ALGORITHM 2.1

Algorithm for recursive set partitioning to encode a $2^m \times 2^m$ block of non-negative source elements.

(1) **Initialization:**

  (a) Find $n_{\max}$, the most significant bit of the largest magnitude of the source.

  (b) Create a list LIS (List of Insignificant Sets), initially empty, to contain '0'-labelled (called insignificant) sets.

  (c) Put descriptor of $2^m \times 2^m$ source block onto LIS (upper left corner coordinates, plus size m).

  (d) Set $n = n_{\max}$. Encode $n_{\max}$ and write to codestream buffer.

(2) **Testing and Partitioning:**

  (a) For each set on the LIS, do

    i. If maximum element in set is less than 2^n, write '0' to codestream buffer.

      A. if set has more than one element, return to 2a.

      B. if set has one element, write value using n bits to codestream buffer. Remove set from LIS.

    ii. Otherwise, if maximum element in set is greater than or equal to 2^n, write '1' to codestream buffer and do the following:

      A. if set has more than one element, divide the set into 4 equal quadrants, and add each quadrant to the end of the LIS.

      B. if set has one element, write its value minus 2^n using n bits to codestream buffer.

      C. remove set from LIS.

(3) If $n > 0$, set $n = n - 1$ and return to 2a. Otherwise stop.

ALGORITHM 2.2

Algorithm for recursive set partitioning to decode a $2^m \times 2^m$ block of source elements.

(1) Initialization:
   (a) Create a list LIS (List of Insignificant Sets), initially empty, to contain '0'-labelled (called insignificant) sets.
   (b) Put descriptor of $2^m \times 2^m$ source block onto LIS (upper left corner coordinates, plus size m).
   (c) Read from codestream buffer and decode $n_{\max}$. Set $n = n_{\max}$.

(2) **Partitioning and recovering values:**
   (a) For each set on the LIS, do
      i. Read next bit from codestream. If '0', do
         A. if set has more than one element, return to 2a.
         B. if set has one element, read next n bits from codestream and decode value. Remove set from LIS.
      ii. Otherwise, if '1', do the following:
         A. if set has more than one element, divide the set into 4 equal quadrants, and add each quadrant to the end of the LIS.
         B. if set has one element, read n bits from codestream to decode partial value. Value of '1'-associated element is decoded partial value plus 2^n.
         C. remove set from LIS.

(3) If $n > 0$, set $n = n - 1$ and return to 2a.

(4) Otherwise, set the value of all elements belonging to all sets remaining in the LIS to zero, and stop.

