I.8.5.3.3.9.1 Derivation process for contour boundary filtered samples

If PartMode is PART\_Nx2N, the following apply,

for ( y = 0; y < nCbSX; y++ )  
 for( x = 0; x < nCbSX; x++ ) {  
 ~~tFlag = segMask[ n \* x ][ Max( 0, n \* ( y − 1 ) ) ]~~ lFlag = segMask[ Max( 0, (n \* ( x − 1 ) ) ][ n \* y ]  
 ~~bFlag = segMask[ n \* x ][ Min( n \* ( y + 1 ), nCbSL − 1 ) ]~~ rFlag = segMask[ Min( n \* ( x + 1 ), nCbSL − 1 ) ][ n\*y ]  
 ~~cFlag = segMask[ n \* x ][ n \* y ]~~  
 filt = p[ x ][ y ]

if ( lFlag != rFlag)

~~if( ( lFlag | | cFlag | | rFlag ) && ( !lFlag | | !cFlag | | !rFlag ) )~~  
 filt = ( p[ Max( 0, x − 1 ) ][ y ] + ( filt << 1 ) + p[ Min( x + 1, nCbSX − 1 ) ][ y ] ) >> 2

~~if( ( tFlag | | cFlag | | bFlag ) && ( !tFlag | | !cFlag | | !bFlag ) )~~  
 ~~filt = ( p[ x ][ Max( 0, y − 1 ) ] + ( filt << 1 ) + p[ x ][ Min( y + 1, nCbS~~~~X~~~~− 1 ) ] ) >> 2~~  
 predSamples[ x ][ y ] = filt  
 }

Otherwise (PartMode is PART\_2NxN), the following apply,

for ( y = 0; y < nCbSX; y++ )  
 for( x = 0; x < nCbSX; x++ ) {  
 tFlag = segMask[ n \* x ][ Max( 0, n \* ( y − 1 ) ) ]  
 bFlag = segMask[ n \* x ][ Min( n \* ( y + 1 ), nCbSL − 1 ) ]

filt = p[ x ][ y ]

if ( tFlag != bFlag)

filt = ( p[ x ][ Max( 0, y − 1 ) ] + ( filt << 1 ) + p[ x ][ Min( y + 1, nCbSX − 1 ) ] ) >> 2

predSamples[ x ][ y ] = filt  
 }