%fdct1 = getelementptr inbounds %struct.jpeg_compress_struct, ... %struct.jpeg_compress_struct* %cinfo, i64 0, i32 58 %0 = bitcast %struct.jpeg_forward_dct** %fdct1 to ... %struct.my_fdct_controller** %1 = load %struct.my fdct controller*, %struct.my fdct controller** %0, .. align 8, !tbaa !3 %num_components = getelementptr inbounds %struct.jpeg_compress_struct, .. %struct.jpeg_compress_struct* %cinfo, i64 0, i32 12 $\%2 = \text{load i} 32, \text{i} 32* \%\text{num_components, align 4, !tbaa !} 11$ %cmp195 = icmp sgt i32 %2, 0 br i1 %cmp195, label %for.body.lr.ph, label %for.end116, !prof!12 for.body: %ci.0197 = phi i32 [0, %for.body.lr.ph], [%inc115, %for.inc114] %compptr.0196 = phi %struct.jpeg_component_info* [%3, %for.body.lr.ph], [.. %incdec.ptr, %for.inc114] %quant_tbl_no = getelementptr inbounds %struct.jpeg_component_info, .. %struct.jpeg_component_info* %compptr.0196, i64 0, i32 4 %5 = load i32, i32* %quant_tbl_no, align 8, !tbaa !14 %6 = icmp ugt i 32 %5, 3%.pre214 = sext i32 %5 to i64 %.pre = getelementptr inbounds %struct.jpeg_compress_struct, .. %struct.jpeg_compress_struct* %cinfo, i64 0, i32 15, i64 %.pre214 br i1 %6, label %if.then, label %lor.lhs.false4, !prof!16 lor.lhs.false4: %7 = load %struct.JQUANT_TBL*, %struct.JQUANT_TBL** %.pre, align 8, !tbaa !17 %cmp5 = icmp eq %struct.JQUANT_TBL* %7, null br i1 %cmp5, label %if.then, label %if.end, !prof!18 if.end: %10 = phi %struct.JQUANT_TBL* [%.pre213, %if.then], [%7, %lor.lhs.false4] %11 = load i32, i32* %dct_method, align 4, !tbaa !25 switch i32 %11, label %sw.default [i32 0, label %sw.bb i32 1, label %sw.bb31 i32 2, label %sw.bb65], !prof !26 def sw.bb: %arrayidx14 = getelementptr inbounds %struct.my_fdct_controller, .. %struct.my_fdct_controller* %1, i64 0, i32 2, i64 %.pre214 %12 = load i32*, i32** %arrayidx14, align 8, !tbaa !17 %cmp15 = icmp eq i32* %12, null br i1 %cmp15, label %if.then16, label %vector.ph, !prof!27 vector.ph: %17 = phi i32* [%16, %if.then16], [%12, %sw.bb] br label %vector.body vector.body: %18 = bitcast %struct.JQUANT_TBL* %10 to <4 x i16>* %wide.load = load <4 x i16>, <4 x i16>* %18, align 2, !tbaa !31 %19 = zext < 4 x i 16 > % wide.load to < 4 x i 32 > 6%20 = shl nuw nsw <4 x i32> %19, <i32 3, i32 3, i32 3, i32 3> %21 = bitcast i32* %17 to <4 x i32>*store <4 x i32> %20, <4 x i32>* %21, align 4, !tbaa !23 %22 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, .. i64 0, i32 0, i64 4 %23 = bitcast i16* %22 to <4 x i16>*%wide.load.1 = load <4 x i16>, <4 x i16>* %23, align 2, !tbaa !31 %24 = zext < 4 x i16 > %wide.load.1 to < 4 x i32 > $%25 = \text{shl nuw nsw} < 4 \times i32 > %24, < i32 3, i32 3, i32 3, i32 3 > %24, < i32 3, i$ %26 = getelementptr inbounds i32, i32* %17, i64 4 %27 = bitcast i32* %26 to <4 x i32>*store <4 x i32> %25, <4 x i32>* %27, align 4, !tbaa !23 %28 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, .. i64 