Against All Odds

'Even parity' ensures that the number of 1 bits in a byte is always even. This makes transmission errors easier to detect

One of the main advantages of digital computers over analogue devices is that the errors and inaccuracies that occur in all electrical circuits do not accumulate as a signal is passed through many circuits (see page 239). However, when data is transmitted over any distance — whether by means of a serial interface and a pair of wires, or over a telephone line — the background electrical 'noise' in the line can sometimes be enough to flip a single bit from 0 to 1, or vice versa. Normally, the receiving computer would have no way of knowing that this had happened, and would accept the erroneous data as being correct.

Look at what happens if one bit in the ASCII code for the letter Q becomes corrupted:

[ ] 1 0 1 0 0 0 1 (Transmitted ASCII code for Q)
[ ] 1 0 0 0 0 0 1 (Received ASCII code for A)

An error such as this in the transmission of data would, at the least, be a nuisance and could be potentially catastrophic. However, you will remember that ASCII codes are assigned only to values up to 127, which requires only seven bits (numbered 0 to 6). The Most Significant Bit (bit seven) is therefore often used as a 'parity' bit, to detect when an error has occurred.

There are two conventions for using parity bits: 'even parity' and 'odd parity'. We shall consider the former. 'Even parity' means that the parity bit (bit seven in an ASCII code) is set so that the total number of 1 bits in the byte is always an even number. Here's how the letters A and Q would look with even parity:

[0] 1 0 0 0 0 0 1
(the ASCII code for A with even parity)
[1] 1 0 1 0 0 0 1
(the ASCII code for Q with even parity)

There are two 1 bits in the ASCII code for A, so the parity bit is made 0 so that the total of all eight bits is even. In the ASCII code for Q, there are three 1 bits, so the parity bit is made a 1. This brings the total number of 1 bits to four, which is an even number.

Now let's see what would happen if bit four in our ASCII letter Q became corrupted as in the example above.

[1] 1 0 0 0 0 0 1 (corrupted ASCII Q)

When the parity of the byte is checked (either by software or by special hardware) it is seen that the correct Q has an even number of 1s in it (including the parity bit). The corrupted Q, by contrast, accidentally had bit four changed from a 1 to a 0, but the original parity bit — bit seven — is still a 1. When the parity of this corrupted byte is checked, it will be found to have an odd number of 1 bits, and so this byte is known to be corrupted and can be rejected. If you think about it, you will see that even if the parity bit itself were to become corrupted in transmission, the fact that an error had occurred would still be picked up by the parity checking process, and the byte would be rejected.

If you look at the ASCII codes used in your own computer, you will probably find that bit seven (the Most Significant Bit, or MSB) is in fact used, but not as a parity bit. This is done to enable the computer to have an additional character set (usually a set of graphics characters), and because errors in data transmission *inside* a computer are very rare. Parity is normally used only when transmitting data over long distances, or when recording data onto a magnetic recording surface (such as tape or disk) which is equally susceptible to 'bit errors'.

Parity checking is fine for indicating that a given byte has been transmitted incorrectly, but it does not indicate which bit in the byte was wrongly transmitted, so the error cannot be corrected by the receiving computer. Worse still, if two bits in a byte become corrupted, an incorrectly transmitted byte could be taken as a correct one.

But in cases where the receiving device detects an error, it can send back an error message and the software can arrange for the incorrect byte to be transmitted again. More sophisticated error detecting and correcting schemes have been devised that can pin-point which bit or bits were in error, enabling them to be corrected automatically. Error correcting codes are a subject that will be discussed later in the course.

Just Checking
The last digit in an International Standard Book Number (ISBN) is a check digit — equivalent to parity in a computer. Multiply the first digit (0 here) by 10, the second (5) by 9, and so on, then add the results together. You will find that the check digit has been set such that the result is exactly divisible by 11