```
%12 = shl i64 %8, 6
                                                                                            %13 = \text{shl } i64 \%9, 5
                                                                                            %mul6.i = mul i32 %6, %1
                                                                                            %cmp970.i = icmp sgt i32 %2, 0
                                                                                            %14 = zext i32 %2 to i64
                                                                                            %15 = add nsw i64 \%14, -1
                                                                                            %xtraiter.i = and i64 %14, 3
                                                                                            %16 = icmp ult i64 %15, 3
                                                                                            %unroll iter.i = sub nuw nsw i64 %14, %xtraiter.i
                                                                                             %lcmp.mod.i = icmp eq i64 %xtraiter.i, 0
                                                                                            br label %pregion for entry.pregion for init.i
                                                                                     pregion for entry.pregion for init.i:
                                                                                     \c^{1}%_local_id_y.0 = phi i64 [0, \c^{1}11], [%55, %pregion_for_end.i]
                                                                                     \%\overline{17} = \overline{a}d\overline{d} nuw nsw i64 % local id y.0, %13
                                                                                      %conv2.i = trunc i64 %17 to i32
                                                                                      %cmp4.i = icmp slt i32 %conv2.i, %1
                                                                                     %reass.add.i = add i32 %mul6.i, %conv2.i
                                                                                      %reass.mul.i = mul i32 %reass.add.i, %2
                                                                                      %18 = sext i32 %reass.mul.i to i64
                                                                                     br label %pregion for entry.entry.i
                                                             pregion for entry.entry.i:
                                                             % [ocal\ id\ x.0] = phi\ i64\ [0, \%pregion\ for\ entry.pregion\ for\ init.i], |
                                                             ... \( \bar{856}, \bar{8if.end.i} \)
                                                             %19 = add nuw nsw i64 % local id x.0, %12
                                                             %conv.i = trunc i64 %19 to i32
                                                             %cmp.i = icmp slt i32 %conv.i, %2
                                                             %or.cond.i = and i1 %cmp4.i, %cmp.i
                                                              br i1 %or.cond.i, label %if.then.i, label %if.end.i
                                                                                                                   F
                                 if.then.i:
                                 %add8.i = add nsw i32 %reass.mul.i, %conv.i
                                 %idxprom.i = sext i32 %add8.i to i64
                                 %arrayidx.i = getelementptr inbounds float, float* %5, i64 %idxprom.i
                                 store float 0.000000e+00, float* %arrayidx.i, align 4, !tbaa !12,
                                 ...!llvm.access.group!16
                                 br i1 %cmp970.i, label %for.body.preheader.i, label %if.end.i
                                                                                        F
for.body.preheader.i:
 %sext.i = shl i64 %19, 32
 %20 = ashr exact i64 %sext.i, 32
 br i1 %16, label %if.end.loopexit.unr-lcssa.i, label %for.body.i.preheader
                  Τ
                                                         F
                                               for.bodv.i.preheader:
                                                br label %for.body.i
                 for.body.i:
                  %niter.nsub.3.i7 = phi i64 [ %niter.nsub.3.i, %for.body.i ], [
                  ... %unroll iter.i, %for.body.i.preheader ]
                  %indvars.iv.next.3.i4 = phi i64 [ %indvars.iv.next.3.i, %for.body.i ], [ 0,
                  ... %for.body.i.preheader ]
                  %21 = phi float [ %45, %for.body.i ], [ 0.000000e+00, %for.body.i.preheader ]
                  %22 = add nsw i64 %indvars.iv.next.3.i4, %18
                  %arrayidx24.i = getelementptr inbounds float, float* %3, i64 %22
                  %23 = load float, float* %arrayidx24.i, align 4, !tbaa !12
                  %24 = mul nuw nsw i64 %indvars.iv.next.3.i4, %14
                  %25 = add nsw i64 %24, %20
                  %arrayidx28.i = getelementptr inbounds float, float* %4, i64 %25
                  %26 = load float, float* %arrayidx28.i, align 4, !tbaa !12
                  %27 = tail call float @llvm.fmuladd.f32(float %23, float %26, float %21) #2
                  store float %27, float* %arrayidx.i, align 4, !tbaa !12, !llvm.access.group
                  ... !16
                  %indvars.iv.next.i = or i64 %indvars.iv.next.3.i4, 1
                  %28 = add nsw i64 %indvars.iv.next.i, %18
                  %arrayidx24.1.i = getelementptr inbounds float, float* %3, i64 %28
                  %29 = load float, float* %arrayidx24.1.i, align 4, !tbaa !12
                  %30 = mul nuw nsw i64 %indvars.iv.next.i, %14
                  %31 = add nsw i64 %30, %20
                  %arrayidx28.1.i = getelementptr inbounds float, float* %4, i64 %31