0, i32 0, i64 8 %29 = bitcast i16* %28 to <4 x i16>*%wide.load.2 = load <4 x i16>, <4 x i16>* %29, align 2, !tbaa !31 $%30 = zext < 4 \times i16 > %wide.load.2 to < 4 \times i32 >$ %31 = shl nuw nsw <4 x i32> %30, <i32 3, i32 3, i32 3, i32 3> %32 = getelementptr inbounds i32, i32* %17, i64 8 %33 = bitcast i32* %32 to <4 x i32>*store <4 x i32> %31, <4 x i32>* %33, align 4, !tbaa !23 for.cond85.preheader: %34 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %indvars.iv204 = phi i64 [0, %if.end77], [%indvars.iv.next205, .. i64 0, i32 0, i64 12 ... %for.cond85.preheader] %35 = bitcast i16* %34 to <4 x i16>*%indvars.iv202 = phi i64 [0, %if.end77], [%indvars.iv.next203, %wide.load.3 = load <4 x i16>, <4 x i16>* %35, align 2, !tbaa !31 ... %for.cond85.preheader] $%36 = zext < 4 \times i16 > %wide.load.3 to < 4 \times i32 >$ %arrayidx94 = getelementptr inbounds [8 x double], [8 x double]* %37 = shl nuw nsw <4 x i32> %36, <i32 3, i32 3, i32 3, i32 3> ... @start_pass_fdctmgr.aanscalefactor, i64 0, i64 %indvars.iv202 %38 = getelementptr inbounds i32, i32* %17, i64 12 %149 = load double, double* %arrayidx94, align 8 %39 = bitcast i32* %38 to <4 x i32>*%arrayidx91 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* store <4 x i32> %37, <4 x i32>* %39, align 4, !tbaa !23 ... %10, i64 0, i32 0, i64 %indvars.iv204 %40 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %150 = load i16, i16* %arrayidx91, align 2, !tbaa !31 .. i64 0, i32 0, i64 16 %conv92 = uitofp i16 %150 to double %41 = bitcast i16* %40 to <4 x i16>*%arrayidx102 = getelementptr inbounds float, float* %148, i64 %indvars.iv204 %wide.load.4 = load <4 x i16>, <4 x i16>* %41, align 2, !tbaa !31 %indvars.iv.next201 = or i64 %indvars.iv204, 1 %42 = zext < 4 x i16 > %wide.load.4 to < 4 x i32 >%arrayidx91.1 = getelementptr inbounds %struct.JQUANT_TBL, %43 = shl nuw nsw <4 x i32> %42, <i32 3, i32 3, i32 3, i32 3> ... %struct.JQUANT_TBL* %10, i64 0, i32 0, i64 %indvars.iv.next201 %44 = getelementptr inbounds i32, i32* %17, i64 16 %151 = load i16, i16* %arrayidx91.1, align 2, !tbaa !31 %45 = bitcast i32* %44 to <4 x i32>*%conv92.1 = uitofp i16 %151 to double store <4 x i32> %43, <4 x i32>* %45, align 4, !tbaa !23 %mul95.1 = fmul double %conv92.1, %149 %46 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %indvars.iv.next201.1 = add nsw i64 %indvars.iv.next201, 1 .. i64 0, i32 0, i64 20 %arrayidx91.2 = getelementptr inbounds %struct.JQUANT_TBL, %47 = bitcast i16* %46 to <4 x i16>* .. %struct.JQUANT_TBL* %10, i64 0, i32 0, i64 %indvars.iv.next201.1 %wide.load.5 = load <4 x i16>, <4 x i16>* %47, align 2, !tbaa !31 %152 = load i16, i16* %arrayidx91.2, align 2, !tbaa !31 %48 = zext < 4 x i16 > %wide.load.5 to < 4 x i32 >%conv92.2 = uitofp i16 %152 to double %49 = shl nuw nsw <4 x i32> %48, <i32 3, i32 3, i32 3, i32 