Table 2.6 Example of SPECK Coding of Wavelet Transform, From Bit-Plane $n = 5$

Comment	Point or Set	Output Bits	Action	Control Lists
$n = 5$ Sorting $\mathcal{S}$=$S^1(0,0)$, $\mathcal{I}$ = rest				LIS = {(0,0)**1**} LSP = ϕ
	$S^1(0,0)$	1	quad split, add to LIS(0)	LIS ={ (0,0)**0**, (0,1)**0**,(1,0)**0**,(1,1)**0**} LSP = ϕ
	(0,0)	1+	(0,0) to LSP	LIS = {(0,1)**0**, (1,0)**0**,(1,1)**0**} LSP = {(0,0)}
	(0,1)	1–	(0,1) to LSP	LIS = {(1,0)**0**,(1,1)**0**} LSP = {(0,0),(0,1)}
	(1,0)	0	none	
	(1,1)	0	none	
Test $\mathcal{I}$	$\mathcal{S}(\mathcal{I})$	1	split to 3 $\mathcal{S}$'s, new $\mathcal{I}$	
	$S^1(0,2)$	1	quad split, add to LIS(0)	LIS = {(1,0)**0**,(1,1)**0**},(0,2)**0**,(0,3)**0**}(1,2)**0**,(1,3)**0**}
	(0,2)	1+	(0,2) to LSP	LSP = {(0,0),(0,1),(0,2)} LIS = {(1,0)**0**,(1,1)**0**},(0,3)**0**}(1,2)**0**,(1,3)**0**}
	(0,3)	0	none	
	(1,2)	0	none	
	(1,3)	0	none	
	$S^1(2,0)$	0	add to LIS(1)	LIS = {(1,0)**0**,(1,1)**0**},(0,3)**0**}(1,2)**0**,(1,3)**0**,(2,0)**1**}
	$S^1(2,2)$	0	add to LIS(1)	LIS = {(1,0)**0**,(1,1)**0**},(0,3)**0**}(1,2)**0**,(1,3)**0**,(2,0)**1**,(2,2)**1**}
Test $\mathcal{I}$	$\mathcal{S}(\mathcal{I})$	1	split to 3 $\mathcal{S}$'s	
	$S^2(0,4)$	0	add to LIS(2)	LIS = {(1,0)**0**,(1,1)**0**},(0,3)**0**}(1,2)**0**,(1,3)**0**,(2,0)**1**,(2,2)**1**,(0,4)**2**}
	$S^2(4,0)$	1	quad split, add to LIS(1)	LIS = {(1,0)**0**,(1,1)**0**,(0,3)**0**,(1,2)**0**,(1,3)**0**,(2,0)**1**,(2,2)**1**,(4,0)**1**, (4,2)**1**,(6,0)**1**,(6,2)**1**,(0,4)**2**}
	$S^1(4,0)$	0	none	
	$S^1(4,2)$	1	quad split, add to LIS(0)	LIS = {(1,0)**0**,(1,1)**0**,(0,3)**0**,(1,2)**0**,(1,3)**0**,(4,2)**0**,(4,3)**0**,(5,2)**0**, (5,3)**0**,(2,0)**1**,(2,2)**1**,(4,0)**1**,(4,2)**1**,(6,0)**1**,(6,2)**1**,(0,4)**2**}
	(4,2)	0	none	
	(4,3)	1+	move (4,3) to LSP	LSP = {(0,0),(0,1),(0,2),(4,3)} LIS = {(1,0)**0**,(1,1)**0**,(0,3)**0**}(1,2)**0**,(1,3)**0**,(4,2)**0**,(5,2)**0**,(5,3)**0**, (2,0)**1**,(2,2)**1**,(4,0)**1**,(4,2)**1**,(6,0)**1**,(6,2)**1**,(0,4)**2**}
	(5,2)	0	none	
	(5,3)	0	none	
	$S^1(6,0)$	0	none	
	$S^1(6,2)$	0	none	
End $n = 5$ Sorting	$S^2(4,4)$	0	add to LIS(2)	LIS = {(1,0)**0**,(1,1)**0**,(0,3)**0**,(1,2)**0**,(1,3)**0**,(4,2)**0**,(5,2)**0**,(5,3)**0**, (2,0)**1**,(2,2)**1** ,(4,0)**1**, (4,2)**1**,(6,0)**1**,(6,2)**1**,(0,4)**2**, (4,4)**2**} LSP = {(0,0),(0,1),(0,2),(4,3)}

Table 2.7 Example of SPECK Coding of Wavelet Transform, Continued to Bit-Plane $n = 4$

Comment	Point or Set	Output Bits	Action	Control Lists
$n = 4$ Sorting				LIS = {(1,0)**0**,(1,1)**0**,(0,3)**0**,(1,2)**0**,(1,3)**0**,(4,2)**0**,(5,2)**0**,(5,3)**0**, (2,0)**1**,(2,2)**1**,(4,0)**1** , (4,2)**1**,(6,0)**1**,(6,2)**1**,(0,4)**2**, (4,4)**2**} LSP = {(0,0),(0,1),(0,2),(4,3)}
Test LIS(0)	(1,0)	1–	(1,0) to LSP	
	(1,1)	1+	(1,1) to LSP	LIS = {(0,3)**0**,(1,2)**0**,(1,3)**0**,(4,2)**0**,(5,2)**0**,(5,3)**0**,(2,0)**1**, (2,0)**1**,(2,2)**1**,(4,0)**1**, (4,2)**1**,(6,0)**1**,(6,2)**1**,(0,4)**2**, (4,4)**2**} LSP = {(0,0),(0,1),(0,2),(4,3),(1,0),(1,1)}
	(0,3)	0	none	
	(1,2)	0	none	
	(1,3)	0	none	
	(4,2)	0	none	
	(5,2)	0	none	
	(5,3)	0	none	
Test LIS(1)	$S^1(2,0)$	0	none	
	$S^1(2,2)$	0	none	
	$S^1(4,0)$	0	none	
	$S^1(6,0)$	0	none	
	$S^1(6,2)$	0	none	
Test LIS(2)	$S^2(0,4)$	0	none	
	$S^2(4,4)$	0	none	
Refinement	(0,0)	1	decoder adds 2^4	
	(0,1)	0	decoder subtracts 0	
	((0,2)	1	decoder adds 2^4	
	(4,3)	0	decoder adds 0	
End $n = 4$				