```
%11:
 %mul.i.i = shl i64 %8, 8
 %cmp259.i = icmp sgt i32 %6, 0
 %12 = sext i32 %5 to i64
 %wide.trip.count.i = zext i32 %6 to i64
 br i1 %cmp259.i, label %pregion_for_entry.entry.i.us.preheader, label
... %pregion_for_entry.entry.i.preheader
```

```
pregion_for_entry.entry.i.us.preheader:
 br label %pregion_for_entry.entry.i.us
```

```
pregion_for_entry.entry.i.preheader:
 %div.i = fdiv float 0.000000e+00, %3
 %13 = tail call float @llvm.sqrt.f32(float %div.i) #2
 %cmp27.i = fcmp ugt float %13, %4
 %storemerge.i = select i1 %cmp27.i, float %13, float 1.000000e+00
 %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.splatinsert12 = insertelement <8 x i32> undef, i32 %5, i32 0
 %broadcast.splat13 = shufflevector <8 x i32> %broadcast.splatinsert12, <8 x
... i32> undef, <8 x i32> zeroinitializer
 %broadcast.splatinsert14 = insertelement <8 x i32> undef, i32 %5, i32 0
 %broadcast.splat15 = shufflevector <8 x i32> %broadcast.splatinsert14, <8 x
... i32> undef, <8 x i32> zeroinitializer
 %broadcast.splatinsert16 = insertelement <8 x float> undef, float
... %storemerge.i, i32 0
 %broadcast.splat17 = shufflevector <8 x float> %broadcast.splatinsert16, <8
... x float> undef, <8 x i32> zeroinitializer
 %broadcast.splatinsert18 = insertelement <8 x float> undef, float
... %storemerge.i, i32 0
 %broadcast.splat19 = shufflevector <8 x float> %broadcast.splatinsert18, <8
... x float> undef, <8 x i32> zeroinitializer
 %14 = or <8 x i64> %broadcast.splat, <i64 0, i64 1, i64 2, i64 3, i64 4, i64
... 5, i64 6, i64 7>
 %15 = trunc <8 x i64> %14 to <8 x i32>, !llvm.access.group !12
 %16 = trunc i64 %mul.i.i to i32
 %17 = or i32 %16, 8
 %18 = insertelement <8 x i32> undef, i32 %17, i64 0
 %19 = shufflevector <8 x i32> %18, <8 x i32> undef, <8 x i32> zeroinitializer
 %20 = or <8 x i32> %19, <i32 0, i32 1, i32 2, i32 3, i32 4, i32 5, i32 6,
... i32 7>
 %21 = icmp sgt <8 x i32> %broadcast.splat13, %15, !llvm.access.group !12
 %22 = icmp sgt <8 x i32> %broadcast.splat15, %20, !llvm.access.group !12
 %23 = extractelement <8 x i64> %14, i32 0
 %24 = shl i64 %23, 32, !llvm.access.group !12
 %25 = ashr exact i64 %24, 32, !llvm.access.group !12
 %26 = getelementptr inbounds float, float* %1, i64 %25, !llvm.access.group
... !12
 %27 = bitcast float* %26 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %27, i32 4, <8 x i1> %21), !tbaa !14, !llvm.access.group !12
 %28 = getelementptr inbounds float, float* %26, i64 8
 %29 = bitcast float* %28 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %29, i32 4, <8 x i1> %22), !tbaa !14, !llvm.access.group !12
 %30 = or <8 x i64> %broadcast.splat, <i64 16, i64 17, i64 18, i64 19, i64
... 20, i64 21, i64 22, i64 23>
 %31 = trunc <8 x i64> %30 to <8 x i32>, !llvm.access.group !12
 %32 = trunc i64 %mul.i.i to i32
 %33 = or i32 %32, 8
 %34 = insertelement <8 x i32> undef, i32 %33, i64 0
 %35 = shufflevector <8 x i32> %34, <8 x i32> undef, <8 x i32> zeroinitializer
 %36 = or <8 x i32> %35, <i32 16, i32 17, i32 18, i32 19, i32 20, i32 21, i32
... 22, i32 23>
 %37 = icmp sgt <8 x i32> %broadcast.splat13, %31, !llvm.access.group !12
 %38 = icmp sgt <8 x i32> %broadcast.splat15, %36, !llvm.access.group !12
 %39 = extractelement <8 x i64> %30, i32 0
 %40 = shl i64 %39, 32, !llvm.access.group !12
 %41 = ashr exact i64 %40, 32, !llvm.access.group !12
 %42 = getelementptr inbounds float, float* %1, i64 %41, !llvm.access.group
