Message Format

Each message consists of two blocks of data.

(1) The address block - a block containing the address of a single data word.

(2) The data block - a block containing the condition of that word.

Each block consists of 11 bits allocated as follows:

(a) One start bit (the leading bit), marking the start of the block.
(b) Eight bits constituting the message.
(c) One parity bit.
(d) One stop bit, marking the end of the block.

As is conventional in serial transmission, the least significant bit of the message follows the start bit, the rest of the message then follows in increasing significance, and the parity bit follows the most significant bit.

The address and data blocks are differentiated by whether the most significant bit of the eight bits constituting the message is set high or low.

High marks the block as an address block.

Low marks the block as a data block.

In the data block the remaining seven bits of the message represent the condition of the data word.

In the address block the most significant of the remaining seven bits is always set high (for reasons associated with the internal operation of the control card), and the remaining six bits represent the address of the data word.

There is a fixed time period (known as a window) following the reception of the address block during which the control card will accept the data block.

If a demultiplexing control card receives a message which does not conform to this format, the message is rejected.

(You may be puzzled when I write of one bit in the message being more "significant" than another. Considered as bits of data this is, I agree, non-sensical: however when I use the term here, I am, for convenience in the description, treating the message as a binary number - see the Glossary: **binary**.)

The Use of Parity

You may know that it is possible effectively to guarantee the integrity of a transmitted message through a technique known as "redundancy". This technique allows the receiving equipment to detect and, given a sufficient degree of redundancy, even to correct, errors in transmission. Without going into detail, the technique consists of adding extra (or "redundant") bits to the message. These inter-relate to the message and to each other in such a way that the transmission as a whole can be processed to extract the message, despite extensive corruption.

The drawback is that the more effective a redundancy scheme is, the more bits it uses and the more processing power is required to encode and decode the message. This is particularly significant in a small remote control system such as the DAM since the extra bits would seriously increase the time required for the background scan, and the encoding/decoding of the message would overload the Control card processor. Given that line safety relies ultimately on the remote interlocking, and hence **the remote control is a non-vital system**, the use of extensive redundancy in this application is not justified.

You will see from the message format that one redundant bit, the **parity** bit has been included. This gives the system the ability to detect any **odd number** of bit errors in the transmission.

In the DAM system, message parity has an additional use, which is to **mark the direction of transmission**.

Odd parity is used for a transmission from office to field **(controls).**

Even parity is used for a transmission from field to office **(indications).**

This has been included as a precaution against crosstalk between the two channels. (Any "controls" message induced on the "indications" channel will be rejected, and vice-versa.)