3> %mul95.2 = fmul double %conv92.2, %149 %50 = getelementptr inbounds i32, i32* %17, i64 20 vector.body219: %indvars.iv.next201.2 = or i64 %indvars.iv204, 3 %51 = bitcast i32* %50 to <4 x i32>*%index224 = phi i64 [0, %vector.ph223], [%index.next225.1, %arrayidx91.3 = getelementptr inbounds %struct.JQUANT_TBL, store <4 x i32> %49, <4 x i32>* %51, align 4, !tbaa !23 ... %struct.JQUANT_TBL* %10, i64 0, i32 0, i64 %indvars.iv.next201.2 .. %vector.body219] %52 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %118 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %153 = load i16, i16* %arrayidx91.3, align 2, !tbaa !31 i 64 0, i 32 0, i 64 24 . i64 0, i32 0, i64 %index224 %conv92.3 = uitofp i16 %153 to double %53 = bitcast i16* %52 to <4 x i16>*%119 = bitcast i16* %118 to <2 x i16>* %mul95.3 = fmul double %conv92.3, %149 %wide.load.6 = load <4 x i16>, <4 x i16>* %53, align 2, !tbaa !31 %wide.load231 = load <2 x i16>, <2 x i16>* %119, align 2, !tbaa !31 %154 = insertelement <4 x double> undef, double %conv92, i32 0 $%54 = zext < 4 \times i16 > %wide.load.6 to < 4 \times i32 >$ %120 = zext < 2 x i 16 > % wide.load 231 to < 2 x i 64 > 64%155 = insertelement <4 x double> %154, double %mul95.1, i32 1 %55 = shl nuw nsw <4 x i32> %54, <i32 3, i32 3, i32 3, i32 3> %121 = getelementptr inbounds [64 x i16], [64 x i16]*%156 = insertelement <4 x double> %155, double %mul95.2, i32 2 %56 = getelementptr inbounds i32, i32* %17, i64 24 @start_pass_fdctmgr.aanscales, i64 0, i64 %index224 %157 = insertelement <4 x double> %156, double %mul95.3, i32 3 %57 = bitcast i32* %56 to <4 x i32>*%122 = bitcast i16* %121 to <2 x i16>*%158 = insertelement <4 x double> undef, double %149, i32 0 store <4 x i32> %55, <4 x i32>* %57, align 4, !tbaa !23 %wide.load232 = load <2 x i16>, <2 x i16>* %122, align 8, !tbaa !31 %159 = insertelement <4 x double> %158, double 0x3FF63150B14861EF, i32 1 %58 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, $%123 = \text{sext} < 2 \times 116 > \% \text{ wide.load} = 232 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{ to } < 2 \times 164 > 322 \text{$ %160 = insertelement <4 x double> %159, double 0x3FF4E7AE914D6FCA, i32 2 . i64 0, i32 0, i64 28 %124 = mul nsw <2 x i64> %123, %120 %161 = insertelement <4 x double> %160, double 0x3FF2D062EF6C11AA, i32 3 %59 = bitcast i16* %58 to <4 x i16>*%125 = add nsw <2 x i64> %124, <i64 1024, i64 1024> %162 = fmul <4 x double> %157, %161 %wide.load.7 = load <4 x i16>, <4 x i16>* %59, align 2, !tbaa !31 $%163 = \text{fmul} < 4 \text{ x double} > %162, < \text{double } 8.000000e + 00, double } 8.000000e + 00,$ %126 = lshr <2 x i64> %125, <i64 11, i64 11> $\%60 = \text{zext} < 4 \times i16 > \% \text{ wide.load.7 to } < 4 \times i32 > 6 \times i26 = 2 \times i26 \times i26 = 2 \times i26 \times i26$. double 8.000000e+00, double 8.000000e+00> $%127 = \text{trunc} < 2 \times i64 > %126 \text{ to} < 2 \times i32 >$ %61 = shl nuw nsw <4 x i32> %60, <i32 3, i32 3, i32 3, i32 3> %128 = getelementptr inbounds i32, i32* %117, i64 %index224 %164 = fdiv <4 x double> <double 1.000000e+00, double 1.000000e+00, double %62 = getelementptr inbounds i32, i32* %17, i64 28 %129 = bitcast i32* %128 to <2 x i32>*.. 