... !12
 %43 = bitcast float* %42 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %43, i32 4, <8 x i1> %37), !tbaa !14, !llvm.access.group !12
 %44 = getelementptr inbounds float, float* %42, i64 8
 %45 = bitcast float* %44 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %45, i32 4, <8 x i1> %38), !tbaa !14, !llvm.access.group !12
 %46 = or <8 x i64> %broadcast.splat, <i64 32, i64 33, i64 34, i64 35, i64
... 36, i64 37, i64 38, i64 39>
 %47 = trunc <8 x i64> %46 to <8 x i32>, !llvm.access.group !12
 %48 = trunc i64 %mul.i.i to i32
 %49 = or i32 %48, 8
 %50 = insertelement <8 x i32> undef, i32 %49, i64 0
 %51 = shufflevector <8 x i32> %50, <8 x i32> undef, <8 x i32> zeroinitializer
 %52 = or <8 x i32> %51, <i32 32, i32 33, i32 34, i32 35, i32 36, i32 37, i32
... 38, i32 39>
 %53 = icmp sgt <8 x i32> %broadcast.splat13, %47, !llvm.access.group !12
 %54 = icmp sgt <8 x i32> %broadcast.splat15, %52, !llvm.access.group !12
 %55 = extractelement <8 x i64> %46, i32 0
 %56 = shl i64 %55, 32, !llvm.access.group !12
 %57 = ashr exact i64 %56, 32, !llvm.access.group !12
 %58 = getelementptr inbounds float, float* %1, i64 %57, !llvm.access.group
... !12
 %59 = bitcast float* %58 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %59, i32 4, <8 x i1> %53), !tbaa !14, !llvm.access.group !12
 %60 = getelementptr inbounds float, float* %58, i64 8
 %61 = bitcast float* %60 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %61, i32 4, <8 x i1> %54), !tbaa !14, !llvm.access.group !12
 %62 = or <8 x i64> %broadcast.splat, <i64 48, i64 49, i64 50, i64 51, i64
... 52, i64 53, i64 54, i64 55>
 %63 = trunc <8 x i64> %62 to <8 x i32>, !llvm.access.group !12
 %64 = trunc i64 %mul.i.i to i32
 %65 = or i32 %64, 8
 %66 = insertelement <8 x i32> undef, i32 %65, i64 0
 %67 = shufflevector <8 x i32> %66, <8 x i32> undef, <8 x i32> zeroinitializer
 %68 = or <8 x i32> %67, <i32 48, i32 49, i32 50, i32 51, i32 52, i32 53, i32
... 54, i32 55>
 %69 = icmp sgt <8 x i32> %broadcast.splat13, %63, !llvm.access.group !12
 %70 = icmp sgt <8 x i32> %broadcast.splat15, %68, !llvm.access.group !12
 %71 = extractelement <8 x i64> %62, i32 0
 %72 = shl i64 %71, 32, !llvm.access.group !12
 %73 = ashr exact i64 %72, 32, !llvm.access.group !12
 %74 = getelementptr inbounds float, float* %1, i64 %73, !llvm.access.group
... !12
 %75 = bitcast float* %74 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %75, i32 4, <8 x i1> %69), !tbaa !14, !llvm.access.group !12
 %76 = getelementptr inbounds float, float* %74, i64 8
 %77 = bitcast float* %76 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %77, i32 4, <8 x i1> %70), !tbaa !14, !llvm.access.group !12
 %78 = or <8 x i64> %broadcast.splat, <i64 64, i64 65, i64 66, i64 67, i64
... 68, i64 69, i64 70, i64 71>
 %79 = trunc <8 x i64> %78 to <8 x i32>, !llvm.access.group !12
 %80 = trunc i64 %mul.i.i to i32
 %81 = or i32 %80, 8
 %82 = insertelement <8 x i32> undef, i32 %81, i64 0
 %83 = shufflevector <8 x i32> %82, <8 x i32> undef, <8 x i32> zeroinitializer
 %84 = or <8 x i32> %83, <i32 64, i32 65, i32 66, i32 67, i32 68, i32 69, i32
... 70, i32 71>
 %85 = icmp sgt <8 x i32> %broadcast.splat13, %79, !llvm.access.group !12
 %86 = icmp sgt <8 x i32> %broadcast.splat15, %84, !llvm.access.group !12
 %87 = extractelement <8 x i64> %78, i32 0
 %88 = shl i64 %87, 32, !llvm.access.group !12
 %89 = ashr exact i64 %88, 32, !llvm.access.group !12
