```
%8:
 %mul.i.i = shl i64 %5, 8
 %cmp458.i = icmp sgt i32 %3, 0
 %9 = sext i32 %2 to i64
 %wide.trip.count.i = zext i32 %3 to i64
 br label %pregion_for_entry.entry.i
```

```
pregion_for_entry.entry.i:
 %_local_id_x.0 = phi i64 [ 0, %8 ], [ %24, %for.end30.r_exit.i ]
 %add1.i.i = add nuw nsw 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, %2, !llvm.access.group !12
 %add.i = add nsw i32 %mul.i, %conv.i, !llvm.access.group !12
 %idxprom.i = sext i32 %add.i to i64, !llvm.access.group !12
 %arrayidx.i = getelementptr inbounds float, float* %0, i64 %idxprom.i,
... !llvm.access.group !12
 store float 1.000000e+00, float* %arrayidx.i, align 4, !tbaa !14,
... !llvm.access.group !12
 %j2.060.i = add i32 %conv.i, 1, !llvm.access.group !12
 %cmp61.i = icmp slt i32 %j2.060.i, %2, !llvm.access.group !12
 br i1 %cmp61.i, label %for.cond3.preheader.lr.ph.i, label
... %for.end30.r_exit.i, !llvm.access.group !12
```

```
for.cond3.preheader.lr.ph.i:
 %sext.i = shl i64 %add1.i.i, 32, !llvm.access.group !12
 %10 = ashr exact i64 %sext.i, 32, !llvm.access.group !12
 %11 = sext i32 %j2.060.i to i64, !llvm.access.group !12
 %12 = sext i32 %mul.i to i64, !llvm.access.group !12
 br label %for.cond3.preheader.i, !llvm.access.group !12
```

```
for.cond3.preheader.i:
 %indvars.iv.next67.i5 = phi i64 [ %indvars.iv.next67.i, %for.end.i ], [ %11,
... %for.cond3.preheader.lr.ph.i ]
 %13 = add nsw i64 %indvars.iv.next67.i5, %12, !llvm.access.group !12
 %arrayidx19.i = getelementptr inbounds float, float* %0, i64 %13,
... !llvm.access.group !12
 %.pre.i = load float, float* %arrayidx19.i, align 4, !tbaa !14,
... !llvm.access.group !12
 br i1 %cmp458.i, label %for.body6.i.preheader, label %for.end.i,
... !llvm.access.group !12
```

```
for.body6.i.preheader:
 br label %for.body6.i
```

```
for.body6.i:
 %indvars.iv.next.i3 = phi i64 [ %indvars.iv.next.i, %for.body6.i ], [ 0,
... %for.body6.i.preheader ]
 %14 = phi float [ %20, %for.body6.i ], [ %.pre.i, %for.body6.i.preheader ]
 %15 = mul nsw i64 %indvars.iv.next.i3, %9, !llvm.access.group !12
 %16 = add nsw i64 %15, %10, !llvm.access.group !12
 %arrayidx10.i = getelementptr inbounds float, float* %1, i64 %16,
... !llvm.access.group !12
 %17 = load float, float* %arrayidx10.i, align 4, !tbaa !14,
... !llvm.access.group !12
 %18 = add nsw i64 %15, %indvars.iv.next67.i5, !llvm.access.group !12
 %arrayidx14.i = getelementptr inbounds float, float* %1, i64 %18,
... !llvm.access.group !12
 %19 = load float, float* %arrayidx14.i, align 4, !tbaa !14,
... !llvm.access.group !12
 %20 = tail call float @llvm.fmuladd.f32(float %17, float %19, float %14) #3,
... !llvm.access.group !12
 store float %20, float* %arrayidx19.i, align 4, !tbaa !14,
... !llvm.access.group !12
 %indvars.iv.next.i = add nuw nsw i64 %indvars.iv.next.i3, 1,
... !llvm.access.group !12
 %exitcond.not.i = icmp eq i64 %indvars.iv.next.i, %wide.trip.count.i,
... !llvm.access.group !12
 br i1 %exitcond.not.i, label %for.end.i.loopexit, label %for.body6.i,
... !llvm.loop !18, !llvm.access.group !12
```

```
for.end.i.loopexit:
 %.lcssa = phi float [ %20, %for.body6.i ]
 br label %for.end.i
```

```
for.end.i:
 %21 = phi float [ %.pre.i, %for.cond3.preheader.i ], [ %.lcssa,
... %for.end.i.loopexit ]
 %22 = mul nsw i64 %indvars.iv.next67.i5, %9, !llvm.access.group !12
 %23 = add nsw i64 %22, %10, !llvm.access.group !12
 %arrayidx27.i = getelementptr inbounds float, float* %0, i64 %23,
... !llvm.access.group !12
 store float %21, float* %arrayidx27.i, align 4, !tbaa !14,
... !llvm.access.group !12
 %indvars.iv.next67.i = add nsw i64 %indvars.iv.next67.i5, 1,
... !llvm.access.group !12
 %lftr.wideiv.i = trunc i64 %indvars.iv.next67.i to i32, !llvm.access.group
... !12
 %exitcond72.not.i = icmp eq i32 %lftr.wideiv.i, %2, !llvm.access.group !12
 br i1 %exitcond72.not.i, label %for.end30.r_exit.i.loopexit, label
... %for.cond3.preheader.i, !llvm.loop !20, !llvm.access.group !12
```

```
for.end30.r_exit.i.loopexit:
 br label %for.end30.r_exit.i
```

```
for.end30.r_exit.i:
 %24 = add nuw nsw i64 %_local_id_x.0, 1
 %exitcond.not = icmp eq i64 %24, 256
 br i1 %exitcond.not, label %corr_kernel.exit, label
... %pregion_for_entry.entry.i, !llvm.loop !21
```

```
corr_kernel.exit:
 ret void
```

CFG for '_pocl_kernel_corr_kernel' function