Here are the errors we have found: Highlighted in yellow:

```
Void TDecCu::xDecodePaletteTexture(TComDataCU* pcCU, const UInt uiPartIdx, Pel* pPalette,  Pel* pLevel, UChar *pSPoint, Pel *pPixelValue, Pel* piReco,const UInt uiStride, const UInt uiWidth, const UInt uiHeight, const ComponentID compID, UChar* pEscapeFlag )
{
```

```
  if(!bRotation)
  {
    for(UInt uiY = 0; uiY < uiHeight; uiY++ )
    {
      for(UInt uiX = 0; uiX < uiWidth; uiX++ )
      {
        uiIdx = (uiY<<pcCU->getPic()->getComponentScaleY(compID))*(uiWidth<<pcCU->getPic()->getComponentScaleX(compID))+(uiX<<pcCU->getPic()->getComponentScaleX(compID));
        UInt uiIdxComp = uiY*uiWidth + uiX;
        if( pEscapeFlag[uiIdx] )
        {
          if ( bLossless )
          {
            iValue = pPixelValue[uiIdxComp];
          }
          else
          {
            assert(!pcCU->getColourTransform(uiPartIdx));
            QpParam cQP(*pcCU, compID, uiPartIdx);
            Int iQP = cQP.Qp;
            Int iQPrem = iQP % 6;
            Int iQPper = iQP / 6;
            Int InvquantiserRightShift = IQUANT_SHIFT;
            Int iAdd = 1 << (InvquantiserRightShift - 1);
            iValue = ((((pPixelValue[uiIdxComp]*g_invQuantScales[iQPrem])<<iQPper) + iAdd)>>InvquantiserRightShift);
            iValue = Pel(ClipBD<Int>(iValue, pcCU->getSlice()->getSPS()->getBitDepths().recon[compID?1:0]));
          }
        }
        else
        {
          iValue = pPalette[pLevel[uiIdx]];
        }
        piReco[uiY*uiStride+uiX] = iValue;
        piPicReco[uiY*uiPicStride+uiX] = iValue;
      }
    }
  }
```

```
  else
  {
    for(UInt uiY = 0; uiY < uiWidth; uiY++ )
    {
      for(UInt uiX = 0; uiX < uiHeight; uiX++ )
      {
        uiIdx = (uiY<<pcCU->getPic()->getComponentScaleX(compID))*(uiHeight<<pcCU->getPic()->getComponentScaleY(compID))+(uiX<<pcCU->getPic()->getComponentScaleY(compID));
        UInt uiIdxComp = uiY*uiHeight + uiX;
        if( pEscapeFlag[uiIdx] )
        {
          if ( bLossless )
          {
            iValue = pPixelValue[uiIdxComp];
          }
          else
          {
            assert(!pcCU->getColourTransform(uiPartIdx));
            QpParam cQP(*pcCU, compID, uiPartIdx);
            Int iQP = cQP.Qp;
            Int iQPrem = iQP % 6;
            Int iQPper = iQP / 6;
            Int InvquantiserRightShift = IQUANT_SHIFT;
            Int iAdd = 1 << (InvquantiserRightShift - 1);
            iValue = ((((pPixelValue[uiIdxComp]*g_invQuantScales[iQPrem])<<iQPper) + iAdd)>>InvquantiserRightShift);
            iValue = Pel(ClipBD<Int>(iValue, pcCU->getSlice()->getSPS()->getBitDepths().recon[compID?1:0]));
          }
        }
        else
        {
          iValue = pPalette[pLevel[uiIdx]];
        }

        piReco[uiX*uiStride+uiY] = iValue;
        piPicReco[uiX*uiPicStride+uiY] = iValue;
      }
    }
  }
}
```

5. The following applies:

$$tmpVal = (PaletteEscapeVal[cIdx][xCb + xL][yCb + yL] * levelScale[qP\%6]) << (qP / 6) + 32) >> 6 \quad (8\text{-}77)$$

$$recSamples[x][y] = Clip3(0, (1 << bitDepth) - 1, tmpVal) \quad (8\text{-}78)$$