 %90 = getelementptr inbounds float, float* %1, i64 %89, !llvm.access.group
... !12
 %91 = bitcast float* %90 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %91, i32 4, <8 x i1> %85), !tbaa !14, !llvm.access.group !12
 %92 = getelementptr inbounds float, float* %90, i64 8
 %93 = bitcast float* %92 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %93, i32 4, <8 x i1> %86), !tbaa !14, !llvm.access.group !12
 %94 = or <8 x i64> %broadcast.splat, <i64 80, i64 81, i64 82, i64 83, i64
... 84, i64 85, i64 86, i64 87>
 %95 = trunc <8 x i64> %94 to <8 x i32>, !llvm.access.group !12
 %96 = trunc i64 %mul.i.i to i32
 %97 = or i32 %96, 8
 %98 = insertelement <8 x i32> undef, i32 %97, i64 0
 %99 = shufflevector <8 x i32> %98, <8 x i32> undef, <8 x i32> zeroinitializer
 %100 = or <8 x i32> %99, <i32 80, i32 81, i32 82, i32 83, i32 84, i32 85,
... i32 86, i32 87>
 %101 = icmp sgt <8 x i32> %broadcast.splat13, %95, !llvm.access.group !12
 %102 = icmp sgt <8 x i32> %broadcast.splat15, %100, !llvm.access.group !12
 %103 = extractelement <8 x i64> %94, i32 0
 %104 = shl i64 %103, 32, !llvm.access.group !12
 %105 = ashr exact i64 %104, 32, !llvm.access.group !12
 %106 = getelementptr inbounds float, float* %1, i64 %105, !llvm.access.group
... !12
 %107 = bitcast float* %106 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %107, i32 4, <8 x i1> %101), !tbaa !14, !llvm.access.group !12
 %108 = getelementptr inbounds float, float* %106, i64 8
 %109 = bitcast float* %108 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %109, i32 4, <8 x i1> %102), !tbaa !14, !llvm.access.group !12
 %110 = or <8 x i64> %broadcast.splat, <i64 96, i64 97, i64 98, i64 99, i64
... 100, i64 101, i64 102, i64 103>
 %111 = trunc <8 x i64> %110 to <8 x i32>, !llvm.access.group !12
 %112 = trunc i64 %mul.i.i to i32
 %113 = or i32 %112, 8
 %114 = insertelement <8 x i32> undef, i32 %113, i64 0
 %115 = shufflevector <8 x i32> %114, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %116 = or <8 x i32> %115, <i32 96, i32 97, i32 98, i32 99, i32 100, i32 101,
... i32 102, i32 103>
 %117 = icmp sgt <8 x i32> %broadcast.splat13, %111, !llvm.access.group !12
 %118 = icmp sgt <8 x i32> %broadcast.splat15, %116, !llvm.access.group !12
 %119 = extractelement <8 x i64> %110, i32 0
 %120 = shl i64 %119, 32, !llvm.access.group !12
 %121 = ashr exact i64 %120, 32, !llvm.access.group !12
 %122 = getelementptr inbounds float, float* %1, i64 %121, !llvm.access.group
... !12
 %123 = bitcast float* %122 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %123, i32 4, <8 x i1> %117), !tbaa !14, !llvm.access.group !12
 %124 = getelementptr inbounds float, float* %122, i64 8
 %125 = bitcast float* %124 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %125, i32 4, <8 x i1> %118), !tbaa !14, !llvm.access.group !12
 %126 = or <8 x i64> %broadcast.splat, <i64 112, i64 113, i64 114, i64 115,
... i64 116, i64 117, i64 118, i64 119>
 %127 = trunc <8 x i64> %126 to <8 x i32>, !llvm.access.group !12
 %128 = trunc i64 %mul.i.i to i32
 %129 = or i32 %128, 8
 %130 = insertelement <8 x i32> undef, i32 %129, i64 0
 %131 = shufflevector <8 x i32> %130, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %132 = or <8 x i32> %131, <i32 112, i32 113, i32 114, i32 115, i32 116, i32
... 117, i32 118, i32 119>
 %133 = icmp sgt <8 x i32> %broadcast.splat13, %127, !llvm.access.group !12
