

BLOCK CODING & DECODING

PREPARATION	60
block coding	60
PCM encoded data format	60
block code format	61
block code select	62
typical usage	63
block decoding	63
EXPERIMENT	64
encoding	64
decoding	65
conclusion	66
TUTORIAL QUESTIONS	66
APPENDIX.....	67
automatic frame synchronization	67

BLOCK CODING & DECODING

ACHIEVEMENTS: viewing of a serial data stream before and after block encoding. Decoding. SNR improvement due to block coding.

PREREQUISITES: completion of the experiment entitled *PCM encoding* in Volume D1.

ADVANCED MODULES: PCM ENCODER, BLOCK CODE ENCODER, BLOCK CODE DECODER, LINE-CODE ENCODER.

PREPARATION

block coding

This experiment examines the BLOCK CODE ENCODER and BLOCK CODE DECODER modules.

Block coding refers to the technique of adding extra bits to a digital word in order to improve the reliability of transmission. The word consists of the message bits (often called information, or data) *plus* code bits. It may also, as in the present case, contain a frame synchronization bit.

A block code adds bits to existing message bits, or blocks, *independently* of adjacent blocks¹.

In this experiment the blocks will be prepared by the PCM ENCODER module. These blocks were examined in the experiment entitled *PCM encoding*.

PCM encoded data format

When extra code bits are added to a PCM word (initially containing only message bits) then the word will get longer. If the bit rate remained the same then the message bits would arrive at a slower rate than before. To maintain the same message rate the bit rate would need to be increased. This would require an increased transmission bandwidth.

In the TMS PCM ENCODER module a different scenario has been adopted.

¹ instead of being distributed over a number of blocks, as, for example, in a convolutional code.

The PCM word has been generated from the input message and placed in a frame of fixed length. These are the *message bits*. Not all slots in the frame are used. When extra *coding bits* are added, they go in the previously unused slots. Thus, in either case (extra code bits or not):

- the frame length remains the same
- the message rate remains the same
- the channel bandwidth will remain the same, as the bit rate has not changed

The TMS arrangement may waste time in the un-block-coded state (there are three unused slots in the frame), and so be called inefficient (which it is). But it is convenient for our purpose.

The PCM ENCODER module was examined in the experiment entitled *PCM encoding* in volume D1.

block code format

The BLOCK CODE ENCODER module is designed to expect input blocks of length eight slots, where some of these slots are empty. These come from the PCM ENCODER module (in the 4-bit mode).

The incoming data frame is illustrated in Figure 1 below.

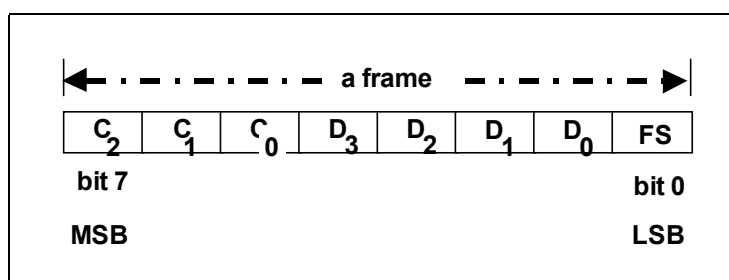


Figure 1: a data frame of eight slots, one per clock period

The message bits are shown as D_3 , D_2 , D_1 , and D_0 , where D_3 is the most significant bit of the message.

The frame synchronization bit is shown as FS.

The slots marked C_2 , C_1 , and C_0 will be used by the BLOCK ENCODER for code bits.

For the BLOCK CODE ENCODER module to function correctly it must always receive three digital signals:

1. TTL binary data in an 8-bit wide frame (typically from a PCM ENCODER in 4-bit mode). The data must occupy frames 4, 3, 2, and 1 (as defined in Figure 1 above).
2. a TTL clock, to which the incoming data is synchronized. Typically this will be at 2.083 kHz (the module is restricted to a clock rate below 8 kHz).
3. a TTL frame synchronization signal FS, which signals the *end* of the frame.

