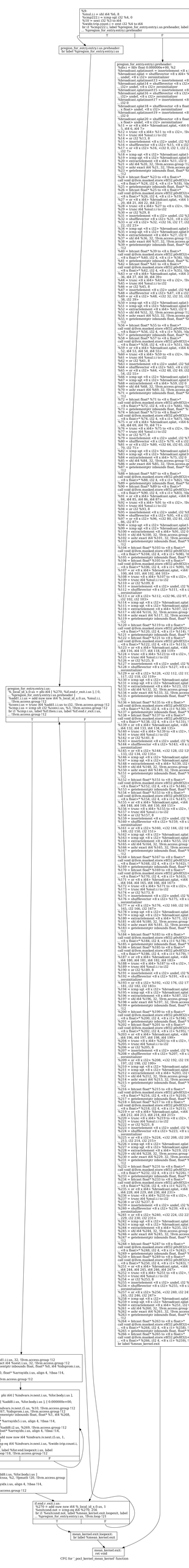
%9:
%mul.i.i = shl i64 %6, 8
%add22.i.i = sub nsw i32 %4, 0
%cmp.i.i = icmp sgt i32 %4, 0
%wide.trip.count.i = zext i32 %4 to i64
br i1 %cmp22.i.i, label %pregion_for_entry.entry.i.ua.preheader, label %pregion_for_entry.entry.i.preheader
T
F
pregion_for_entry.entry.i.ua.preheader:
br label %pregion_for_entry.entry.i.ua
pregion_for_entry.entry.i.preheader:
%dx.i = fdiv float 0.000000e+00, %2
%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 %3, i32 0
%broadcast.splat14 = shufflevector <8 x i32> %broadcast.splatinsert1, <8 x i32> undef, <8 x i32> zeroinitializer
%broadcast.splatinsert15 = insertelement <8 x i32> undef, i32 %3, i32 0
%broadcast.splat16 = shufflevector <8 x i32> %broadcast.splatinsert15, <8 x i32> undef, <8 x i32> zeroinitializer
%broadcast.splatinsert17 = insertelement <8 x float> undef, float %dx.i, i32 0
%broadcast.splat18 = shufflevector <8 x float> %broadcast.splatinsert17, <8 x float> undef, <8 x i32> zeroinitializer
%broadcast.splatinsert19 = insertelement <8 x float> undef, float %dx.i, i32 0
%broadcast.splat20 = shufflevector <8 x float> %broadcast.splatinsert19, <8 x float> undef, <8 x i32> zeroinitializer
%11 = or <8 x i64> %broadcast.splat, <64 0, i64 1, i64 2, i64 3, i64 4, i64 5, i64 6, i64 7>
%12 = trunc <8 x i64> %11 to <8 x i32>, !llvm.access.group.112
%13 = trunc i64 %mul.i.i to i32
%14 = or i32 %13, 8
%15 = insertelement <8 x i32> undef, i32 %14, i64 0
%16 = shufflevector <8 x i32> %15, <8 x i32> undef, <8 x i32> zeroinitializer
%17 = <8 x i32> <i32 %16, i32 0, i32 1, i32 2, i32 3, i32 4, i32 5, i32 6, ...32 7>
%18 = icmp sgt <8 x i32> %broadcast.splat14, %12, !llvm.access.group.112
%19 = sext <8 x i32> %broadcast.splat16 to <8 x i64>, !llvm.access.group.112
%20 = extractelement <8 x i64> %11, i32 0
%21 = shl i64 %20, 32, !llvm.access.group.112
%22 = ashr exact i64 %21, 32, !llvm.access.group.112
%23 = getelementptr inbounds float, float* %0, i64 %22, !llvm.access.group.112
%24 = bitcast float* %23 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>* %24, i32 4, <8 x i1> %18), !llvm.access.group.112
%25 = getelementptr inbounds float, float* %23, i64 8
%26 = bitcast float*** %25 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>* %26, i32 4, <8 x i1> %19), !llvm.access.group.112
%27 = add nsw i64 %mul.i.i, <64 0, i64 1, i64 2, i64 3, i64 4, i64 5, i64 6, i64 7>, !llvm.access.group.112
%28 = trunc <8 x i64> %27 to <8 x i32>, !llvm.access.group.112
%29 = or i32 %28, 8
%30 = insertelement <8 x i32> undef, i32 %30, i64 0
%31 = shufflevector <8 x i32> %31, <8 x i32> undef, <8 x i32> zeroinitializer
%32 = or <8 x i32> %32, <32 16, i32 17, i32 18, i32 19, i32 20, i32 21, i32 22, i32 23>
%33 = icmp sgt <8 x i32> %broadcast.splat14, %28, !llvm.access.group.112
%35 = icmp sgt <8 x i32> %broadcast.splat16, %33, !llvm.access.group.112
%36 = sext <8 x i32> %32 to <8 x i64>
%37 = shl i64 %36, 32, !llvm.access.group.112
%38 = ashr exact i64 %37, 32, !llvm.access.group.112
%39 = getelementptr inbounds float, float* %0, i64 %38, !llvm.access.group.112