1.000000e+00, double 1.000000e+00>, %163 %63 = bitcast i32* %62 to <4 x i32>*store <2 x i32> %127, <2 x i32>* %129, align 4, !tbaa !23 %165 = fptrunc < 4 x double > %164 to < 4 x float >store <4 x i32> %61, <4 x i32>* %63, align 4, !tbaa !23 %index.next225 = or i64 %index224, 2 %166 = bitcast float* %arrayidx102 to <4 x float>* %64 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %130 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, store <4 x float> %165, <4 x float>* %166, align 4, !tbaa !35 .. i64 0, i32 0, i64 32 . i64 0, i32 0, i64 %index.next225 %indvars.iv.next201.3 = add nsw i64 %indvars.iv.next201.2, 1 %65 = bitcast i16* %64 to <4 x i16>* %131 = bitcast i16* %130 to <2 x i16>* %arrayidx91.4 = getelementptr inbounds %struct.JQUANT_TBL, %wide.load.8 = load <4 x i16>, <4 x i16>* %65, align 2, !tbaa !31 % wide.load231.1 = load <2 x i16>, <2 x i16>* %131, align 2, !tbaa !31 . %struct.JQUANT_TBL* %10, i64 0, i32 0, i64 %indvars.iv.next201.3 $\%66 = \text{zext} < 4 \times 116 > \% \text{ wide.load.} 8 \text{ to } < 4 \times 132 > 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 = 100 =$ %132 = zext <2 x i16> %wide.load231.1 to <2 x i64> %167 = load i16, i16* %arrayidx91.4, align 2, !tbaa !31 %67 = shl nuw nsw <4 x i32> %66, <i32 3, i32 3, i32 3, i32 3> %133 = getelementptr inbounds [64 x i 16], [64 x i 16]*%conv92.4 = uitofp i16 %167 to double %68 = getelementptr inbounds i32, i32* %17, i64 32 . @start_pass_fdctmgr.aanscales, i64 0, i64 %index.next225 %arrayidx102.4 = getelementptr inbounds float, float* %148, i64 %69 = bitcast i32* %68 to <4 x i32>*%134 = bitcast i16* %133 to <2 x i16>*. %indvars.iv.next201.3 store <4 x i32> %67, <4 x i32>* %69, align 4, !tbaa !23 % wide.load232.1 = load <2 x i16>, <2 x i16>* %134, align 4, !tbaa !31 %indvars.iv.next201.4 = add nsw i64 %indvars.iv.next201.2, 2 %70 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %arrayidx91.5 = getelementptr inbounds %struct.JQUANT_TBL, $%135 = \text{sext} < 2 \times 116 > \% \text{ wide.load232.1 to } < 2 \times 164 > 6$.. i64 0, i32 0, i64 36 %136 = mul nsw <2 x i64> %135, %132 ... %struct.JQUANT_TBL* %10, i64 0, i32 0, i64 %indvars.iv.next201.4 %71 = bitcast i16* %70 to <4 x i16>*%168 = load i16, i16* %arrayidx91.5, align 2, !tbaa !31 %137 = add nsw <2 x i64> %136, <i64 1024, i64 1024> %wide.load.9 = load <4 x i16>, <4 x i16>* %71, align 2, !tbaa !31 %conv92.5 = uitofp i16 %168 to double %138 = lshr <2 x i64> %137, <i64 11, i64 11> %mul95.5 = fmul double %conv92.5, %149 $%139 = \text{trunc} < 2 \times i64 > %138 \text{ to} < 2 \times i32 >$ $\%73 = \text{shl nuw nsw} < 4 \times i32 > \%72, < i32 3, i32 3, i32 3, i32 3 > \%73 = \text{shl nuw nsw} < 4 \times i32 > \%72, < i32 3, i32$ %140 = getelementptr inbounds i32, i32* %117, i64 %index.next225 %indvars.iv.next201.5 = add nsw i64 %indvars.iv.next201.2, 3 %74 = getelementptr inbounds i32, i32* %17, i64 36 %141 = bitcast i32* %140 to <2 x i32>*%arrayidx91.6 = getelementptr inbounds %struct.JQUANT_TBL, %75 = bitcast i32* %74 to <4 x i32>*store <2 x i32> %139, <2 x i32>* %141, align 4, !tbaa !23 .. %struct.JQUANT_TBL* %10, i64 0, i32 0, i64 %indvars.iv.next201.5 store <4 x i32> %73, <4 x i32>* %75, align 4, !tbaa !23 %index.next225.1 = add nsw i64 %index224, 4 %169 = load i16, i16* %arrayidx91.6, align 2, !tbaa !31 %76 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %142 = icmp eq i64 %index.next225.1, 64 %conv92.6 = uitofp i16 %169 to double . i64 0, i32 0, i64 40 %mul95.6 = fmul double %conv92.6, %149 br i1 %142, label %for.inc114.loopexit233, label %vector.body219, !llvm.loop %77 = bitcast i16* %76 to <4 x i16>*%indvars.iv.next201.6 = or i64 %indvars.iv204, 7 . !32 %wide.load.10 = load <4 x i16>, <4 x i16>* %77, align 2, !tbaa !31 %arrayidx91.7 = getelementptr inbounds %struct.JQUANT_TBL, $\%78 = \text{zext} < 4 \times 116 > \% \text{ wide.load.} 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 > 10 \text{ to } < 4 \times 132 >$... %struct.JQUANT_TBL* %10, i64 0, i32 0, i64 %indvars.iv.next201.6 %79 = shl nuw nsw <4 x i32> %78, <i32 3, i32 3, i32 3, i32 3> %170 = load i16, i16* %arrayidx91.7, align 2, !tbaa !31 %80 = getelementptr inbounds i32, i32* %17, i64 40 %conv92.7 = uitofp i16 %170 to double %81 = bitcast i32* %80 to <4 x i32>*%mul95.7 = fmul double %conv92.7, %149 store <4 x i32> %79, <4 x i32>* %81, align 4, !tbaa !23 %171 = insertelement <4 x double> undef, double %conv92.4, i32 0 %82 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %172 = insertelement <4 x double> %171, double %mul95.5, i32 1 .. i64 0, i32 0, i64 44 %173 = insertelement <4 x double> %172, double %mul95.6, i32 2 %83 = bitcast i16* %82 to <4 x i16>* %174 = insertelement <4 x double> %173, double %mul95.7, i32 3 %wide.load.11 = load <4 x i16>, <4 x i16>* %83, align 2, !tbaa !31 %175 = insertelement <4 x double> %158, double 0x3FE92469C0A7BF3B, i32 1 %84 = zext <4 x i16> % wide.load.11 to <4 x i32> %176 = insertelement <4 x double> %175, double 5.411961e-01, i32 2 %85 = shl nuw nsw <4 x i32> %84, <i32 3, i32 3, i32 3, i32 3> %177 = insertelement <4 x double> %176, double 0x3FD1A855DE72AB5D, i32 3 %86 = getelementptr inbounds i32, i32* %17, i64 44 %178 = fmul <4 x double> %174, %177 %87 = bitcast i32* %86 to <4 x i32>*%179 = fmul < 4 x double > %178, < double 8.000000e + 00, double 8.000000e + 00. store <4 x i32> %85, <4 x i32>* %87, align 4, !tbaa !23 .. double 8.000000e+00, double 8.000000e+00> %88 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, %180 = fdiv < 4 x double > < double 1.0000000e + 00, double 1.0000000e + 00, double.. i64 0, i32 0, i64 48 ... 