 %134 = icmp sgt <8 x i32> %broadcast.splat15, %132, !llvm.access.group !12
 %135 = extractelement <8 x i64> %126, i32 0
 %136 = shl i64 %135, 32, !llvm.access.group !12
 %137 = ashr exact i64 %136, 32, !llvm.access.group !12
 %138 = getelementptr inbounds float, float* %1, i64 %137, !llvm.access.group
... !12
 %139 = bitcast float* %138 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %139, i32 4, <8 x i1> %133), !tbaa !14, !llvm.access.group !12
 %140 = getelementptr inbounds float, float* %138, i64 8
 %141 = bitcast float* %140 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %141, i32 4, <8 x i1> %134), !tbaa !14, !llvm.access.group !12
 %142 = or <8 x i64> %broadcast.splat, <i64 128, i64 129, i64 130, i64 131,
... i64 132, i64 133, i64 134, i64 135>
 %143 = trunc <8 x i64> %142 to <8 x i32>, !llvm.access.group !12
 %144 = trunc i64 %mul.i.i to i32
 %145 = or i32 %144, 8
 %146 = insertelement <8 x i32> undef, i32 %145, i64 0
 %147 = shufflevector <8 x i32> %146, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %148 = or <8 x i32> %147, <i32 128, i32 129, i32 130, i32 131, i32 132, i32
... 133, i32 134, i32 135>
 %149 = icmp sgt <8 x i32> %broadcast.splat13, %143, !llvm.access.group !12
 %150 = icmp sgt <8 x i32> %broadcast.splat15, %148, !llvm.access.group !12
 %151 = extractelement <8 x i64> %142, i32 0
 %152 = shl i64 %151, 32, !llvm.access.group !12
 %153 = ashr exact i64 %152, 32, !llvm.access.group !12
 %154 = getelementptr inbounds float, float* %1, i64 %153, !llvm.access.group
... !12
 %155 = bitcast float* %154 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %155, i32 4, <8 x i1> %149), !tbaa !14, !llvm.access.group !12
 %156 = getelementptr inbounds float, float* %154, i64 8
 %157 = bitcast float* %156 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %157, i32 4, <8 x i1> %150), !tbaa !14, !llvm.access.group !12
 %158 = or <8 x i64> %broadcast.splat, <i64 144, i64 145, i64 146, i64 147,
... i64 148, i64 149, i64 150, i64 151>
 %159 = trunc <8 x i64> %158 to <8 x i32>, !llvm.access.group !12
 %160 = trunc i64 %mul.i.i to i32
 %161 = or i32 %160, 8
 %162 = insertelement <8 x i32> undef, i32 %161, i64 0
 %163 = shufflevector <8 x i32> %162, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %164 = or <8 x i32> %163, <i32 144, i32 145, i32 146, i32 147, i32 148, i32
... 149, i32 150, i32 151>
 %165 = icmp sgt <8 x i32> %broadcast.splat13, %159, !llvm.access.group !12
 %166 = icmp sgt <8 x i32> %broadcast.splat15, %164, !llvm.access.group !12
 %167 = extractelement <8 x i64> %158, i32 0
 %168 = shl i64 %167, 32, !llvm.access.group !12
 %169 = ashr exact i64 %168, 32, !llvm.access.group !12
 %170 = getelementptr inbounds float, float* %1, i64 %169, !llvm.access.group
... !12
 %171 = bitcast float* %170 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %171, i32 4, <8 x i1> %165), !tbaa !14, !llvm.access.group !12
 %172 = getelementptr inbounds float, float* %170, i64 8
 %173 = bitcast float* %172 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %173, i32 4, <8 x i1> %166), !tbaa !14, !llvm.access.group !12
 %174 = or <8 x i64> %broadcast.splat, <i64 160, i64 161, i64 162, i64 163,
... i64 164, i64 165, i64 166, i64 167>
 %175 = trunc <8 x i64> %174 to <8 x i32>, !llvm.access.group !12
 %176 = trunc i64 %mul.i.i to i32
 %177 = or i32 %176, 8
 %178 = insertelement <8 x i32> undef, i32 %177, i64 0