%40 = bitcast float* %39 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>* %40, i32 4, <8 x i1> %34), !llvm.access.group.112
%41 = getelementptr inbounds float, float* %39, i64 8
%42 = bitcast float*41 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>* %42, i32 4, <8 x i1> %35), !llvm.114, !llvm.access.group.112
%43 = or <8 x i64> %broadcast.splat, <64 32, i64 33, i64 34, i64 35, i64 36, i64 37, i64 38, i64 39>
%44 = trunc <8 x i64> %43 to <8 x i32>, !llvm.access.group.112
%45 = trunc i64 %mul.i.i to i32
%46 = or i32 %45, 8
%47 = insertelement <8 x i32> undef, i32 %46, i64 0
%48 = shufflevector <8 x i32> %47, <8 x i32> undef, <8 x i32> zeroinitializer
%49 = or <8 x i32> %48, <i32 32, i32 33, i32 34, i32 35, i32 36, i32 37, i32 38, i32 39>
%50 = icmp sgt <8 x i32> %broadcast.splat14, %44, !llvm.access.group.112
%52 = icmp sgt <8 x i32> %broadcast.splat16, %49, !llvm.access.group.112
%53 = sext <8 x i32> %52 to <8 x i64>
%54 = ashr exact i64 %53, 32, !llvm.access.group.112
%55 = getelementptr inbounds float, float* %0, i64 %54, !llvm.access.group.112
%56 = bitcast float* %55 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>* %56, i32 4, <8 x i1> %50), !llvm.114, !llvm.access.group.112
%57 = getelementptr inbounds float, float* %55, i64 8
%58 = bitcast float*57 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>* %58, i32 4, <8 x i1> %53), !llvm.114, !llvm.access.group.112
%59 = or <8 x i64> %broadcast.splat, <64 48, i64 49, i64 50, i64 51, i64 52, i64 53, i64 54, i64 55>
%60 = trunc <8 x i64> %59 to <8 x i32>, !llvm.access.group.112
%61 = trunc i64 %mul.i.i to i32
%62 = or i32 %61, 8
%63 = insertelement <8 x i32> undef, i32 %62, i64 0
%64 = shufflevector <8 x i32> %63, <8 x i32> undef, <8 x i32> zeroinitializer
%65 = or <8 x i32> %64, <i32 48, i32 49, i32 50, i32 51, i32 52, i32 53, i32 54, i32 55>
%66 = icmp sgt <8 x i32> %broadcast.splat14, %60, !llvm.access.group.112
%68 = icmp sgt <8 x i32> %broadcast.splat16, %65, !llvm.access.group.112
%69 = sext <8 x i64> %68, 32, !llvm.access.group.112
%70 = ashr exact i64 %69, 32, !llvm.access.group.112
%71 = getelementptr inbounds float, float* %0, i64 %70, !llvm.access.group.112
%72 = bitcast float*71 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>* %72, i32 4, <8 x i1> %66), !llvm.114, !llvm.access.group.112
%73 = getelementptr inbounds float, float* %71, i64 8
%74 = bitcast float*73 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>* %74, i32 4, <8 x i1> %67), !llvm.114, !llvm.access.group.112
%75 = or <8 x i64> %broadcast.splat, <64 64, i64 65, i64 66, i64 67, i64 68, i64 69, i64 70, i64 71>
%76 = trunc <8 x i64> %75 to <8 x i32>, !llvm.access.group.112
%77 = trunc i64 %mul.i.i to i32
%78 = or i32 %77, 8
%79 = insertelement <8 x i32> undef, i32 %78, i64 0
%80 = shufflevector <8 x i32> %79, <8 x i32> undef, <8 x i32> zeroinitializer
%81 = or <8 x i32> %80, <i32 64, i32 65, i32 66, i32 67, i32 68, i32 69, i32 70, i32 71>
%82 = icmp sgt <8 x i32> %broadcast.splat14, %76, !llvm.access.group.112
%83 = icmp sgt <8 x i32> %broadcast.splat16, %81, !llvm.access.group.112
%84 = sext <8 x i64> %83 to <8 x i64>
%85 = ashr exact i64 %84, 32, !llvm.access.group.112
