%10 = shl i64 %6, 5 %cmp217.i = icmp sgt i32 %4, 0 %wide.trip.count.i = zext i32 %4 to i64 %11 = add nsw i64 %wide.trip.count.i, -1 %xtraiter.i = and i64 %wide.trip.count.i, 3 %12 = icmp ult i64 %11, 3%unroll iter.i = sub nuw nsw i64 %wide.trip.count.i, %xtraiter.i %lcmp.mod.i = icmp eq i64 %xtraiter.i, 0 br i1 %cmp217.i, label %pregion for entry.entry.i.us.preheader, label ... %atax kernel1.exit F pregion for entry.entry.i.us.preheader: br label %pregion for entry.entry.i.us pregion for entry.entry.i.us: % local id  $\bar{x}$ .0.us = phi i64 [ %39, %if.end.i.us ], [ 0, ... %pregion\_for\_entry.entry.i.us.preheader ]
%13 = add nuw nsw i64 %\_local\_id\_x.0.us, %10 %conv.i.us = trunc i64 %13 to i32%cmp.i.us = icmp slt i32 %conv.i.us, %3 br i1 %cmp.i.us, label %for.body.lr.ph.i.us, label %if.end.i.us for.body.lr.ph.i.us: %mul.i.us = mul nsw i32 %conv.i.us, %4 %sext.i.us = shl i64 %13, 32 %idxprom7.i.us = ashr exact i64 %sext.i.us, 32 %arrayidx8.i.us = getelementptr inbounds float, float\* %2, i64 %idxprom7.i.us %14 = sext i32 %mul.i.us to i64 %.pre.i1.us15 = load float, float\* %arrayidx8.i.us, align 4, !tbaa !12 br i1 %12, label %if.end.loopexit.unr-lcssa.i.us, label ... %for.body.i.us.preheader Τ F for.body.i.us.preheader: br label %for.body.i.us for.body.i.us: %niter.nsub.3.i7.us = phi i64 [ %niter.nsub.3.i.us, %for.body.i.us ], [ ... %unroll iter.i, %for.body.i.us.preheader ] %indvars.iv.next.3.i4.us = phi i64 [ %indvars.iv.next.3.i.us, %for.body.i.us ...], [0, %for.body.i.us.preheader] %15 = phi float [ %31, %for.body.i.us ], [ %.pre.i1.us15, ... %for.body.i.us.preheader ] %16 = add nsw i64 %indvars.iv.next.3.i4.us, %14 %arrayidx.i.us = getelementptr inbounds float, float\* %0, i64 %16 %17 = load float, float\* %arrayidx.i.us, align 4, !tbaa !12 %arrayidx5.i.us = getelementptr inbounds float, float\* %1, i64 ... %indvars.iv.next.3.i4.us %18 = load float, float\* %arrayidx5.i.us, align 4, !tbaa !12 %19 = tail call float @llvm.fmuladd.f32(float %17, float %18, float %15) #2 store float %19, float\* %arrayidx8.i.us, align 4, !tbaa !12, ...!llvm.access.group!16 %indvars.iv.next.i.us = or i64 %indvars.iv.next.3.i4.us, 1 %20 = add nsw i64 %indvars.iv.next.i.us, %14 %arrayidx.1.i.us = getelementptr inbounds float, float\* %0, i64 %20 %21 = load float, float\* %arrayidx.1.i.us, align 4, !tbaa !12 %arravidx5.1.i.us = getelementptr inbounds float, float\* %1, i64 ... %indvars.iv.next.i.us %22 = load float, float\* %arrayidx5.1.i.us, align 4, !tbaa !12 %23 = tail call float @llvm.fmuladd.f32(float %21, float %22, float %19) #2 store float %23, float\* %arrayidx8.i.us, align 4, !tbaa !12, ...!llvm.access.group!16 %indvars.iv.next.1.i.us = or i64 %indvars.iv.next.3.i4.us, 2 %24 = add nsw i64 %indvars.iv.next.1.i.us, %14 %arrayidx.2.i.us = getelementptr inbounds float, float\* %0, i64 %24 %25 = load float, float\* %arrayidx.2.i.us, align 4, !tbaa !12 %arrayidx5.2.i.us = getelementptr inbounds float, float\* %1, i64 ... %indvars.iv.next.1.i.us %26 = load float, float\* %arrayidx5.2.i.us, align 4, !tbaa !12 %27 = tail call float @llvm.fmuladd.f32(float %25, float %26, float %23) #2 store float %27, float\* %arrayidx8.i.us, align 4, !tbaa !12, ...!llvm.access.group!16 %indvars.iv.next.2.i.us = or i64 %indvars.iv.next.3.i4.us, 3 %28 = add nsw i64 %indvars.iv.next.2.i.us, %14 %arrayidx.3.i.us = getelementptr inbounds float, float\* %0, i64 %28 %29 = load float, float\* %arrayidx.3.i.us, align 4, !tbaa !12 %arrayidx5.3.i.us = getelementptr inbounds float, float\* %1, i64 ... %indvars.iv.next.2.i.us %30 = load float, float\* %arrayidx5.3.i.us, align 4, !tbaa !12 %31 = tail call float @llvm.fmuladd.f32(float %29, float %30, float %27) #2 store float %31, float\* %arrayidx8.i.us, align 4, !tbaa !12, ...!llvm.access.group!16 %indvars.iv.next.3.i.us = add nuw nsw i64 %indvars.iv.next.3.i4.us, 4 %niter.nsub.3.i.us = add i64 %niter.nsub.3.i7.us, -4 %niter.ncmp.3.i.us = icmp eq i64 %niter.nsub.3.i.us, 0 br i1 %niter.ncmp.3.i.us, label %if.end.loopexit.unr-lcssa.i.us.loopexit, ... label %for.bodv.i.us F if.end.loopexit.unr-lcssa.i.us.loopexit: %.lcssa = phi float [ %31, %for.body.i.us ] %indvars.iv.next.3.i.us.lcssa = phi i64 [ %indvars.iv.next.3.i.us, ... %for.body.i.us ] br label %if.end.loopexit.unr-lcssa.i.us if.end.loopexit.unr-lcssa.i.us: %32 = phi float [ %.pre.i1.us15, %for.body.lr.ph.i.us ], [ %.lcssa, ... %if.end.loopexit.unr-lcssa.i.us.loopexit ] %33 = phi i64 [ 0, %for.body.lr.ph.i.us ], [ %indvars.iv.next.3.i.us.lcssa, %if.end.loopexit.unr-lcssa.i.us.loopexit ] br i1 %lcmp.mod.i, label %if.end.i.us, label %for.body.epil.i.us.preheader for.body.epil.i.us.preheader: br label %for.body.epil.i.us for.body.epil.i.us: %epil.iter.sub.i13.us = phi i64 [ %epil.iter.sub.i.us, %for.body.epil.i.us ...], [ %xtraiter.i, %for.body.epil.i.us.preheader ] %indvars.iv.next.epil.i11.us = phi i64 [ %indvars.iv.next.epil.i.us, ... %for.body.epil.i.us ], [ %33, %for.body.epil.i.us.preheader ] %34 = phi float [ %38, %for.body.epil.i.us ], [ %32, ... %for.body.epil.i.us.preheader ] %35 = add nsw i64 %indvars.iv.next.epil.i11.us, %14 %arrayidx.epil.i.us = getelementptr inbounds float, float\* %0, i64 %35 %36 = load float, float\* %arrayidx.epil.i.us, align 4, !tbaa !12 %arrayidx5.epil.i.us = getelementptr inbounds float, float\* %1, i64 ... %indvars.iv.next.epil.i11.us
%37 = load float, float\* %arrayidx5.epil.i.us, align 4, !tbaa !12 %38 = tail call float @llvm.fmuladd.f32(float %36, float %37, float %34) #2 store float %38, float\* %arrayidx8.i.us, align 4, !tbaa !12, ...!llvm.access.group!16 %indvars.iv.next.epil.i.us = add nuw nsw i64 %indvars.iv.next.epil.i11.us, 1 %epil.iter.sub.i.us = add nsw i64 %epil.iter.sub.i13.us, -1 %epil.iter.cmp.i.us = icmp eq i64 %epil.iter.sub.i.us, 0 br i1 %epil.iter.cmp.i.us, label %if.end.i.us.loopexit, label ... %for.body.epil.i.us, !llvm.loop !18 F if.end.i.us.loopexit: br label %if.end.i.us if.end.i.us: %39 = add nuw nsw i64 % local id x.0.us, 1%exitcond = icmp eq i64  $\sqrt{3}$ 9,  $3\overline{2}$ br i1 %exitcond, label %atax kernel1.exit.loopexit, label ... %pregion for entry.entry.i.us, !llvm.loop !20 atax kernel1.exit.loopexit: br label %atax kernel1.exit atax kernel1.exit:

ret void