 %179 = shufflevector <8 x i32> %178, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %180 = or <8 x i32> %179, <i32 160, i32 161, i32 162, i32 163, i32 164, i32
... 165, i32 166, i32 167>
 %181 = icmp sgt <8 x i32> %broadcast.splat13, %175, !llvm.access.group !12
 %182 = icmp sgt <8 x i32> %broadcast.splat15, %180, !llvm.access.group !12
 %183 = extractelement <8 x i64> %174, i32 0
 %184 = shl i64 %183, 32, !llvm.access.group !12
 %185 = ashr exact i64 %184, 32, !llvm.access.group !12
 %186 = getelementptr inbounds float, float* %1, i64 %185, !llvm.access.group
... !12
 %187 = bitcast float* %186 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %187, i32 4, <8 x i1> %181), !tbaa !14, !llvm.access.group !12
 %188 = getelementptr inbounds float, float* %186, i64 8
 %189 = bitcast float* %188 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %189, i32 4, <8 x i1> %182), !tbaa !14, !llvm.access.group !12
 %190 = or <8 x i64> %broadcast.splat, <i64 176, i64 177, i64 178, i64 179,
... i64 180, i64 181, i64 182, i64 183>
 %191 = trunc <8 x i64> %190 to <8 x i32>, !llvm.access.group !12
 %192 = trunc i64 %mul.i.i to i32
 %193 = or i32 %192, 8
 %194 = insertelement <8 x i32> undef, i32 %193, i64 0
 %195 = shufflevector <8 x i32> %194, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %196 = or <8 x i32> %195, <i32 176, i32 177, i32 178, i32 179, i32 180, i32
... 181, i32 182, i32 183>
 %197 = icmp sgt <8 x i32> %broadcast.splat13, %191, !llvm.access.group !12
 %198 = icmp sgt <8 x i32> %broadcast.splat15, %196, !llvm.access.group !12
 %199 = extractelement <8 x i64> %190, i32 0
 %200 = shl i64 %199, 32, !llvm.access.group !12
 %201 = ashr exact i64 %200, 32, !llvm.access.group !12
 %202 = getelementptr inbounds float, float* %1, i64 %201, !llvm.access.group
... !12
 %203 = bitcast float* %202 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %203, i32 4, <8 x i1> %197), !tbaa !14, !llvm.access.group !12
 %204 = getelementptr inbounds float, float* %202, i64 8
 %205 = bitcast float* %204 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %205, i32 4, <8 x i1> %198), !tbaa !14, !llvm.access.group !12
 %206 = or <8 x i64> %broadcast.splat, <i64 192, i64 193, i64 194, i64 195,
... i64 196, i64 197, i64 198, i64 199>
 %207 = trunc <8 x i64> %206 to <8 x i32>, !llvm.access.group !12
 %208 = trunc i64 %mul.i.i to i32
 %209 = or i32 %208, 8
 %210 = insertelement <8 x i32> undef, i32 %209, i64 0
 %211 = shufflevector <8 x i32> %210, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %212 = or <8 x i32> %211, <i32 192, i32 193, i32 194, i32 195, i32 196, i32
... 197, i32 198, i32 199>
 %213 = icmp sgt <8 x i32> %broadcast.splat13, %207, !llvm.access.group !12
 %214 = icmp sgt <8 x i32> %broadcast.splat15, %212, !llvm.access.group !12
 %215 = extractelement <8 x i64> %206, i32 0
 %216 = shl i64 %215, 32, !llvm.access.group !12
 %217 = ashr exact i64 %216, 32, !llvm.access.group !12
 %218 = getelementptr inbounds float, float* %1, i64 %217, !llvm.access.group
... !12
 %219 = bitcast float* %218 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %219, i32 4, <8 x i1> %213), !tbaa !14, !llvm.access.group !12
 %220 = getelementptr inbounds float, float* %218, i64 8
 %221 = bitcast float* %220 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %221, i32 4, <8 x i1> %214), !tbaa !14, !llvm.access.group !12
 %222 = or <8 x i64> %broadcast.splat, <i64 208, i64 209, i64 210, i64 211,
