```
%10:
                  %11 = \text{sext i} 32 \% 4 \text{ to i} 64
                   %12 = icmp slt i64 %11, 32
                   %13 = select i1 %12, i64 %11, i64 32
                   %14 = \text{sext i} 32 \% 5 \text{ to i} 64
                   %15 = icmp slt i64 %14, 8
                  %16 = select i1 %15, i64 %14, i64 8
                   %mul.i.i = shl i64 %7, 5
                   %mul3.i.i = shl i64 %8, 3
                  %17 = tail call float @llvm.sqrt.f32(float %3) #3, !llvm.access.group !12
                   %18 = icmp ugt i64 \%13, 1
                  %umax = select i1 %18, i64 %13, i64 1
                   %19 = icmp ugt i64 \%16, 1
                   %umax1 = select i1 %19, i64 %16, i64 1
                   %20 = add nsw i64 %umax, -1
                  %21 = trunc i64 %7 to i32
                   %22 = shl i32 \%21, 5
                   %23 = trunc i64 %8 to i32
                   %24 = \text{mul i} 32 \%23, \%4
                  %25 = \text{shl i} 32 \%24, 3
                   %26 = \text{zext i} 32 \% 25 \text{ to i} 64
                   %27 = \text{zext i} 32 \% 22 \text{ to i} 64
                   %28 = add nuw nsw i64 %26, %27
                   %29 = \text{zext i} 32 \% 4 \text{ to i} 64
                   br label %pregion for entry.pregion for init.i
                  pregion for entry.pregion for init.i:
                   \%_{local_id_y.0} = phi i64 [0, \%10], [\%66, \%pregion_for_end.i]
                   \%30 = \overline{\text{mul}} \text{ i}64 \% \text{local} \text{id} \text{ y}.0, \%29
                   %31 = add i64 \% \bar{2}8, \% \bar{3}0
                   %32 = \text{trunc } i64 \%31 \text{ to } i32
                   %add6.i.i = add i64 %_local_id_y.0, %mul3.i.i, !llvm.access.group !12
                   %conv2.i = trunc i64 %add6.i.i to i32, !llvm.access.group !12
                   %mul.i = mul nsw i32 %conv2.i, %4, !llvm.access.group !12
                   %min.iters.check = icmp ult i64 %umax, 8
                   br i1 %min.iters.check, label %pregion for entry.entry.i.preheader, label
                   ... %vector.scevcheck
                                          Τ
                                                                                              F
                                             vector.scevcheck:
                                              %33 = trunc i64 %20 to i32
                                              %34 = add i32 \%22, \%33
                                              %35 = icmp slt i32 %34, %22
                                              %36 = icmp ugt i64 %20, 4294967295
                                              %37 = \text{ or i } 1 \% 35, \% 36
                                              %38 = trunc i64 %20 to i32
                                              %39 = add i32 %32, %38
                                              %40 = icmp slt i32 %39, %32
                                              %41 = icmp ugt i64 %20, 4294967295
                                              %42 = \text{ or i } 1 \%40, \%41
                                              %43 = \text{ or i } 1 \%37, \%42
                                              br i1 %43, label %pregion for entry.entry.i.preheader, label %vector.ph
                                                                    Τ
                                                                  vector.ph:
                                                                   %n.vec = and i64 %umax, -8
                                                                   %broadcast.splatinsert = insertelement <8 x float> undef, float %17, i32 0
                                                                   %broadcast.splat = shufflevector <8 x float> %broadcast.splatinsert, <8 x
                                                                  ... float> undef, <8 x i32> zeroinitializer
                                                                  br label %vector.body
                                                             vector.body:
                                                              %index = phi i64 [ 0, %vector.ph ], [ %index.next, %vector.body ]
                                                              %44 = add i64 %index, %mul.i.i, !llvm.access.group !12
                                                              %45 = trunc i64 %44 to i32, !llvm.access.group !12
                                                              %46 = shl i64 %44, 32, !llvm.access.group !12
                                                              %47 = ashr exact i64 %46, 32, !llvm.access.group !12
                                                              %48 = getelementptr inbounds float, float* %0, i64 %47, !llvm.access.group
                                                              ...!12
                                                              %49 = bitcast float* %48 to <8 x float>*
                                                              %wide.load = load < 8 \times float >, < 8 \times float > %49, align 4, !tbaa !15,
                                                             ...!llvm.access.group!12
                                                              %50 = add nsw i32 %mul.i, %45, !llvm.access.group !12
                                                              %51 = sext i32 %50 to i64, !llvm.access.group !12