%86 = getelementptr inbounds float, float* %0, i64 %85, !llvm.access.group.112
%87 = getelementptr inbounds float, float* %0, i64 %86, !llvm.access.group.112
%88 = bitcast float*87 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>* %88, i32 4, <8 x i1> %82), !llvm.114, !llvm.access.group.112
%89 = getelementptr inbounds float, float*87, i64 8
%90 = bitcast float*89 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>* %90, i32 4, <8 x i1> %83), !llvm.114, !llvm.access.group.112
%91 = or <8 x i64> %broadcast.splat, <64 80, i64 81, i64 82, i64 83, i64 84, i64 85, i64 86, i64 87>
%92 = trunc <8 x i64> %91 to <8 x i32>, !llvm.access.group.112
%93 = trunc i64 %mul.i.i to i32
%94 = or i32 %93, 8
%95 = insertelement <8 x i32> undef, i32 %94, i64 0
%96 = shufflevector <8 x i32> %95, <8 x i32> undef, <8 x i32> zeroinitializer
%97 = or <8 x i32> %96, <i32 80, i32 81, i32 82, i32 83, i32 84, i32 85, i32 86, i32 87>
%98 = icmp sgt <8 x i32> %broadcast.splat14, %92, !llvm.access.group.112
%99 = icmp sgt <8 x i32> %broadcast.splat16, %97, !llvm.access.group.112
%100 = extractelement <8 x i64> %91, i32 0
%101 = shl i64 %100, 32, !llvm.access.group.112
%102 = ashr exact i64 %101, 32, !llvm.access.group.112
%103 = getelementptr inbounds float, float* %0, i64 %102, !llvm.access.group.112
%104 = bitcast float*%103 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%104, i32 4, <8 x i1> %98), !llvm.114, !llvm.access.group.112
%105 = getelementptr inbounds float, float* %103, i64 8
%106 = bitcast float*%105 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%106, i32 4, <8 x i1> %99), !llvm.114, !llvm.access.group.112
%107 = or <8 x i64> %broadcast.splat, <64 96, i64 97, i64 98, i64 99, i64 100, i64 101, i64 102, i64 103>
%108 = trunc <8 x i64> %107 to <8 x i32>, !llvm.access.group.112
%109 = trunc i64 %mul.i.i to i32
%110 = or i32 %109, 8
%111 = insertelement <8 x i32> undef, i32 %110, i64 0
%112 = shufflevector <8 x i32> %111, <8 x i32> undef, <8 x i32> zeroinitializer
%113 = or <8 x i32> %112, <i32 96, i32 97, i32 98, i32 99, i32 100, i32 101, ...32 102, i32 103>
%114 = icmp sgt <8 x i32> %broadcast.splat14, %108, !llvm.access.group.112
%116 = icmp sgt <8 x i32> %broadcast.splat16, %113, !llvm.access.group.112
%118 = sext <8 x i64> %107, 32
%117 = shl i64 %116, 32, !llvm.access.group.112
%118 = ashr exact i64 %117, 32, !llvm.access.group.112
%119 = getelementptr inbounds float, float* %0, i64 %118, !llvm.access.group.112
%120 = bitcast float*%119 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%120, i32 4, <8 x i1> %114), !llvm.114, !llvm.access.group.112
%121 = getelementptr inbounds float, float*%119, i64 8
%122 = bitcast float*%121 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%122, i32 4, <8 x i1> %116), !llvm.114, !llvm.access.group.112
%123 = or <8 x i64> %broadcast.splat, <64 112, i64 113, i64 114, i64 115, i64 116, i64 117, i64 118, i64 119>
%124 = trunc <8 x i64> %123 to <8 x i32>, !llvm.access.group.112
%125 = trunc i64 %mul.i.i to i32
%126 = or i32 %125, 8
%127 = insertelement <8 x i32> undef, i32 %126, i64 0
