%9: %10 = shl i64 %6, 5 %cmp217.i = icmp sgt i32 %3, 0 %11 = sext i32 %4 to i64 %wide.trip.count.i = zext i32 %3 to i64 %12 = add nsw i64 %wide.trip.count.i, -1 %xtraiter.i = and i64 %wide.trip.count.i, 3 %13 = icmp ult i64 %12, 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 kernel2.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 [%44, %if.end.i.us], [0, ... \(\frac{\pi}{\pi}\)pre\(\frac{\pi}{\text{on}}\)_for_entry.entry.i.us.preheader] %14 = add nuw nsw i64 % local id x.0.us, %10 %conv.i.us = trunc i64 %14 to i32%cmp.i.us = icmp slt i32 %conv.i.us, %4 br i1 %cmp.i.us, label %for.body.lr.ph.i.us, label %if.end.i.us for.body.lr.ph.i.us: %sext.i.us = shl i64 %14, 32 %idxprom7.i.us = ashr exact i64 %sext.i.us, 32 %arrayidx8.i.us = getelementptr inbounds float, float* %1, i64 %idxprom7.i.us %.pre.i2.us16 = load float, float* %arrayidx8.i.us, align 4, !tbaa !12 br i1 %13, label %if.end.loopexit.unr-lcssa.i.us, label ... %for.body.i.us.preheader F for.body.i.us.preheader: br label %for.body.i.us for.body.i.us: %niter.nsub.3.i8.us = phi i64 [%niter.nsub.3.i.us, %for.body.i.us], [... %unroll iter.i, %for.body.i.us.preheader] %indvars.iv.next.3.i5.us = phi i64 [%indvars.iv.next.3.i.us, %for.body.i.us ...], [0, %for.body.i.us.preheader] %15 = phi float [%35, %for.body.i.us], [%.pre.i2.us16, ... %for.body.i.us.preheader] %16 = mul nsw i64 %indvars.iv.next.3.i5.us, %11 %17 = add nsw i64 %16, %idxprom7.i.us %arrayidx.i.us = getelementptr inbounds float, float* %0, i64 %17 %18 = load float, float* %arrayidx.i.us, align 4, !tbaa !12 %arrayidx5.i.us = getelementptr inbounds float, float* %2, i64 ... %indvars.iv.next.3.i5.us %19 = load float, float* %arrayidx5.i.us, align 4, !tbaa !12 %20 = tail call float @llvm.fmuladd.f32(float %18, float %19, float %15) #2 store float %20, float* %arrayidx8.i.us, align 4, !tbaa !12, ...!llvm.access.group!16 %indvars.iv.next.i.us = or i64 %indvars.iv.next.3.i5.us, 1 %21 = mul nsw i64 %indvars.iv.next.i.us, %11 %22 = add nsw i64 %21, %idxprom7.i.us %arrayidx.1.i.us = getelementptr inbounds float, float* %0, i64 %22 %23 = load float, float* %arrayidx.1.i.us, align 4, !tbaa !12 %arrayidx5.1.i.us = getelementptr inbounds float, float* %2, i64 ... %indvars.iv.next.i.us %24 = load float, float* %arrayidx5.1.i.us, align 4, !tbaa !12 %25 = tail call float @llvm.fmuladd.f32(float %23, float %24, float %20) #2 store float %25, float* %arrayidx8.i.us, align 4, !tbaa !12, ...!llvm.access.group!16 %indvars.iv.next.1.i.us = or i64 %indvars.iv.next.3.i5.us, 2 %26 = mul nsw i64 %indvars.iv.next.1.i.us, %11 %27 = add nsw i64 %26, %idxprom7.i.us %arrayidx.2.i.us = getelementptr inbounds float, float* %0, i64 %27 %28 = load float, float* %arrayidx.2.i.us, align 4, !tbaa !12 %arrayidx5.2.i.us = getelementptr inbounds float, float* %2, i64 ... %indvars.iv.next.1.i.us %29 = load float, float* %arrayidx5.2.i.us, align 4, !tbaa !12 %30 = tail call float @llvm.fmuladd.f32(float %28, float %29, float %25) #2 store float %30, float* %arrayidx8.i.us, align 4, !tbaa !12, ...!llvm.access.group!16 %indvars.iv.next.2.i.us = or i64 %indvars.iv.next.3.i5.us, 3 %31 = mul nsw i64 %indvars.iv.next.2.i.us, %11 %32 = add nsw i64 %31, %idxprom7.i.us %arrayidx.3.i.us = getelementptr inbounds float, float* %0, i64 %32 %33 = load float, float* %arrayidx.3.i.us, align 4, !tbaa !12 %arrayidx5.3.i.us = getelementptr inbounds float, float* %2, i64 ... %indvars.iv.next.2.i.us %34 = load float, float* %arrayidx5.3.i.us, align 4, !tbaa !12 %35 = tail call float @llvm.fmuladd.f32(float %33, float %34, float %30) #2 store float %35, 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.i5.us, 4 %niter.nsub.3.i.us = add i64 %niter.nsub.3.i8.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.body.i.us F if.end.loopexit.unr-lcssa.i.us.loopexit: %.lcssa = phi float [%35, %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: %36 = phi float [%.pre.i2.us16, %for.body.lr.ph.i.us], [%.lcssa, %if.end.loopexit.unr-lcssa.i.us.loopexit] %37 = 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.i14.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.i12.us = phi i64 [%indvars.iv.next.epil.i.us, ... %for.body.epil.i.us], [%37, %for.body.epil.i.us.preheader] %38 = phi float [%43, %for.body.epil.i.us], [%36, ... %for.body.epil.i.us.preheader] %39 = mul nsw i64 %indvars.iv.next.epil.i12.us, %11 %40 = add nsw i64 %39, %idxprom7.i.us %arrayidx.epil.i.us = getelementptr inbounds float, float* %0, i64 %40 %41 = load float, float* %arrayidx.epil.i.us, align 4, !tbaa !12 %arrayidx5.epil.i.us = getelementptr inbounds float, float* %2, i64 ... %indvars.iv.next.epil.i12.us %42 = load float, float* %arrayidx5.epil.i.us, align 4, !tbaa !12 %43 = tail call float @llvm.fmuladd.f32(float %41, float %42, float %38) #2 store float %43, 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.i12.us, 1 %epil.iter.sub.i.us = add nsw i64 %epil.iter.sub.i14.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: %44 = add nuw nsw i64 % local id x.0.us, 1%exitcond = icmp eq i64 $\sqrt[6]{44}$, $3\overline{2}$ br i1 %exitcond, label %atax kernel2.exit.loopexit, label ... %pregion for entry.entry.i.us, !llvm.loop !20 atax kernel2.exit.loopexit: br label %atax kernel2.exit atax kernel2.exit:

ret void