##### Derivation process for luma filter coefficients

Inputs to this process are

* + a variable alfPrecisionBit specifying the ALF coefficient precision

Outputs of this process are the luma filter coefficients cL.

The luma filter coefficients cL with elements cL[ i ][ j ], i = 0..AlfNumFilters–1, j = 0..9 are derived as follows:

* If alf\_nb\_pred\_luma\_flag[ i ] is equal to 0,
* If alf\_pred\_flag is equal to 0 or the value of i is equal to 0,

cL[ i ][ j ] = alf\_filt\_coeff [ 0 ][ ry ][ rx ][ i ][ j ] (8‑464)

* Otherwise (alf\_pred\_flag is equal to 1 and the value of i is greater than 1),

cL[ i ][ j ] = alf\_filt\_coeff [ 0 ][ ry ][ rx ][ i ][ j ] + cL[ i – 1 ][ j ] (8‑465)

* Otherwise (alf\_nb\_pred\_luma\_flag[i] is equal to 1), the following ordered steps apply:

1. The luma filter coefficients cL with elements cL[ i ][ j ], i = 0..AlfNumFilters–1, j = 0..7 are derived as follows:
   * + If alf\_pred\_flag is equal to 0 or the value of i is equal to 0,

cL[ i ][ j ] = alf\_filt\_coeff [ 0 ][ ry ][ rx ][ i ][ j ] (8‑464)

* + - Otherwise (alf\_pred\_flag is equal to 1 and the value of i is greater than 1),

cL[ i ][ j ] = alf\_filt\_coeff [ 0 ][ ry ][ rx ][ i ][ j ] + cL[ i – 1 ][ j ] (8‑465)

1. The luma filter coefficients cL with elements cL[ i ][ j ], i = 0..AlfNumFilters–1, j = 8 are derived as follows:
   * + If alf\_pred\_flag is equal to 0 or the value of i is equal to 0,

cL[ i ][ j ] = alf\_filt\_coeff [ 0 ][ ry ][ rx ][ i ][ j ] + ( 1 << alfPrecisionBit ) –  
 ( Σk2 \* alf\_filt\_coeff[ 0 ][ ry ][ rx ][ i ][ k ] ) >> 2 (8‑465)  
 with k=0..j−1

* + - Otherwise,

alf\_coeff\_luma\_nb[ i ] = alf\_filt\_coeff[ 0 ][ ry ][ rx ][ i ][ j ] –  
 ( Σk2\* alf\_filt\_coeff[ 0 ][ ry ][ rx ][ i ][ k ] ) >> 2 (8‑465)

cL[ i ][ j ] = alf\_coeff\_luma\_nb[ i ] + cL[ i – 1 ][ j ]  (8‑465)  
 with k=0..j−1

1. The luma filter coefficients cL with elements cL[ i ][ j ], i = 0..AlfNumFilters–1, j = 9 are derived as follows:
   * + If alf\_pred\_flag is equal to 0 or the value of i is equal to 0,

cL[ i ][ j ] = alf\_filt\_coeff[ 0 ][ ry ][ rx ][ i ][ j ] + ( 1 << alfPrecisionBit ) –  
 Σk(2\* alf\_filt\_coeff[ 0 ][ ry ][ rx ][ i ][ k ]) – 2\*cL[ i ][ j – 1 ] (8‑465)  
 with k=0..j−2

* + - Otherwise,

cL[ i ][ j ] = alf\_filt\_coeff[ 0 ][ ry ][ rx ][ i ][ j ] + cL[ i – 1 ][ j ] –  
 Σk(2\* alf\_filt\_coeff[ 0 ][ ry ][ rx ][ i ][ k ]) – 2\*alf\_coeff\_luma\_nb[ i ] (8‑465)  
 with k=0..j−2

Considering the symmetry of the filter, the luma filter coefficients cL with elements cL[ i ][ j ], i = 0..AlfNumFilters – 1, j = 10..18 are derived as follows:

cL[ i ][  j ] = cL[ i ][ 18 – j ] (8‑467)  
 with j = 10..18

##### Derivation process for chroma filter coefficients

Inputs to this process are

* + a variable cIdx specifying colour component index,
  + a variable alfPrecisionBit specifying the ALF coefficient precision

Outputs of this process are filter coefficients cc for the samples for the colour component cIdx.

The chroma filter coefficients cC with elements cC[ i ], i = 0..18 are derived as follows:

* If i is equal to 9, the coefficient cC[i] is derived as

sum = Σj( alf\_filt\_coeff [ cIdx ][ ry ][ rx ][ 0 ] [ j ] << 1 ) (8‑469)  
 with j = 0..8

cC[ i ] = ( 1 << alfPrecisionBit ) − sum + alf\_filt\_coeff [ cIdx ][ ry ][ rx ][ 0 ][ i ] (8‑468)

* Otherwise if i is less than 9, the chroma filter coefficients cC with elements cC[ i ], i = 0..8 are derived as follows:

cC[ i ] = alf\_filt\_coeff [ cIdx ][ ry ][ rx ][ 0 ] [ i ] (8‑471)

* Otherwise (if i is more than 9), considering the symmetry of the filter, the chroma filter coefficients cC with elements cC[ i ], i = 10..18 are derived as follows:

cC[ i ] = cC[ 18 − i ] (8‑472)  
 with i = 10..18