%128 = shufflevector <8 x i32> %127, <8 x i32> undef, <8 x i32> zeroinitializer
%129 = or <8 x i32> %128, <i32 112, i32 113, i32 114, i32 115, i32 116, i32 117, i32 118, i32 119>
%130 = icmp sgt <8 x i32> %broadcast.splat14, %124, !llvm.access.group.112
%133 = icmp sgt <8 x i32> %broadcast.splat16, %129, !llvm.access.group.112
%134 = sext <8 x i64> %123 to <8 x i64>
%135 = shl i64 %132, 32, !llvm.access.group.112
%134 = ashr exact i64 %133, 32, !llvm.access.group.112
%135 = getelementptr inbounds float, float* %0, i64 %134, !llvm.access.group.112
%136 = bitcast float*%135 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%136, i32 4, <8 x i1> %130), !llvm.114, !llvm.access.group.112
%138 = getelementptr inbounds float, float*%135, i64 128, i64 129, i64 130, i64 131, i64 132, i64 133, i64 134, i64 135>
%140 = trunc <8 x i64> %139 to <8 x i32>, !llvm.access.group.112
%141 = trunc i64 %mul.i.i to i32
%142 = or i32 %141, 8
%143 = insertelement <8 x i32> undef, i32 %142, i64 0
%144 = shufflevector <8 x i32> %143, <8 x i32> undef, <8 x i32> zeroinitializer
%145 = or <8 x i32> %144, <i32 128, i32 129, i32 130, i32 131, i32 132, i32 133, i32 134, i32 135>
%146 = icmp sgt <8 x i32> %broadcast.splat14, %140, !llvm.access.group.112
%147 = icmp sgt <8 x i32> %broadcast.splat16, %145, !llvm.access.group.112
%148 = sext <8 x i64> %139 to <8 x i64>
%149 = shl i64 %148, 32, !llvm.access.group.112
%150 = ashr exact i64 %149, 32, !llvm.access.group.112
%151 = getelementptr inbounds float, float* %0, i64 %150, !llvm.access.group.112
%152 = bitcast float*%151 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%152, i32 4, <8 x i1> %146), !llvm.114, !llvm.access.group.112
%153 = getelementptr inbounds float, float*%151, i64 8
%154 = bitcast float*%153 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%154, i32 4, <8 x i1> %147), !llvm.114, !llvm.access.group.112
%155 = or <8 x i64> %broadcast.splat, <64 144, i64 145, i64 146, i64 147, i64 148, i64 149, i64 150, i64 151>
%156 = trunc <8 x i64> %155 to <8 x i32>, !llvm.access.group.112
%157 = trunc i64 %mul.i.i to i32
%158 = or i32 %157, 8
%159 = insertelement <8 x i32> undef, i32 %158, i64 0
%160 = shufflevector <8 x i32> %159, <8 x i32> undef, <8 x i32> zeroinitializer
%161 = or <8 x i32> %160, <i32 144, i32 145, i32 146, i32 147, i32 148, i32 149, i32 150, i32 151>
%162 = icmp sgt <8 x i32> %broadcast.splat14, %156, !llvm.access.group.112
%163 = icmp sgt <8 x i32> %broadcast.splat16, %161, !llvm.access.group.112
%164 = sext <8 x i64> %155 to <8 x i64>
%165 = shl i64 %164, 32, !llvm.access.group.112
%166 = ashr exact i64 %165, 32, !llvm.access.group.112
%167 = getelementptr inbounds float, float* %0, i64 %166, !llvm.access.group.112
%168 = bitcast float*%167 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%168, i32 4, <8 x i1> %162), !llvm.114, !llvm.access.group.112
%169 = getelementptr inbounds float, float*%167, i64 8
%170 = bitcast float*%169 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%170, i32 4, <8 x i1> %163), !llvm.114, !llvm.access.group.112
%171 = or <8 x i64> %broadcast.splat, <64 160, i64 161, i64 162, i64 163, i64 164, i64 165, i64 166, i64 167>
