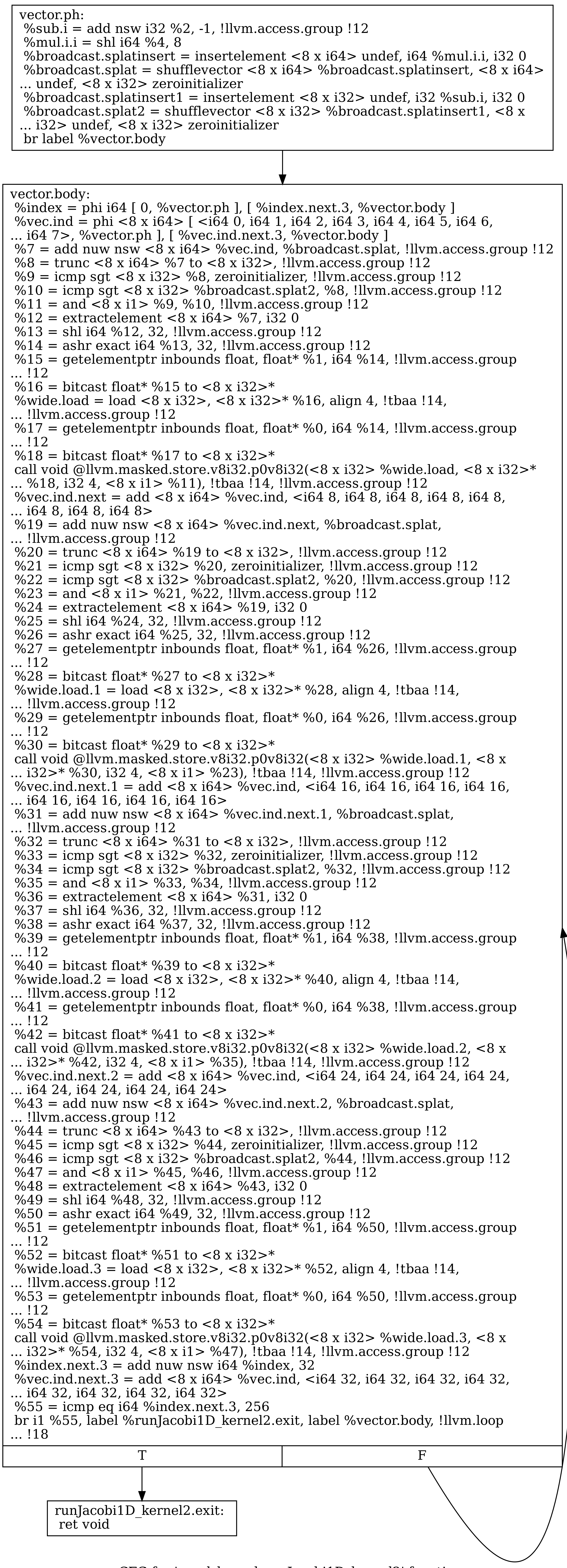

```
vector.ph:
 %sub.i = add nsw i32 %2, -1, !llvm.access.group !12
 %mul.i.i = shl i64 %4, 8
 %broadcast.splatinsert = insertelement <8 x i64> undef, i64 %mul.i.i, i32 0
 %broadcast.splat = shufflevector <8 x i64> %broadcast.splatinsert, <8 x i64>
... undef, <8 x i32> zeroinitializer
 %broadcast.splatinsert1 = insertelement <8 x i32> undef, i32 %sub.i, i32 0
 %broadcast.splat2 = shufflevector <8 x i32> %broadcast.splatinsert1, <8 x
... i32> undef, <8 x i32> zeroinitializer
 br label %vector.body
```

```
vector.body:
 %index = phi i64 [ 0, %vector.ph ], [ %index.next.3, %vector.body ]
 %vec.ind = phi <8 x i64> [ <i64 0, i64 1, i64 2, i64 3, i64 4, i64 5, i64 6,
... i64 7>, %vector.ph ], [ %vec.ind.next.3, %vector.body ]
 %7 = add nuw nsw <8 x i64> %vec.ind, %broadcast.splat, !llvm.access.group !12
 %8 = trunc <8 x i64> %7 to <8 x i32>, !llvm.access.group !12
 %9 = icmp sgt <8 x i32> %8, zeroinitializer, !llvm.access.group !12
 %10 = icmp sgt <8 x i32> %broadcast.splat2, %8, !llvm.access.group !12
 %11 = and <8 x i1> %9, %10, !llvm.access.group !12
 %12 = extractelement <8 x i64> %7, i32 0
 %13 = shl i64 %12, 32, !llvm.access.group !12
 %14 = ashr exact i64 %13, 32, !llvm.access.group !12
 %15 = getelementptr inbounds float, float* %1, i64 %14, !llvm.access.group
... !12
 %16 = bitcast float* %15 to <8 x i32>*
 %wide.load = load <8 x i32>, <8 x i32>* %16, align 4, !tbaa !14,
... !llvm.access.group !12
 %17 = getelementptr inbounds float, float* %0, i64 %14, !llvm.access.group
... !12
 %18 = bitcast float* %17 to <8 x i32>*
 call void @llvm.masked.store.v8i32.p0v8i32(<8 x i32> %wide.load, <8 x i32>*
... %18, i32 4, <8 x i1> %11), !tbaa !14, !llvm.access.group !12
 %vec.ind.next = add <8 x i64> %vec.ind, <i64 8, i64 8, i64 8, i64 8, i64 8,
... i64 8, i64 8, i64 8>
 %19 = add nuw nsw <8 x i64> %vec.ind.next, %broadcast.splat,
... !llvm.access.group !12
 %20 = trunc <8 x i64> %19 to <8 x i32>, !llvm.access.group !12
 %21 = icmp sgt <8 x i32> %20, zeroinitializer, !llvm.access.group !12
 %22 = icmp sgt <8 x i32> %broadcast.splat2, %20, !llvm.access.group !12
 %23 = and <8 x i1> %21, %22, !llvm.access.group !12
 %24 = extractelement <8 x i64> %19, i32 0
 %25 = shl i64 %24, 32, !llvm.access.group !12
 %26 = ashr exact i64 %25, 32, !llvm.access.group !12
 %27 = getelementptr inbounds float, float* %1, i64 %26, !llvm.access.group
... !12
 %28 = bitcast float* %27 to <8 x i32>*
 %wide.load.1 = load <8 x i32>, <8 x i32>* %28, align 4, !tbaa !14,
... !llvm.access.group !12
 %29 = getelementptr inbounds float, float* %0, i64 %26, !llvm.access.group
... !12
 %30 = bitcast float* %29 to <8 x i32>*