                                                              %52 = getelementptr inbounds float, float* %2, i64 %51, !llvm.access.group
                                                             ... !12
                                                              %53 = bitcast float* %52 to <8 x float>*
                                                              %wide.load6 = load <8 x float>, <8 x float>* %53, align 4, !tbaa !15,
                                                              ...!llvm.access.group!12
                                                              %54 = fsub <8 x float> %wide.load6, %wide.load, !llvm.access.group !12
                                                             \%55 = bitcast float* \%52 to <8 x float>*
                                                             store <8 x float> %54, <8 x float>* %55, align 4, !tbaa !15,
                                                              ...!llvm.access.group!12
                                                              %56 = getelementptr inbounds float, float* %1, i64 %47, !llvm.access.group
                                                             ... !12
                                                              \%57 = bitcast float* \%56 to <8 x float>*
                                                              %wide.load7 = load < 8 \times float >, < 8 \times float > %57, align 4, !tbaa !15,
                                                             ...!llvm.access.group!12
                                                              %58 = fmul <8 x float> %broadcast.splat, %wide.load7, !llvm.access.group !12
                                                              %59 = fdiv <8 x float> %54, %58, !fpmath !19, !llvm.access.group !12
                                                              \%60 = bitcast float* \%52 to <8 x float>*
                                                             store <8 x float> %59, <8 x float>* %60, align 4, !tbaa !15,
                                                             ...!llvm.access.group!12
                                                              %index.next = add i64 %index, 8
                                                              %61 = icmp eq i64 %index.next, %n.vec
                                                              br i1 %61, label %middle.block, label %vector.body, !llvm.loop !20
                                                          middle.block:
                                                           %cmp.n = icmp eq i64 %umax, %n.vec
                                                           br il %cmp.n, label %pregion for end.i, label
                                                           ... %pregion for entry.entry.i.preheader
     pregion for entry.entry.i.preheader:
     % local id x.0.ph = phi i64 [0, %vector.scevcheck], 
     ... %pregion for entry pregion for init.i ], [ %n.vec, %middle.block ]
     br label %pregion for entry.entry.i
pregion for entry.entry.i:
\%_{local\_id\_x.0} = phi i64 [ \%65, \%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, !llvm.access.group !12
%conv.i = trunc i64 %add1.i.i to i32, !llvm.access.group !12
%sext.i = shl i64 %add1.i.i, 32, !llvm.access.group !12
%idxprom.i = ashr exact i64 %sext.i, 32, !llvm.access.group !12
%arravidx.i = getelementptr inbounds float, float* %0, i64 %idxprom.i,
...!llvm.access.group!12
%62 = load float, float* %arrayidx.i, align 4, !tbaa !15, !llvm.access.group
%add.i = add nsw i32 %mul.i, %conv.i, !llvm.access.group !12
%idxprom6.i = sext i32 %add.i to i64, !llvm.access.group !12
%arrayidx7.i = getelementptr inbounds float, float* %2, i64 %idxprom6.i,
...!llvm.access.group!12
%63 = load float, float* %arrayidx7.i, align 4, !tbaa !15,
...!llvm.access.group!12
%sub.i = fsub float %63, %62, !llvm.access.group !12
store float %sub.i, float* %arrayidx7.i, align 4, !tbaa !15,
...!llvm.access.group!12
%arrayidx10.i = getelementptr inbounds float, float* %1, i64 %idxprom.i,
...!llvm.access.group!12
%64 = load float, float* %arrayidx10.i, align 4, !tbaa !15,
...!llvm.access.group!12
%mul11.i = fmul float %17, %64, !llvm.access.group !12
%div.i = fdiv float %sub.i, %mul11.i, !fpmath !19, !llvm.access.group !12
store float %div.i, float* %arrayidx7.i, align 4, !tbaa !15,
...!llvm.access.group!12
\%65 = \text{add nuw } i64\% \text{ local id } x.0, 1
%exitcond.not = icmp eq i6\overline{4} %65, %umax
br i1 %exitcond.not, label %pregion for end.i.loopexit, label
... %pregion for entry.entry.i, !llvm.loop \bar{1}23
                                                                             F
                                     pregion for end.i.loopexit:
                                     br label %pregion for end.i
                                                                            pregion for end.i:
                                                                             \%66 = add nuw i64 \% local id v.0, 1
                                                                             %exitcond2.not = icm\overline{p} eq i\overline{6}4 \overline{\%}66, %umax1
                                                                             br i1 %exitcond2.not, label %reduce kernel.exit, label
                                                                             ... %pregion for entry pregion for init.i, !llvm.loop!24
```

reduce kernel.exit:

CFG for 'pocl kernel reduce kernel' function

ret void