%172 = trunc <8 x i64> %171 to <8 x i32>, !llvm.access.group.112
%173 = trunc i64 %mul.i.i to i32
%174 = or i32 %173, 8
%175 = insertelement <8 x i32> undef, i32 %174, i64 0
%176 = shufflevector <8 x i32> %175, <8 x i32> undef, <8 x i32> zeroinitializer
%177 = or <8 x i32> %176, <i32 160, i32 161, i32 162, i32 163, i32 164, i32 165, i32 166, i32 167>
%178 = icmp sgt <8 x i32> %broadcast.splat14, %172, !llvm.access.group.112
%179 = icmp sgt <8 x i32> %broadcast.splat16, %177, !llvm.access.group.112
%180 = extractelement <8 x i64> %171, i32 0
%181 = shl i64 %180, 32, !llvm.access.group.112
%182 = ashr exact i64 %181, 32, !llvm.access.group.112
%183 = getelementptr inbounds float, float* %0, i64 %182, !llvm.access.group.112
%184 = bitcast float*%183 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%184, i32 4, <8 x i1> %178), !llvm.114, !llvm.access.group.112
%185 = getelementptr inbounds float, float*%183, i64 8
%186 = bitcast float*%185 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%186, i32 4, <8 x i1> %179), !llvm.114, !llvm.access.group.112
%187 = or <8 x i64> %broadcast.splat, <64 176, i64 177, i64 178, i64 179, i64 180, i64 181, i64 182, i64 183>
%188 = trunc <8 x i64> %187 to <8 x i32>, !llvm.access.group.112
%189 = trunc i64 %mul.i.i to i32
%190 = or i32 %189, 8
%191 = insertelement <8 x i32> undef, i32 %190, i64 0
%192 = shufflevector <8 x i32> %191, <8 x i32> undef, <8 x i32> zeroinitializer
%193 = or <8 x i32> %192, <i32 176, i32 177, i32 178, i32 179, i32 180, i32 181, i32 182, i32 183>
%194 = icmp sgt <8 x i32> %broadcast.splat14, %188, !llvm.access.group.112
%195 = icmp sgt <8 x i32> %broadcast.splat16, %193, !llvm.access.group.112
%196 = extractelement <8 x i64> %187, i32 0
%197 = shl i64 %196, 32, !llvm.access.group.112
%198 = ashr exact i64 %197, 32, !llvm.access.group.112
%199 = getelementptr inbounds float, float* %0, i64 %198, !llvm.access.group.112
%200 = bitcast float*%199 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%200, i32 4, <8 x i1> %194), !llvm.114, !llvm.access.group.112
%201 = getelementptr inbounds float, float*%199, i64 8
%202 = bitcast float*%201 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%202, i32 4, <8 x i1> %195), !llvm.114, !llvm.access.group.112
%203 = or <8 x i64> %broadcast.splat, <64 192, i64 193, i64 194, i64 195, i64 196, i64 197, i64 198, i64 199>
%204 = trunc <8 x i64> %203 to <8 x i32>, !llvm.access.group.112
%205 = trunc i64 %mul.i.i to i32
%206 = or i32 %205, 8
%207 = insertelement <8 x i32> undef, i32 %206, i64 0
%208 = shufflevector <8 x i32> %207, <8 x i32> undef, <8 x i32> zeroinitializer
%209 = or <8 x i32> %208, <i32 192, i32 193, i32 194, i32 195, i32 196, i32 197, i32 198, i32 199>
%210 = icmp sgt <8 x i32> %broadcast.splat14, %204, !llvm.access.group.112
%213 = icmp sgt <8 x i32> %broadcast.splat16, %209, !llvm.access.group.112
%213 = extractelement <8 x i64> %203, i32 0
%213 = shl i64 %212, 32, !llvm.access.group.112
%213 = ashr exact i64 %213, 32, !llvm.access.group.112
%213 = getelementptr inbounds float, float* %0, i64 %214, !llvm.access.group.112