                  %32 = load float, float* %arrayidx28.1.i, align 4, !tbaa !12
                  %33 = tail call float @llvm.fmuladd.f32(float %29, float %32, float %27) #2
                  store float %33, float* %arrayidx.i, align 4, !tbaa !12, !llvm.access.group
                  ... !16
                  %indvars.iv.next.1.i = or i64 %indvars.iv.next.3.i4, 2
                  %34 = add nsw i64 %indvars.iv.next.1.i, %18
                  %arrayidx24.2.i = getelementptr inbounds float, float* %3, i64 %34
                  %35 = load float, float* %arrayidx24.2.i, align 4, !tbaa !12
                  %36 = mul nuw nsw i64 %indvars.iv.next.1.i, %14
                  %37 = add nsw i64 %36, %20
                  %arrayidx28.2.i = getelementptr inbounds float, float* %4, i64 %37
                  %38 = load float, float* %arrayidx28.2.i, align 4, !tbaa !12
                  %39 = tail call float @llvm.fmuladd.f32(float %35, float %38, float %33) #2
                  store float %39, float* %arrayidx.i, align 4, !tbaa !12, !llvm.access.group
                  ... !16
                  %indvars.iv.next.2.i = or i64 %indvars.iv.next.3.i4, 3
                  %40 = add nsw i64 %indvars.iv.next.2.i, %18
                  %arrayidx24.3.i = getelementptr inbounds float, float* %3, i64 %40
                  %41 = load float, float* %arrayidx24.3.i, align 4, !tbaa !12
                  %42 = mul nuw nsw i64 %indvars.iv.next.2.i, %14
                  %43 = add nsw i64 %42, %20
                  %arrayidx28.3.i = getelementptr inbounds float, float* %4, i64 %43
                  %44 = load float, float* %arrayidx28.3.i, align 4, !tbaa !12
                  %45 = tail call float @llvm.fmuladd.f32(float %41, float %44, float %39) #2
                  store float %45, float* %arravidx.i, align 4, !tbaa !12, !llvm.access.group
                  ... !16
                  %indvars.iv.next.3.i = add nuw nsw i64 %indvars.iv.next.3.i4, 4
                  %niter.nsub.3.i = add i64 %niter.nsub.3.i7, -4
                  %niter.ncmp.3.i = icmp eq i64 %niter.nsub.3.i, 0
                  br i1 %niter.ncmp.3.i, label %if.end.loopexit.unr-lcssa.i.loopexit, label
                  ... %for.bodv.i
         if.end.loopexit.unr-lcssa.i.loopexit:
         %.lcssa = phi float [ %45, %for.body.i ]
         %indvars.iv.next.3.i.lcssa = phi i64 [ %indvars.iv.next.3.i, %for.body.i ]
         br label %if.end.loopexit.unr-lcssa.i
if.end.loopexit.unr-lcssa.i:
\%46 = phi float [0.000000e+00, \%for.body.preheader.i], [\%.lcssa,
... %if.end.loopexit.unr-lcssa.i.loopexit ]
%47 = phi i64 [ 0, %for.body.preheader.i ], [ %indvars.iv.next.3.i.lcssa,
... %if.end.loopexit.unr-lcssa.i.loopexit ]
br i1 %lcmp.mod.i, label %if.end.i, label %for.body.epil.i.preheader
                                            for.body.epil.i.preheader:
                                             br label %for.body.epil.i
                   for.body.epil.i:
                    %epil.iter.sub.i13 = phi i64 [ %epil.iter.sub.i, %for.body.epil.i ], [
                     . %xtraiter.i, %for.body.epil.i.preheader ]
                   %indvars.iv.next.epil.i11 = phi i64 [ %indvars.iv.next.epil.i, ... %for.body.epil.i ], [ %47, %for.body.epil.i.preheader ]
                    %48 = phi float [ %54, %for.body.epil.i ], [ %46, %for.body.epil.i.preheader
                    %49 = add nsw i64 %indvars.iv.next.epil.i11, %18
                    %arrayidx24.epil.i = getelementptr inbounds float, float* %3, i64 %49
                    %50 = load float, float* %arrayidx24.epil.i, align 4, !tbaa !12
                    %51 = mul nuw nsw i64 %indvars.iv.next.epil.i11, %14
                    %52 = add nsw i64 %51, %20
                    %arrayidx28.epil.i = getelementptr inbounds float, float* %4, i64 %52
                    %53 = load float, float* %arrayidx28.epil.i, align 4, !tbaa !12
                    %54 = tail call float @llvm.fmuladd.f32(float %50, float %53, float %48) #2
                    store float %54, float* %arrayidx.i, align 4, !tbaa !12, !llvm.access.group
                    ... !16
                    %indvars.iv.next.epil.i = add nuw nsw i64 %indvars.iv.next.epil.i11, 1
                    %epil.iter.sub.i = add nsw i64 %epil.iter.sub.i13, -1
                    %epil.iter.cmp.i = icmp eq i64 %epil.iter.sub.i, 0
                    br il %epil.iter.cmp.i, label %if.end.i.loopexit, label %for.body.epil.i,
                    ...!llvm.loop!19
                                                                               F
                                                        if.end.i.loopexit:
                                                         br label %if.end.i
                                                      if.end.i:
                                                      \%56 = \text{add nuw nsw } i64 \% \text{ local id } x.0, 1
                                                      %exitcond = icmp eq i64 \sqrt{5}6, 6\overline{4}
                                                      br i1 %exitcond, label %pregion for end.i, label %pregion for entry.entry.i,
                                                      ...!llvm.loop!23
                                                                                                                 F
                                                                      pregion for end.i:
                                                                      0.055 = add nuw nsw i64 % local id y.0, 1
                                                                       \%exitcond14 = icmp eq i6\overline{4} %55, 3\overline{2}
                                                                      br i1 %exitcond14, label %doitgen_kernel1.exit, label ... %pregion_for_entry.pregion_for_init.i, !llvm.loop !21
                                                                         doitgen kernel1.exit:
```

ret void

CFG for 'pocl kernel doitgen kernel1' function

%11: