float, float* %2, i64 %28
%scevgep23 = bitcast float* %scevgep2 to i8*
%29 = add nsw i64 %umax, %28
%scevgep4 = getelementptr float, float* %2, i64 %29
%scevgep6 = getelementptr float, float* %0, i64 %28
%scevgep8 = getelementptr float, float* %0, i64 %29
%bound0 = icmp ult float* %arrayidx5.i, %scevgep4
%bound1 = icmp ugt i8* %uglygep, %scevgep23
%found.conflict = and i1 %bound0, %bound1
%bound010 = icmp ult float* %scevgep2, %scevgep8
%bound111 = icmp ult float* %scevgep6, %scevgep4
%found.conflict12 = and i1 %bound010, %bound111
%conflict.rdx = or i1 %found.conflict, %found.conflict12
br i1 %conflict.rdx, label %pregion_for_entry.entry.i.preheader, label ... %vector.ph

T

F

vector.ph:
%n.vec = and i64 %umax, -8
br label %vector.body

vector.body:
%index = phi i64 [0, %vector.ph], [%index.next, %vector.body]
%30 = add i64 %index, %mul.i.i
%31 = trunc i64 %30 to i32
%32 = mul nsw i32 %31, %5
%33 = add nsw i32 %32, %3
%34 = sext i32 %33 to i64
%35 = getelementptr inbounds float, float* %0, i64 %34
%36 = bitcast float* %35 to <8 x float>*
%wide.load = load <8 x float>, <8 x float>* %36, align 4, !tbaa !12,
... !alias.scope !16
%37 = load float, float* %arrayidx5.i, align 4, !tbaa !12, !alias.scope !19,
... !noalias !21
%broadcast.splatinsert = insertelement <8 x float> undef, float %37, i32 0
%broadcast.splat = shufflevector <8 x float> %broadcast.splatinsert, <8 x
... float> undef, <8 x i32> zeroinitializer
%38 = fdiv <8 x float> %wide.load, %broadcast.splat, !fpmath !23
%39 = getelementptr inbounds float, float* %2, i64 %34
%40 = bitcast float* %39 to <8 x float>*
store <8 x float> %38, <8 x float>* %40, align 4, !tbaa !12, !alias.scope
... !21, !noalias !16, !llvm.access.group !24
%index.next = add i64 %index, 8
%41 = icmp eq i64 %index.next, %n.vec
br i1 %41, label %middle.block, label %vector.body, !llvm.loop !26

T

F

middle.block:
%cmp.n = icmp eq i64 %umax, %n.vec
br i1 %cmp.n, label %gramschmidt_kernel2.exit, label ... %pregion_for_entry.entry.i.preheader

T

F

pregion_for_entry.entry.i.preheader:
%_local_id_x.0.ph = phi i64 [0, %vector.memcheck], [0, %vector.scevcheck
...], [0, %10], [%n.vec, %middle.block]
br label %pregion_for_entry.entry.i

pregion for entry.entry.i:
%_local_id_x.0 = phi i64 [%44, %pregion_for_entry.entry.i], [
... %_local_id_x.0.ph, %pregion_for_entry.entry.i.preheader]
%add1.i.i = add i64 %_local_id_x.0, %mul.i.i
%conv.i = trunc i64 %add1.i.i to i32
%mul.i = mul nsw i32 %conv.i, %5
%add.i = add nsw i32 %mul.i, %3
%idxprom.i = sext i32 %add.i to i64
%arrayidx.i = getelementptr inbounds float, float* %0, i64 %idxprom.i
%42 = load float, float* %arrayidx.i, align 4, !tbaa !12
%43 = load float, float* %arrayidx5.i, align 4, !tbaa !12
%div.i = fdiv float %42, %43, !fpmath !23
%arrayidx9.i = getelementptr inbounds float, float* %2, i64 %idxprom.i
store float %div.i, float* %arrayidx9.i, align 4, !tbaa !12,
... !llvm.access.group !24
%44 = add nuw i64 %_local_id_x.0, 1
%exitcond.not = icmp eq i64 %44, %umax
br i1 %exitcond.not, label %gramschmidt_kernel2.exit.loopexit, label
... %pregion_for_entry.entry.i, !llvm.loop !29

T

F

gramschmidt_kernel2.exit.loopexit:
br label %gramschmidt_kernel2.exit

gramschmidt_kernel2.exit:
ret void

CFG for '_pocl_kernel_gramschmidt_kernel2' function