%216 = bitcast float*%215 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%216, i32 4, <8 x i1> %210), !llvm.114, !llvm.access.group.112
%217 = getelementptr inbounds float, float*%215, i64 8
%218 = bitcast float*%217 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%218, i32 4, <8 x i1> %213), !llvm.114, !llvm.access.group.112
%219 = or <8 x i64> %broadcast.splat, <64 208, i64 209, i64 210, i64 211, i64 212, i64 213, i64 214, i64 215>
%220 = trunc <8 x i64> %219 to <8 x i32>, !llvm.access.group.112
%221 = trunc i64 %mul.i.i to i32
%222 = or i32 %221, 8
%223 = insertelement <8 x i32> undef, i32 %222, i64 0
%224 = shufflevector <8 x i32> %223, <8 x i32> undef, <8 x i32> zeroinitializer
%225 = or <8 x i32> %224, <i32 208, i32 209, i32 210, i32 211, i32 212, i32 213, i32 214, i32 215>
%226 = icmp sgt <8 x i32> %broadcast.splat14, %220, !llvm.access.group.112
%228 = icmp sgt <8 x i32> %broadcast.splat16, %225, !llvm.access.group.112
%228 = extractelement <8 x i64> %219, i32 0
%229 = shl i64 %228, 32, !llvm.access.group.112
%229 = ashr exact i64 %229, 32, !llvm.access.group.112
%229 = getelementptr inbounds float, float* %0, i64 %230, !llvm.access.group.112
%232 = bitcast float*%231 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%232, i32 4, <8 x i1> %226), !llvm.114, !llvm.access.group.112
%233 = getelementptr inbounds float, float*%231, i64 8
%234 = bitcast float*%233 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%234, i32 4, <8 x i1> %227), !llvm.114, !llvm.access.group.112
%235 = or <8 x i64> %broadcast.splat, <64 224, i64 225, i64 226, i64 227, i64 228, i64 229, i64 230, i64 231>
%236 = trunc <8 x i64> %235 to <8 x i32>, !llvm.access.group.112
%237 = trunc i64 %mul.i.i to i32
%238 = or i32 %237, 8
%239 = insertelement <8 x i32> undef, i32 %238, i64 0
%240 = shufflevector <8 x i32> %239, <8 x i32> undef, <8 x i32> zeroinitializer
%241 = or <8 x i32> %240, <i32 224, i32 225, i32 226, i32 227, i32 228, i32 229, i32 230, i32 231>
%242 = icmp sgt <8 x i32> %broadcast.splat14, %236, !llvm.access.group.112
%243 = icmp sgt <8 x i32> %broadcast.splat16, %241, !llvm.access.group.112
%243 = extractelement <8 x i64> %235, i32 0
%245 = shl i64 %244, 32, !llvm.access.group.112
%246 = ashr exact i64 %245, 32, !llvm.access.group.112
%247 = getelementptr inbounds float, float* %0, i64 %246, !llvm.access.group.112
%248 = bitcast float*%247 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%248, i32 4, <8 x i1> %242), !llvm.114, !llvm.access.group.112
%249 = getelementptr inbounds float, float*%247, i64 8
%250 = bitcast float*%249 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%250, i32 4, <8 x i1> %243), !llvm.114, !llvm.access.group.112
%251 = or <8 x i64> %broadcast.splat, <64 240, i64 241, i64 242, i64 243, i64 244, i64 245, i64 246, i64 247>
%252 = trunc <8 x i64> %251 to <8 x i32>, !llvm.access.group.112
%253 = trunc i64 %mul.i.i to i32
%254 = or i32 %253, 8
%255 = insertelement <8 x i32> undef, i32 %254, i64 0
%256 = shufflevector <8 x i32> %255, <8 x i32> undef, <8 x i32> zeroinitializer