1.000000e+00, double 1.000000e+00>, %179 %89 = bitcast i16* %88 to <4 x i16>* %181 = fptrunc <4 x double> %180 to <4 x float> %wide.load.12 = load <4 x i16>, <4 x i16>* %89, align 2, !tbaa !31 %182 = bitcast float* %arrayidx102.4 to <4 x float>* %90 = zext <4 x i16> %wide.load.12 to <4 x i32> store <4 x float> %181, <4 x float>* %182, align 4, !tbaa !35 %91 = shl nuw nsw <4 x i32> %90, <i32 3, i32 3, i32 3, i32 3> %indvars.iv.next203 = add nuw nsw i64 %indvars.iv202, 1 %92 = getelementptr inbounds i32, i32* %17, i64 48 %indvars.iv.next205 = add nuw nsw i64 %indvars.iv204, 8 %93 = bitcast i 32* %92 to <4 x i 32>*%exitcond206 = icmp eq i64 %indvars.iv.next203, 8 store <4 x i32> %91, <4 x i32>* %93, align 4, !tbaa !23 br i1 %exitcond206, label %for.inc114.loopexit234, label %94 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, ... %for.cond85.preheader .. i64 0, i32 0, i64 52 %95 = bitcast i16* %94 to <4 x i16>* %wide.load.13 = load <4 x i16>, <4 x i16>* %95, align 2, !tbaa !31 %96 = zext <4 x i16> % wide.load.13 to <4 x i32> %97 = shl nuw nsw <4 x i32> %96, <i32 3, i32 3, i32 3, i32 3> %98 = getelementptr inbounds i32, i32* %17, i64 52 %99 = bitcast i 32 %98 to <4 x i 32>*store <4 x i32> %97, <4 x i32>* %99, align 4, !tbaa !23 %100 = getelementptr inbounds %struct.JQUANT TBL, %struct.JQUANT TBL* %10, .. i64 0, i32 0, i64 56 %101 = bitcast i16* %100 to <4 x i16>* %wide.load.14 = load <4 x i16>, <4 x i16>* %101, align 2, !tbaa !31 %102 = zext <4 x i16> %wide.load.14 to <4 x i32> %103 = shl nuw nsw <4 x i32> %102, <i32 3, i32 3, i32 3, i32 3> %104 = getelementptr inbounds i32, i32* %17, i64 56 %105 = bitcast i32* %104 to <4 x i32>*store <4 x i32> %103, <4 x i32>* %105, align 4, !tbaa !23 %106 = getelementptr inbounds %struct.JQUANT_TBL, %struct.JQUANT_TBL* %10, .. i64 0, i32 0, i64 60 %107 = bitcast i16* %106 to <4 x i16>*%wide.load.15 = load <4 x i16>, <4 x i16>* %107, align 2, !tbaa !31 %108 = zext <4 x i16> %wide.load.15 to <4 x i32> %109 = shl nuw nsw <4 x i32> %108, <i32 3, i32 3, i32 3, i32 3> %110 = getelementptr inbounds i32, i32* %17, i64 60 %111 = bitcast i32* %110 to <4 x i32>*store <4 x i32> %109, <4 x i32>* %111, align 4, !tbaa !23 br label %for.inc114 for.inc114: %inc115 = add nuw nsw i32 %ci.0197, 1 %incdec.ptr = getelementptr inbounds %struct.jpeg_component_info, . %struct.jpeg_component_info* %compptr.0196, i64 1 %185 = load i32, i32* %num components, align 4, !tbaa !11 %cmp = icmp slt i32 %inc115, %185 br i1 %cmp, label %for.body, label %for.end116.loopexit, !prof !12 T for.end116: ret void

CFG for 'start_pass_fdctmgr' function