 call void @llvm.masked.store.v8i32.p0v8i32(<8 x i32> %wide.load.1, <8 x
... i32>* %30, i32 4, <8 x i1> %23), !tbaa !14, !llvm.access.group !12
 %vec.ind.next.1 = add <8 x i64> %vec.ind, <i64 16, i64 16, i64 16, i64 16,
... i64 16, i64 16, i64 16, i64 16>
 %31 = add nuw nsw <8 x i64> %vec.ind.next.1, %broadcast.splat,
... !llvm.access.group !12
 %32 = trunc <8 x i64> %31 to <8 x i32>, !llvm.access.group !12
 %33 = icmp sgt <8 x i32> %32, zeroinitializer, !llvm.access.group !12
 %34 = icmp sgt <8 x i32> %broadcast.splat2, %32, !llvm.access.group !12
 %35 = and <8 x i1> %33, %34, !llvm.access.group !12
 %36 = extractelement <8 x i64> %31, i32 0
 %37 = shl i64 %36, 32, !llvm.access.group !12
 %38 = ashr exact i64 %37, 32, !llvm.access.group !12
 %39 = getelementptr inbounds float, float* %1, i64 %38, !llvm.access.group
... !12
 %40 = bitcast float* %39 to <8 x i32>*
 %wide.load.2 = load <8 x i32>, <8 x i32>* %40, align 4, !tbaa !14,
... !llvm.access.group !12
 %41 = getelementptr inbounds float, float* %0, i64 %38, !llvm.access.group
... !12
 %42 = bitcast float* %41 to <8 x i32>*
 call void @llvm.masked.store.v8i32.p0v8i32(<8 x i32> %wide.load.2, <8 x
... i32>* %42, i32 4, <8 x i1> %35), !tbaa !14, !llvm.access.group !12
 %vec.ind.next.2 = add <8 x i64> %vec.ind, <i64 24, i64 24, i64 24, i64 24,
... i64 24, i64 24, i64 24, i64 24>
 %43 = add nuw nsw <8 x i64> %vec.ind.next.2, %broadcast.splat,
... !llvm.access.group !12
 %44 = trunc <8 x i64> %43 to <8 x i32>, !llvm.access.group !12
 %45 = icmp sgt <8 x i32> %44, zeroinitializer, !llvm.access.group !12
 %46 = icmp sgt <8 x i32> %broadcast.splat2, %44, !llvm.access.group !12
 %47 = and <8 x i1> %45, %46, !llvm.access.group !12
 %48 = extractelement <8 x i64> %43, i32 0
 %49 = shl i64 %48, 32, !llvm.access.group !12
 %50 = ashr exact i64 %49, 32, !llvm.access.group !12
 %51 = getelementptr inbounds float, float* %1, i64 %50, !llvm.access.group
... !12
 %52 = bitcast float* %51 to <8 x i32>*
 %wide.load.3 = load <8 x i32>, <8 x i32>* %52, align 4, !tbaa !14,
... !llvm.access.group !12
 %53 = getelementptr inbounds float, float* %0, i64 %50, !llvm.access.group
... !12
 %54 = bitcast float* %53 to <8 x i32>*
 call void @llvm.masked.store.v8i32.p0v8i32(<8 x i32> %wide.load.3, <8 x
... i32>* %54, i32 4, <8 x i1> %47), !tbaa !14, !llvm.access.group !12
 %index.next.3 = add nuw nsw i64 %index, 32
 %vec.ind.next.3 = add <8 x i64> %vec.ind, <i64 32, i64 32, i64 32, i64 32,
... i64 32, i64 32, i64 32, i64 32>
 %55 = icmp eq i64 %index.next.3, 256
 br i1 %55, label %runJacobi1D_kernel2.exit, label %vector.body, !llvm.loop
... !18
```

```
runJacobi1D_kernel2.exit:
 ret void
```

CFG for '_pocl_kernel_runJacobi1D_kernel2' function