```
%8:
                   \%9 = \text{sext i} 32 \% 3 \text{ to i} 64
                   %10 = icmp slt i64 %9, 256
                   %11 = select i1 %10, i64 %9, i64 256
                   %mul.i.i = shl i64 %5, 8
                   %sub.i = add nsw i32 %3, -1, !llvm.access.group !12
                   %12 = icmp ugt i64 \%11, 1
                   %umax = select i1 %12, i64 %11, i64 1
                   %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:
                                         %ident.check = icmp ne i32 %3, 1
                                         %13 = add nsw i64 %umax, -1
                                         %14 = trunc i64 %5 to i32
                                         %15 = shl i32 %14, 8
                                         %16 = trunc i64 %13 to i32
                                         %17 = add i32 %15, %16
                                         %18 = icmp slt i32 %17, %15
                                         %19 = icmp ugt i64 %13, 4294967295
                                         %20 = \text{ or i } 1 \% 18, \%19
                                         %21 = or i1 %ident.check, %20
                                         br i1 %21, label %pregion for entry.entry.i.preheader, label %vector.ph
                                                                                  vector.ph:
                                                                                   %n.vec = and i64 %umax, -8
                                                                                   br label %vector.body
                                                         vector.body:
                                                          %index = phi i64 [ 0, %vector.ph ], [ %index.next, %vector.body ]
                                                          %22 = add i64 %index, %mul.i.i, !llvm.access.group !12
                                                         %23 = trunc i64 %22 to i32, !llvm.access.group !12
                                                          %24 = mul nsw i32 %23, %3, !llvm.access.group !12
                                                          %25 = add nsw i32 %sub.i, %24, !llvm.access.group !12
                                                          %26 = sext i32 %25 to i64, !llvm.access.group !12
                                                          %27 = getelementptr inbounds float, float* %2, i64 %26, !llvm.access.group
                                                         ... !12
                                                          %28 = bitcast float* %27 to <8 x float>*
                                                          %wide.load = load < 8 \times float >, < 8 \times float > %28, align 4, !tbaa !14,
                                                         ...!llvm.access.group!12
                                                          %29 = getelementptr inbounds float, float* %1, i64 %26, !llvm.access.group
                                                         ...!12
                                                          %30 = bitcast float* %29 to <8 x float>*
                                                          %wide.load1 = load <8 x float>, <8 x float>* \%30, align 4, !tbaa !14,
                                                         ...!llvm.access.group!12
                                                          %31 = fdiv <8 x float> %wide.load, %wide.load1, !fpmath !18,
                                                         ...!llvm.access.group!12
                                                          %32 = bitcast float* %27 to <8 x float>*
                                                          store <8 x float> %31, <8 x float>* %32, align 4, !tbaa !14,
                                                         ...!llvm.access.group!12
                                                          %index.next = add i64 %index, 8
                                                          %33 = icmp eq i64 %index.next, %n.vec
                                                          br i1 %33, label %middle.block, label %vector.body, !llvm.loop !19
                                                     middle.block:
                                                      %cmp.n = icmp eq i64 %umax, %n.vec
                                                      br i1 %cmp.n, label %adi kernel2.exit, label
                                                     ... %pregion for entry.entry.i.preheader
  pregion_for_entry.entry.i.preheader:
   % local id x.0.ph = phi i64 [ 0, %vector.scevcheck ], [ 0, %8 ], [ %n.vec,
   ... %middle.block ]
   br label %pregion for entry.entry.i
pregion for entry.entry.i:
\%_{local\_id\_x.0} = phi i64 [\%36, \%pregion_for_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
%mul.i = mul nsw i32 %conv.i, %3, !llvm.access.group !12 %add.i = add nsw i32 %sub.i, %mul.i, !llvm.access.group !12
%idxprom.i = sext i32 %add.i to i64, !llvm.access.group !12
%arrayidx.i = getelementptr inbounds float, float* %2, i64 %idxprom.i,
...!llvm.access.group!12
%34 = load float, float* %arrayidx.i, align 4, !tbaa !14, !llvm.access.group
... !12
%arrayidx6.i = getelementptr inbounds float, float* %1, i64 %idxprom.i,
...!llvm.access.group!12
%35 = load float, float* %arrayidx6.i, align 4, !tbaa !14,
...!llvm.access.group!12
%div.i = fdiv float %34, %35, !fpmath !18, !llvm.access.group !12
store float %div.i, float* %arrayidx.i, align 4, !tbaa !14,
...!llvm.access.group!12
%36 = add nuw i64\% local id x.0, 1
%exitcond.not = icmp eq i64 %36, %umax
br i1 %exitcond.not, label %adi kernel2.exit.loopexit, label
... %pregion for entry.entry.i, !llvm.loop!22
                                                         F
                            adi kernel2.exit.loopexit:
                            br label %adi kernel2.exit
                                           adi kernel2.exit:
                                           ret void
```

CFG for 'pocl kernel adi kernel2' function