... i64 212, i64 213, i64 214, i64 215>
 %223 = trunc <8 x i64> %222 to <8 x i32>, !llvm.access.group !12
 %224 = trunc i64 %mul.i.i to i32
 %225 = or i32 %224, 8
 %226 = insertelement <8 x i32> undef, i32 %225, i64 0
 %227 = shufflevector <8 x i32> %226, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %228 = or <8 x i32> %227, <i32 208, i32 209, i32 210, i32 211, i32 212, i32
... 213, i32 214, i32 215>
 %229 = icmp sgt <8 x i32> %broadcast.splat13, %223, !llvm.access.group !12
 %230 = icmp sgt <8 x i32> %broadcast.splat15, %228, !llvm.access.group !12
 %231 = extractelement <8 x i64> %222, i32 0
 %232 = shl i64 %231, 32, !llvm.access.group !12
 %233 = ashr exact i64 %232, 32, !llvm.access.group !12
 %234 = getelementptr inbounds float, float* %1, i64 %233, !llvm.access.group
... !12
 %235 = bitcast float* %234 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %235, i32 4, <8 x i1> %229), !tbaa !14, !llvm.access.group !12
 %236 = getelementptr inbounds float, float* %234, i64 8
 %237 = bitcast float* %236 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %237, i32 4, <8 x i1> %230), !tbaa !14, !llvm.access.group !12
 %238 = or <8 x i64> %broadcast.splat, <i64 224, i64 225, i64 226, i64 227,
... i64 228, i64 229, i64 230, i64 231>
 %239 = trunc <8 x i64> %238 to <8 x i32>, !llvm.access.group !12
 %240 = trunc i64 %mul.i.i to i32
 %241 = or i32 %240, 8
 %242 = insertelement <8 x i32> undef, i32 %241, i64 0
 %243 = shufflevector <8 x i32> %242, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %244 = or <8 x i32> %243, <i32 224, i32 225, i32 226, i32 227, i32 228, i32
... 229, i32 230, i32 231>
 %245 = icmp sgt <8 x i32> %broadcast.splat13, %239, !llvm.access.group !12
 %246 = icmp sgt <8 x i32> %broadcast.splat15, %244, !llvm.access.group !12
 %247 = extractelement <8 x i64> %238, i32 0
 %248 = shl i64 %247, 32, !llvm.access.group !12
 %249 = ashr exact i64 %248, 32, !llvm.access.group !12
 %250 = getelementptr inbounds float, float* %1, i64 %249, !llvm.access.group
... !12
 %251 = bitcast float* %250 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %251, i32 4, <8 x i1> %245), !tbaa !14, !llvm.access.group !12
 %252 = getelementptr inbounds float, float* %250, i64 8
 %253 = bitcast float* %252 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %253, i32 4, <8 x i1> %246), !tbaa !14, !llvm.access.group !12
 %254 = or <8 x i64> %broadcast.splat, <i64 240, i64 241, i64 242, i64 243,
... i64 244, i64 245, i64 246, i64 247>
 %255 = trunc <8 x i64> %254 to <8 x i32>, !llvm.access.group !12
 %256 = trunc i64 %mul.i.i to i32
 %257 = or i32 %256, 8
 %258 = insertelement <8 x i32> undef, i32 %257, i64 0
 %259 = shufflevector <8 x i32> %258, <8 x i32> undef, <8 x i32>
... zeroinitializer
 %260 = or <8 x i32> %259, <i32 240, i32 241, i32 242, i32 243, i32 244, i32
... 245, i32 246, i32 247>
 %261 = icmp sgt <8 x i32> %broadcast.splat13, %255, !llvm.access.group !12
 %262 = icmp sgt <8 x i32> %broadcast.splat15, %260, !llvm.access.group !12
 %263 = extractelement <8 x i64> %254, i32 0
 %264 = shl i64 %263, 32, !llvm.access.group !12
 %265 = ashr exact i64 %264, 32, !llvm.access.group !12
 %266 = getelementptr inbounds float, float* %1, i64 %265, !llvm.access.group
... !12
 %267 = bitcast float* %266 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat17,
... <8 x float>* %267, i32 4, <8 x i1> %261), !tbaa !14, !llvm.access.group !12