%257 = or <8 x i32> %256, <i32 240, i32 241, i32 242, i32 243, i32 244, i32 245, i32 246, i32 247>
%258 = icmp sgt <8 x i32> %broadcast.splat14, %252, !llvm.access.group.112
%259 = icmp sgt <8 x i32> %broadcast.splat16, %257, !llvm.access.group.112
%259 = extractelement <8 x i64> %251, i32 0
%261 = shl i64 %260, 32, !llvm.access.group.112
%262 = ashr exact i64 %261, 32, !llvm.access.group.112
%263 = getelementptr inbounds float, float* %0, i64 %262, !llvm.access.group.112
%264 = bitcast float*%263 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat18, <8 x float>*%264, i32 4, <8 x i1> %258), !llvm.114, !llvm.access.group.112
%265 = getelementptr inbounds float, float*%263, i64 8
%266 = bitcast float*%265 to <8 x float>*
call void @llvm.masked.store.v8f32.p0v8f32(<8 x float> %broadcast.splat20, <8 x float>*%266, i32 4, <8 x i1> %259), !llvm.114, !llvm.access.group.112
br label %inner_kernel.exit
pregion_for_entry.entry.i.ua:
%_local_id_x.0.ua = phi i64 [%270, %if.end.r_exit.i.ua], [0, %pregion_for_entry.entry.i.ua.preheader]
%add1.i.i.ua = add nsw nsw i64 %_local_id_x.0.ua, %mul.i.i, !llvm.access.group.112
%conv.i.i.ua = trunc i64 %add1.i.i.ua to i32, !llvm.access.group.112
%cmp.i.i.ua = icmp slt i32 %conv.i.i.ua, %3, !llvm.access.group.112
br i1 %cmp.i.i.ua, label %if.then.i.ua, label %if.end.r_exit.i.ua, !llvm.access.group.112
T
F
if.then.i.ua:
%sext.i.ua = shl i64 %add1.i.i.ua, 32, !llvm.access.group.112
%idxprom.i.ua = ashr exact i64 %sext.i.ua, 32, !llvm.access.group.112
%arrayidx.i.ua = getelementptr inbounds float, float* %0, i64 %idxprom.i.ua, !llvm.access.group.112
store float 0.000000e+00, float*%arrayidx.i.ua, align 4, !llvm.114, !llvm.access.group.112
br label %for.body.i.ua, !llvm.access.group.112
for.body.i.ua:
%indvars.iv.next.i5.ua = phi i64 [%indvars.iv.next.i.ua, %for.body.i.ua], [0, %if.then.i.ua]
%add8.i2.ua = phi float [%add8.i.ua, %for.body.i.ua] [0.000000e+00, %if.then.i.ua]
%267 = mul nsw i64 %indvars.iv.next.i5.ua, %10, !llvm.access.group.112
%sext.i.ua = shl i64 %add1.i.i.ua, 32, !llvm.access.group.112
%arrayidx5.i.ua = getelementptr inbounds float, float* %1, i64 %268, !llvm.access.group.112
%add8.i.ua = load float, %add8.i2.ua, %269, !llvm.access.group.112
store float %add8.i.ua, float*%arrayidx.i.ua, align 4, !llvm.114, !llvm.access.group.112
%exitcond.not.i.ua = icmp eq i64 %indvars.iv.next.i5.ua, 1, !llvm.access.group.112
br i1 %exitcond.not.i.ua, label %for.end.loopexit.i.ua, label %for.body.i.ua, !llvm.loop.118, !llvm.access.group.112
T
F
for.end.loopexit.i.ua:
%add8.i.ua.lcssa = phi float [%add8.i.ua, %for.body.i.ua]
%div.i.ua = fdiv float %add8.i.ua.lcssa, %2, !llvmath.720, !llvm.access.group.112
store float %div.i.ua, float*%arrayidx.i.ua, align 4, !llvm.114, !llvm.access.group.112
br label %if.end.r_exit.i.ua, !llvm.access.group.112
if.end.r_exit.i.ua:
%270 = add nsw nsw i64 %_local_id_x.0.ua, 1
%exitcond.not = icmp eq i64 %270, 256
br i1 %exitcond.not, label %inner_kernel.exit.loopexit, label %pregion_for_entry.entry.i.ua, !llvm.loop.121
T
F
inner_kernel.exit.loopexit:
br label %inner_kernel.exit
inner_kernel.exit:
ret void
CFG for '_pocl_kernel_mean_kernel' function