

vector.scevcheck:
%9 = shl i64 %6, 8
%mul.i = mul nsw i32 %4, %3
%sub.i = add nsw i32 %3, -1
%mul2.i = mul nsw i32 %sub.i, %4
%10 = mul i32 %4, %3
%11 = trunc i64 %6 to i32
%12 = shl i32 %11, 8
%13 = add i32 %10, %12
%14 = icmp sgt i32 %13, 2147483392
%15 = add i32 %3, -1
%16 = mul i32 %15, %4
%17 = add i32 %16, %12
%18 = icmp sgt i32 %17, 2147483392
%19 = or i1 %14, %18
br i1 %19, label %preregion_for_entry.entry.i.preheader, label %vector.memcheck

T

F

vector.memcheck:
%20 = mul i32 %4, %3
%21 = trunc i64 %6 to i32
%22 = shl i32 %21, 8
%23 = add i32 %20, %22
%24 = sext i32 %23 to i64
%scevgep = getelementptr float, float* %2, i64 %24
%25 = add nsw i64 %24, 256
%scevgep5 = getelementptr float, float* %2, i64 %25
%scevgep7 = getelementptr float, float* %1, i64 %24
%scevgep9 = getelementptr float, float* %1, i64 %25
%26 = add i32 %3, -1
%27 = mul i32 %26, %4
%28 = add i32 %27, %22
%29 = sext i32 %28 to i64
%scevgep11 = getelementptr float, float* %2, i64 %29
%30 = add nsw i64 %29, 256
%scevgep13 = getelementptr float, float* %2, i64 %30
%scevgep15 = getelementptr float, float* %0, i64 %24
%scevgep17 = getelementptr float, float* %0, i64 %25
%scevgep19 = getelementptr float, float* %1, i64 %29
%scevgep21 = getelementptr float, float* %1, i64 %30
%bound0 = icmp ult float* %scevgep, %scevgep9
%bound1 = icmp ult float* %scevgep7, %scevgep5
%found.conflict = and i1 %bound0, %bound1
%bound023 = icmp ult float* %scevgep, %scevgep13
%bound124 = icmp ult float* %scevgep11, %scevgep5
%found.conflict25 = and i1 %bound023, %bound124
%conflict.rdx = or i1 %found.conflict, %found.conflict25
%bound026 = icmp ult float* %scevgep, %scevgep17
%bound127 = icmp ult float* %scevgep15, %scevgep5
%found.conflict28 = and i1 %bound026, %bound127
%conflict.rdx29 = or i1 %conflict.rdx, %found.conflict28
%bound030 = icmp ult float* %scevgep, %scevgep21
%bound131 = icmp ult float* %scevgep19, %scevgep5
%found.conflict32 = and i1 %bound030, %bound131
%conflict.rdx33 = or i1 %conflict.rdx29, %found.conflict32
%bound034 = icmp ult float* %scevgep7, %scevgep13
%bound135 = icmp ult float* %scevgep11, %scevgep9
%found.conflict36 = and i1 %bound034, %bound135
%conflict.rdx37 = or i1 %conflict.rdx33, %found.conflict36
%bound038 = icmp ult float* %scevgep7, %scevgep17
%bound139 = icmp ult float* %scevgep15, %scevgep9
%found.conflict40 = and i1 %bound038, %bound139
%conflict.rdx41 = or i1 %conflict.rdx37, %found.conflict40
%bound042 = icmp ult float* %scevgep7, %scevgep21
%bound143 = icmp ult float* %scevgep19, %scevgep9
%found.conflict44 = and i1 %bound042, %bound143
%conflict.rdx45 = or i1 %conflict.rdx41, %found.conflict44
br i1 %conflict.rdx45, label %preregion_for_entry.entry.i.preheader, label ... %vector.ph

T

F

preregion_for_entry.entry.i.preheader:
br label %preregion_for_entry.entry.i

vector.ph:
%broadcast.splatinsert = insertelement <8 x i64> undef, i64 %9, i32 0
%broadcast.splat = shufflevector <8 x i64> %broadcast.splatinsert, <8 x i64>
... undef, <8 x i32> zeroinitializer
%broadcast.splatinsert46 = insertelement <8 x i32> undef, i32 %4, i32 0
%broadcast.splat47 = shufflevector <8 x i32> %broadcast.splatinsert46, <8 x
... i32> undef, <8 x i32> zeroinitializer
br label %vector.body

vector.body:
%index = phi i64 [0, %vector.ph], [%index.next, %vector.body]
%vec.ind = phi <8 x i64> [<i64 0, i64 1, i64 2, i64 3, i64 4, i64 5, i64 6,
... i64 7>, %vector.ph], [%vec.ind.next, %vector.body]
%31 = add nuw nsw <8 x i64> %vec.ind, %broadcast.splat
%32 = trunc <8 x i64> %31 to <8 x i32>
%33 = icmp sgt <8 x i32> %broadcast.splat47, %32
%34 = extractelement <8 x i32> %32, i32 0
%35 = add nsw i32 %mul.i, %34
%36 = sext i32 %35 to i64
%37 = getelementptr inbounds float, float* %2, i64 %36
%38 = bitcast float* %37 to <8 x float>*
%wide.masked.load = call <8 x float> @llvm.masked.load.v8f32.p0v8f32(<8 x
... float>* %38, i32 4, <8 x i1> %33, <8 x float> undef), !tbaa !12, !alias.scope
... !16, !noalias !19
%39 = add nsw i32 %mul2.i, %34
%40 = sext i32 %39 to i64
%41 = getelementptr inbounds float, float* %2, i64 %40
%42 = bitcast float* %41 to <8 x float>*
%wide.masked.load48 = call <8 x float> @llvm.masked.load.v8f32.p0v8f32(<8 x
... float>* %42, i32 4, <8 x i1> %33, <8 x float> undef), !tbaa !12, !alias.scope
... !24
%43 = getelementptr inbounds float, float* %0, i64 %36
%44 = bitcast float* %43 to <8 x float>*
%wide.masked.load49 = call <8 x float> @llvm.masked.load.v8f32.p0v8f32(<8 x
... float>* %44, i32 4, <8 x i1> %33, <8 x float> undef), !tbaa !12, !alias.scope
... !25
%45 = fmul <8 x float> %wide.masked.load48, %wide.masked.load49
%46 = getelementptr inbounds float, float* %1, i64 %40
%47 = bitcast float* %46 to <8 x float>*
%wide.masked.load50 = call <8 x float> @llvm.masked.load.v8f32.p0v8f32(<8 x
... float>* %47, i32 4, <8 x i1> %33, <8 x float> undef), !tbaa !12, !alias.scope
... !26
%48 = fdiv <8 x float> %45, %wide.masked.load50, !fpmath !27
%49 = fsub <8 x float> %wide.masked.load, %48
%50 = bitcast float* %37 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %49, <8 x float>*
... %50, i32 4, <8 x i1> %33), !tbaa !12, !alias.scope !16, !noalias !19,
... !llvm.access.group !28
%51 = getelementptr inbounds float, float* %1, i64 %36
%52 = bitcast float* %51 to <8 x float>*
%wide.masked.load51 = call <8 x float> @llvm.masked.load.v8f32.p0v8f32(<8 x
... float>* %52, i32 4, <8 x i1> %33, <8 x float> undef), !tbaa !12, !alias.scope
... !30, !noalias !31
%53 = bitcast float* %43 to <8 x float>*
%wide.masked.load52 = call <8 x float> @llvm.masked.load.v8f32.p0v8f32(<8 x
... float>* %53, i32 4, <8 x i1> %33, <8 x float> undef), !tbaa !12, !alias.scope
... !25
%54 = fmul <8 x float> %wide.masked.load52, %wide.masked.load52
%55 = bitcast float* %46 to <8 x float>*
%wide.masked.load53 = call <8 x float> @llvm.masked.load.v8f32.p0v8f32(<8 x
... float>* %55, i32 4, <8 x i1> %33, <8 x float> undef), !tbaa !12, !alias.scope
... !26
%56 = fdiv <8 x float> %54, %wide.masked.load53, !fpmath !27
%57 = fsub <8 x float> %wide.masked.load51, %56
%58 = bitcast float* %51 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %57, <8 x float>*
... %58, i32 4, <8 x i1> %33), !tbaa !12, !alias.scope !30, !noalias !31,
... !llvm.access.group !28
%index.next = add i64 %index, 8
%vec.ind.next = add <8 x i64> %vec.ind, <i64 8, i64 8, i64 8, i64 8, i64 8,
... i64 8, i64 8, i64 8>
%59 = icmp eq i64 %index.next, 256
br i1 %59, label %adi_kernel4.exit.loopexit55, label %vector.body,
... !llvm.loop !32

T

F

if.then.i:
%add.i = add nsw i32 %mul.i, %conv.i
%idxprom.i = sext i32 %add.i to i64
%arrayidx.i = getelementptr inbounds float, float* %2, i64 %idxprom.i
%61 = load float, float* %arrayidx.i, align 4, !tbaa !12
%add3.i = add nsw i32 %mul2.i, %conv.i
%idxprom4.i = sext i32 %add3.i to i64
%arrayidx5.i = getelementptr inbounds float, float* %2, i64 %idxprom4.i
%62 = load float, float* %arrayidx5.i, align 4, !tbaa !12
%arrayidx9.i = getelementptr inbounds float, float* %0, i64 %idxprom.i
%63 = load float, float* %arrayidx9.i, align 4, !tbaa !12
%mul10.i = fmul float %62, %63
%arrayidx15.i = getelementptr inbounds float, float* %1, i64 %idxprom4.i
%64 = load float, float* %arrayidx15.i, align 4, !tbaa !12
%div.i = fdiv float %mul10.i, %64, !fpmath !27
%sub16.i = fsub float %61, %div.i
store float %sub16.i, float* %arrayidx.i, align 4, !tbaa !12,
... !llvm.access.group !28
%arrayidx24.i = getelementptr inbounds float, float* %1, i64 %idxprom.i
%65 = load float, float* %arrayidx24.i, align 4, !tbaa !12
%66 = load float, float* %arrayidx9.i, align 4, !tbaa !12
%mul33.i = fmul float %66, %66
%67 = load float, float* %arrayidx15.i, align 4, !tbaa !12
%div39.i = fdiv float %mul33.i, %67, !fpmath !27
%sub40.i = fsub float %65, %div39.i
store float %sub40.i, float* %arrayidx24.i, align 4, !tbaa !12,
... !llvm.access.group !28
br label %if.end.r_exit.i

if.end.r_exit.i:
%68 = add nuw nsw i64 % local_id x.0, 1
%exitcond = icmp eq i64 %68, 256
br i1 %exitcond, label %adi_kernel4.exit.loopexit, label
... %preregion_for_entry.entry.i, !llvm.loop !35

T

F

adi_kernel4.exit.loopexit:
br label %adi_kernel4.exit

adi_kernel4.exit.loopexit55:
br label %adi_kernel4.exit

adi_kernel4.exit:
ret void

CFG for '_pocl_kernel_adi_kernel4' function