```
%7:
                                                                                         %mul.i.i = shl i64 %4, 8
                                                                                         %sub.i = add nsw i32 %2, -1
                                                                                         br label %pregion for entry.entry.i
                                                                            pregion for entry.entry.i:
                                                                            % local id x.0 = phi i64 [ 0, %7 ], [ %15, %if.end.r exit.i.3 ]
                                                                            %add1.i.i = add nuw nsw i64 % local id x.0, %mul.i.i
                                                                            %conv.i = trunc i64 %add1.i.i to i32
                                                                            %cmp.i = icmp sgt i32 %conv.i, 0
                                                                            %cmp2.i = icmp sgt i32 %sub.i, %conv.i
                                                                            %or.cond.i = and i1 %cmp.i, %cmp2.i
                                                                            br i1 %or.cond.i, label %if.then.i, label %if.end.r exit.i
                                    if.then.i:
                                     %sext.i = shl i64 %add1.i.i, 32
                                     %idxprom.i = ashr exact i64 %sext.i, 32
                                     %arrayidx.i = getelementptr inbounds float, float* %1, i64 %idxprom.i
                                     %8 = load float, float* %arrayidx.i, align 4, !tbaa !12
                                     %arrayidx5.i = getelementptr inbounds float, float* %0, i64 %idxprom.i
                                     store float %8, float* %arrayidx5.i, align 4, !tbaa !12, !llvm.access.group
                                     ... !16
                                     br label %if.end.r exit.i
                                               if.end.r exit.i:
                                               \%9 = \text{or } i64 \% \text{ local id } x.0, 1
                                               %add1.i.i.1 = add nuw nsw i64 %9, %mul.i.i
                                               %conv.i.1 = trunc i64 %add1.i.i.1 to i32
                                               %cmp.i.1 = icmp sqt i32 %conv.i.1, 0
                                               %cmp2.i.1 = icmp sqt i32 %sub.i, %conv.i.1
                                               %or.cond.i.1 = and i1 %cmp.i.1, %cmp2.i.1
                                               br i1 %or.cond.i.1, label %if.then.i.1, label %if.end.r exit.i.1
                  if.then.i.1:
                   %sext.i.1 = shl i64 %add1.i.i.1, 32
                   %idxprom.i.1 = ashr exact i64 %sext.i.1, 32
                   %arrayidx.i.1 = getelementptr inbounds float, float* %1, i64 %idxprom.i.1
                   %10 = load float, float* %arrayidx.i.1, align 4, !tbaa !12
                   %arrayidx5.i.1 = getelementptr inbounds float, float* %0, i64 %idxprom.i.1
                   store float %10, float* %arrayidx5.i.1, align 4, !tbaa !12,
                  ...!llvm.access.group!16
                   br label %if.end.r exit.i.1
                             if.end.r exit.i.1:
                              %11 = \text{ or } i64 \% \text{ local id } x.0, 2
                              %add1.i.i.2 = add nuw nsw i64 %11, %mul.i.i
                              %conv.i.2 = trunc i64 %add1.i.i.2 to i32
                              %cmp.i.2 = icmp sqt i32 %conv.i.2, 0
                              %cmp2.i.2 = icmp sqt i32 %sub.i, %conv.i.2
                              %or.cond.i.2 = and i1 %cmp.i.2, %cmp2.i.2
                              br i1 %or.cond.i.2, label %if.then.i.2, label %if.end.r exit.i.2
if.then.i.2:
%sext.i.2 = shl i64 %add1.i.i.2, 32
%idxprom.i.2 = ashr exact i64 %sext.i.2, 32
%arrayidx.i.2 = getelementptr inbounds float, float* %1, i64 %idxprom.i.2
%12 = load float, float* %arrayidx.i.2, align 4, !tbaa !12
%arrayidx5.i.2 = getelementptr inbounds float, float* %0, i64 %idxprom.i.2
store float %12, float* %arrayidx5.i.2, align 4, !tbaa !12,
...!llvm.access.group!16
br label %if.end.r exit.i.2
                                                if.end.r exit.i.2:
                                                %13 = \text{ or } i64 \% \text{ local id } x.0, 3
                                                %add1.i.i.3 = add nuw nsw i64 %13, %mul.i.i
                                                %conv.i.3 = trunc i64 %add1.i.i.3 to i32
                                                %cmp.i.3 = icmp sgt i32 %conv.i.3, 0
                                                %cmp2.i.3 = icmp sgt i32 %sub.i, %conv.i.3
                                                %or.cond.i.3 = and i1 %cmp.i.3, %cmp2.i.3
                                                br i1 %or.cond.i.3, label %if.then.i.3, label %if.end.r exit.i.3
                   if.then.i.3:
                   %sext.i.3 = shl i64 %add1.i.i.3, 32
                   %idxprom.i.3 = ashr exact i64 %sext.i.3, 32
                    %arrayidx.i.3 = getelementptr inbounds float, float* %1, i64 %idxprom.i.3
                   %14 = load float, float* %arrayidx.i.3, align 4, !tbaa !12
                   %arrayidx5.i.3 = getelementptr inbounds float, float* %0, i64 %idxprom.i.3
                   store float %14, float* %arrayidx5.i.3, align 4, !tbaa !12,
                   ...!llvm.access.group!16
                   br label %if.end.r exit.i.3
                                                                     if.end.r exit.i.3:
                                                                     %15 = add nuw nsw i64 % local id x.0, 4
                                                                     %exitcond.not.3 = icmp eq i64 \%15, 256
                                                                     br i1 %exitcond.not.3, label %runJacobi1D kernel2.exit, label
                                                                     ... %pregion for entry entry i, !llvm.loop !1\overline{8}
                                                                                                                     F
                                                                       runJacobi1D kernel2.exit:
                                                                       ret void
```

CFG for 'pocl kernel runJacobi1D kernel2' function