 %268 = getelementptr inbounds float, float* %266, i64 8
 %269 = bitcast float* %268 to <8 x float>*
 call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat19,
... <8 x float>* %269, i32 4, <8 x i1> %262), !tbaa !14, !llvm.access.group !12
 br label %std_kernel.exit
```

```
pregion_for_entry.entry.i.us:
 %_local_id_x.0.us = phi i64 [ %277, %if.end32.r_exit.i.us ], [ 0,
... %pregion_for_entry.entry.i.us.preheader ]
 %add1.i.i.us = add nuw nsw i64 %_local_id_x.0.us, %mul.i.i,
... !llvm.access.group !12
 %conv.i.us = trunc i64 %add1.i.i.us to i32, !llvm.access.group !12
 %cmp.i.us = icmp slt i32 %conv.i.us, %5, !llvm.access.group !12
 br i1 %cmp.i.us, label %if.then.i.us, label %if.end32.r_exit.i.us,
... !llvm.access.group !12
```

```
if.then.i.us:
 %sext.i.us = shl i64 %add1.i.i.us, 32, !llvm.access.group !12
 %idxprom.i.us = ashr exact i64 %sext.i.us, 32, !llvm.access.group !12
 %arrayidx.i.us = getelementptr inbounds float, float* %1, i64 %idxprom.i.us,
... !llvm.access.group !12
 store float 0.000000e+00, float* %arrayidx.i.us, align 4, !tbaa !14,
... !llvm.access.group !12
 %arrayidx7.i.us = getelementptr inbounds float, float* %0, i64
... %idxprom.i.us, !llvm.access.group !12
 br label %for.body.i.us, !llvm.access.group !12
```

```
for.body.i.us:
 %indvars.iv.next.i4.us = phi i64 [ %indvars.iv.next.i.us, %for.body.i.us ],
... [ 0, %if.then.i.us ]
 %270 = phi float [ %275, %for.body.i.us ], [ 0.000000e+00, %if.then.i.us ]
 %271 = mul nsw i64 %indvars.iv.next.i4.us, %12, !llvm.access.group !12
 %272 = add nsw i64 %271, %idxprom.i.us, !llvm.access.group !12
 %arrayidx5.i.us = getelementptr inbounds float, float* %2, i64 %272,
... !llvm.access.group !12
 %273 = load float, float* %arrayidx5.i.us, align 4, !tbaa !14,
... !llvm.access.group !12
 %274 = load float, float* %arrayidx7.i.us, align 4, !tbaa !14,
... !llvm.access.group !12
 %sub.i.us = fsub float %273, %274, !llvm.access.group !12
 %275 = tail call float @llvm.fmuladd.f32(float %sub.i.us, float %sub.i.us,
... float %270) #2, !llvm.access.group !12
 store float %275, float* %arrayidx.i.us, align 4, !tbaa !14,
... !llvm.access.group !12
 %indvars.iv.next.i.us = add nuw nsw i64 %indvars.iv.next.i4.us, 1,
... !llvm.access.group !12
 %exitcond.not.i.us = icmp eq i64 %indvars.iv.next.i.us, %wide.trip.count.i,
... !llvm.access.group !12
 br i1 %exitcond.not.i.us, label %for.end.loopexit.i.us, label
... %for.body.i.us, !llvm.loop !18, !llvm.access.group !12
```

```
for.end.loopexit.i.us:
 %.lcssa = phi float [ %275, %for.body.i.us ]
 %div.i.us = fdiv float %.lcssa, %3, !fpmath !20, !llvm.access.group !12
 %276 = tail call float @llvm.sqrt.f32(float %div.i.us) #2,
... !llvm.access.group !12
 %cmp27.i.us = fcmp ugt float %276, %4, !llvm.access.group !12
 %storemerge.i.us = select i1 %cmp27.i.us, float %276, float 1.000000e+00,
... !llvm.access.group !12
 store float %storemerge.i.us, float* %arrayidx.i.us, align 4, !tbaa !14,
... !llvm.access.group !12
 br label %if.end32.r_exit.i.us, !llvm.access.group !12
```

```
if.end32.r_exit.i.us:
 %277 = add nuw nsw i64 %_local_id_x.0.us, 1
 %exitcond.not = icmp eq i64 %277, 256
 br i1 %exitcond.not, label %std_kernel.exit.loopexit, label
... %pregion_for_entry.entry.i.us, !llvm.loop !21
```

```
std_kernel.exit.loopexit:
 br label %std_kernel.exit
```

```
std_kernel.exit:
 ret void
```

CFG for '_pocl_kernel_std_kernel' function