```
%8:
                                                                        %mul.i.i = shl i64 %5, 5
                                                                        %cmp218.i = icmp sgt i32 %3, 0
                                                                        %wide.trip.count.i = zext i32 %3 to i64
                                                                        br i1 %cmp218.i, label %pregion for entry.entry.i.us.preheader, label
                                                                        ... %mvt 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 x.0.us = phi i64 [ %22, %if.end.r exit.i.us.1 ], [ 0,
                                                     ... %pregion for entry.entry.i.us.preheader ]
                                                     \%add1.i.i.us = add nuw nsw i64 % local id x.0.us, %mul.i.i
                                                     %conv.i.us = trunc i64 %add1.i.i.us 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.r exit.i.us
               for.body.lr.ph.i.us:
               %mul.i.us = mul nsw i32 %conv.i.us, %3
               %sext.i.us = shl i64 %add1.i.i.us, 32
               %idxprom7.i.us = ashr exact i64 %sext.i.us, 32
               %arrayidx8.i.us = getelementptr inbounds float, float* %1, i64 %idxprom7.i.us
               \%9 = \text{sext i} 32 \% \text{mul.i.us to i} 64
               %.pre.i1.us4 = load float, float* %arrayidx8.i.us, align 4, !tbaa !12
               br label %for.body.i.us
                for.body.i.us:
                %indvars.iv.next.i3.us = phi i64 [ %indvars.iv.next.i.us, %for.body.i.us ],
                ... [ 0, %for.body.lr.ph.i.us ]
                %10 = phi float [ %14, %for.body.i.us ], [ %.pre.i1.us4,
                ... %for.body.lr.ph.i.us ]
                %11 = add nsw i64 %indvars.iv.next.i3.us, %9
                 %arrayidx.i.us = getelementptr inbounds float, float* %0, i64 %11
                %12 = load float, float* %arrayidx.i.us, align 4, !tbaa !12
                %arrayidx5.i.us = getelementptr inbounds float, float* %2, i64
                ... %indvars.iv.next.i3.us
                %13 = load float, float* %arrayidx5.i.us, align 4, !tbaa !12
                %14 = tail call float @llvm.fmuladd.f32(float %12, float %13, float %10) #2
                store float %14, float* %arrayidx8.i.us, align 4, !tbaa !12,
                ...!llvm.access.group!16
                %indvars.iv.next.i.us = add nuw nsw i64 %indvars.iv.next.i3.us, 1
                %exitcond.not.i.us = icmp eq i64 %indvars.iv.next.i.us, %wide.trip.count.i
                br i1 %exitcond.not.i.us, label %if.end.r exit.i.us.loopexit, label
                ... %for.body.i.us, !llvm.loop !18
                                 if.end.r exit.i.us.loopexit:
                                  br label %if.end.r exit.i.us
                      if.end.r exit.i.us:
                       %15 = \text{ or } i64 \% \text{ local_id_x.0.us, 1}
                       %add1.i.i.us.1 = add nuw nsw i64 %15, %mul.i.i
                       %conv.i.us.1 = trunc i64 %add1.i.i.us.1 to i32
                       %cmp.i.us.1 = icmp slt i32 %conv.i.us.1, %3
                       br i1 %cmp.i.us.1, label %for.body.lr.ph.i.us.1, label %if.end.r exit.i.us.1
   for.body.lr.ph.i.us.1:
    %mul.i.us.1 = mul nsw i32 %conv.i.us.1, %3
    %sext.i.us.1 = shl i64 %add1.i.i.us.1, 32
    %idxprom7.i.us.1 = ashr exact i64 %sext.i.us.1, 32
    %arrayidx8.i.us.1 = getelementptr inbounds float, float* %1, i64
    ... %idxprom7.i.us.1
    %16 = sext i32 %mul.i.us.1 to i64
    %.pre.i1.us4.1 = load float, float* %arrayidx8.i.us.1, align 4, !tbaa !12
    br label %for.body.i.us.1
for.body.i.us.1:
%indvars.iv.next.i3.us.1 = phi i64 [ %indvars.iv.next.i.us.1,
... %for.body.i.us.1 ], [ 0, %for.body.lr.ph.i.us.1 ]
%17 = phi float [ %21, %for.body.i.us.1 ], [ %.pre.i1.us4.1,
... %for.body.lr.ph.i.us.1 ]
%18 = add nsw i64 %indvars.iv.next.i3.us.1, %16
%arrayidx.i.us.1 = getelementptr inbounds float, float* %0, i64 %18
%19 = load float, float* %arrayidx.i.us.1, align 4, !tbaa !12
%arrayidx5.i.us.1 = getelementptr inbounds float, float* %2, i64
... %indvars.iv.next.i3.us.1
%20 = load float, float* %arrayidx5.i.us.1, align 4, !tbaa !12
%21 = tail call float @llvm.fmuladd.f32(float %19, float %20, float %17) #2
store float %21, float* %arrayidx8.i.us.1, align 4, !tbaa !12,
...!llvm.access.group!16
%indvars.iv.next.i.us.1 = add nuw nsw i64 %indvars.iv.next.i3.us.1, 1
%exitcond.not.i.us.1 = icmp eq i64 %indvars.iv.next.i.us.1,
... %wide.trip.count.i
br i1 %exitcond.not.i.us.1, label %if.end.r exit.i.us.1.loopexit, label
... %for.body.i.us.1, !llvm.loop !18
                                                           F
                            if.end.r exit.i.us.1.loopexit:
                            br label %if.end.r exit.i.us.1
                                                     if.end.r exit.i.us.1:
                                                       \%22 = add nuw nsw i64 % local id x.0.us, 2
                                                       %exitcond.not.1 = icmp eq i64 \%2\overline{2}, 32
                                                       br i1 %exitcond.not.1, label %mvt kernel1.exit.loopexit, label
                                                       ... %pregion for entry.entry.i.us, !llvm.loop !20
                                                                          mvt kernel1.exit.loopexit:
                                                                          br label %mvt kernel1.exit
                                                                                         mvt kernel1.exit:
```

ret void