working technology) both utilize 8B10B coding and can therefore take advantage of the same serdes devices on the market, despite the fact that their frame structures are completely different.

8B10B codes are broken into two sub-blocks to simplify the encoding/decoding process. Rather than directly mapping 256 data values into 1,024 code words, the data byte is separated into its three most-significant bits (designated Y) and its five least-significant bits (designated XX). The data byte is combined with a flag bit to represent its status as either a normal data character or as a special character. In notational form, a data character is represented as D and a special character as K. As shown in Fig. 9.7, the eight data bits are assigned letters from H down to A and the sub-blocks are swapped to yield a final notation of D/KXX.Y. For example, the ASCII character Z (0x5A) is split into two sub-blocks, 010 (Y) and 11010 (XX), yielding a notation of D26.2.

Once the data or special character has been split into sub-blocks, each sub-block is encoded using separate lookup tables whose inputs are the sub-blocks, CRD state information, and the special character flag. Separating the sub-blocks enables smaller lookup tables, which is advantageous from complexity and timing perspectives. The five-bit (XX) sub-block is converted to a six-bit code, and the three-bit sub-block (Y) is converted to a four-bit code, making ten bits in total. When encoding the XX sub-block, the CRD bit reflects the running disparity remaining from the previously encoded character. In the case of the Y sub-block, the CRD bit reflects the running disparity remaining from the 5B6B encoding of the current character. Tables 9.4 and 9.5 list the 5B6B and 3B4B lookup functions. Note that not all special characters are valid and that the CRD bit inverts the code when necessary to maintain a balance of 0s and 1s. The encoded bits are referred to by lower-case letters and, when sub-blocks are combined, form a string of bits: a, b, c, d, e, i, f, g, h, j. Encoded words are transmitted starting with bit "a" and ending with bit "j".

A special case exists in the 3B4B lookup table when encoding Dxx.7 characters to prevent long strings of consecutive zeroes or ones. When the CRD is negative and D17.7, D18.7, or D20.7 are being encoded, the alternate 0111 encoding is used instead of 1110. Likewise, when the CRD is positive and D11.7, D13.7, or D14.7 are being encoded, the alternate 1000 encoding is used instead of 0001. These exception cases present some additional complexity to the 3B4B translation.

In addition to the 256 possible data characters, 12 special characters are supported. All possible K28.y characters are supported along with K23.7, K27.7, K29.7 and K30.7. These special characters are identified by either a unique 5B6B code or a unique 3B4B code. The K28 is the only special character type with a unique 5B6B encoding. The other special characters are identified, because Kxx.7 has a unique 3B4B encoding. Recall that not all ten-bit code word permutations can be used because of 0/1 disparity rules inherent in the 8B10B coding algorithm.

There are numerous possible implementations of 8B10B encoder and decoder circuits, and they can vary by how many bytes are simultaneously processed, the type of lookup table resources available, and the degree of pipelining or other optimization required to achieve the desired throughput. A high-level block diagram of an 8B10B encoder is shown in Fig. 9.8. The larger 5B6B lookup table computes the intermediate CRD´ state that results from mapping the running disparity of the previ-

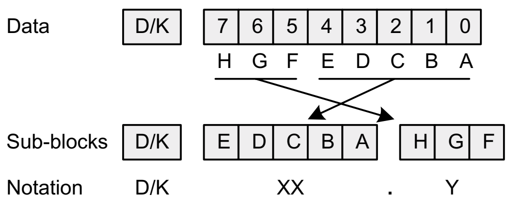

FIGURE 9.7 8B10B coding notation.

TABLE 9.4 5B6B Sub-block Encoding

Input Character	Binary Value EDCBA	Encoded Value abcdei	
		Positive Disparity	Negative Disparity
D00.y	00000	100111	011000
D01.y	00001	011101	100010
D02.y	00010	101101	010010
D03.y	00011	110001	
D04.y	00100	110101	001010
D05.y	00101	101001	
D06.y	00110	011001	
D07.y	00111	111000	000111
D08.y	01000	111001	000110
D09.y	01001	100101	
D10.y	01010	010101	
D11.y	01011	110100	
D12.y	01100	001101	
D13.y	01101	101100	
D14.y	01110	011100	
D15.y	01111	010111	101000
D16.y	10000	011011	100100
D17.y	10001	100011	
D18.y	10010	010011	
D19.y	10011	110010	
D20.y	10100	001011	
D21.y	10101	101010	
D22.y	10110	011010	
D/K23.y	10111	111010	000101
D24.y	11000	110011	001100
D25.y	11001	100110	
D26.y	11010	010110	
D/K27.y	11011	110110	001001
D28.y	11100	001110	
K28.y	11100	001111	110000
D/K29.y	11101	101110	010001
D/K30.y	11110	011110	100001
D31.y	11111	101011	010100