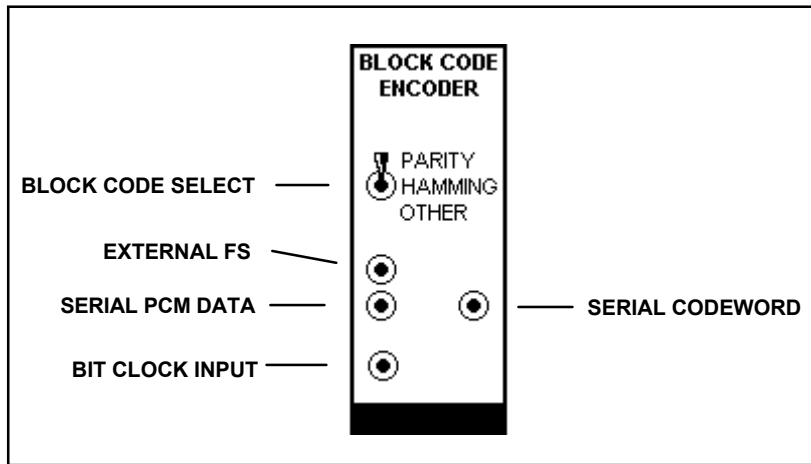


Figure 2: front panel layout - ENCODER

The front panel of the module is illustrated in Figure 2 above. The features should be self explanatory, except for the BLOCK CODE SELECT toggle switch.

block code select

Each BLOCK CODE ENCODER module offers three different coding schemes. These are contained in an EPROM. More than one EPROM is available, any one of which can be installed in the module. The codes they offer are set out in Table 1 below.

EPROM	code 1	code 2	code 3
BLKe1.x	even parity - single bit error detect	Hamming (7,4) - single bit error correct	*Setup - with C _x bit error detect
BLKe2.x	even parity - single bit error detect	Hamming (7,4) - single bit error correct	odd parity - single bit error detect
BLKe3.x	even parity - single bit error detect	Hamming (7,4) - single bit error correct	Cyclic

Table 1: EPROM codes

Any one of the three codes in the installed EPROM can be selected with the front panel toggle switch.

In performing parity checks the FS bit is ignored.

typical usage

In a typical digital communications system, the configuration at the transmitter might appear as in the block diagram of Figure 3 below.

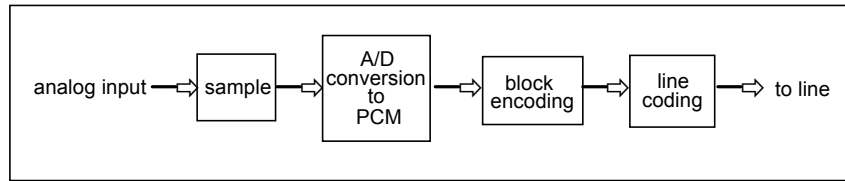


Figure 3: disposition of the block encoder

block decoding

The signals from the BLOCK CODE ENCODER need to be interpreted by a complementary BLOCK CODE DECODER module, the front panel of which is illustrated in Figure 4 below.

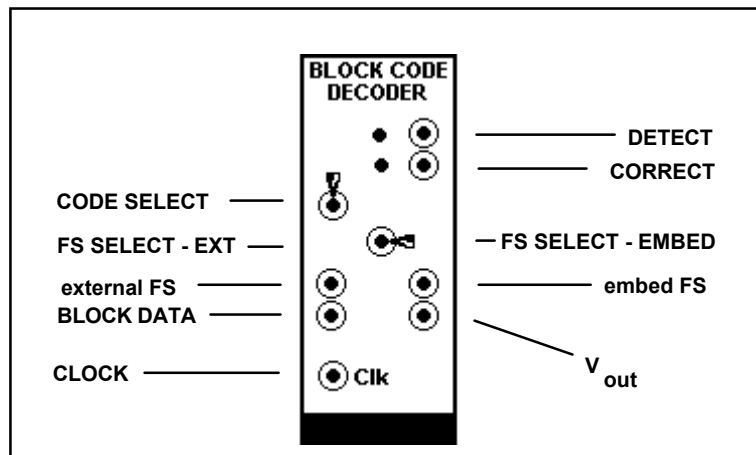


Figure 4: front panel layout - DECODER

The front panel of the decoder module is illustrated in Figure 4 above. The features should be self explanatory, except for the following:

- **DETECT**: for codes which can detect but not correct errors. The LED flashes when an error is detected, but not corrected. There is a TTL high, one bit wide, at the adjacent socket, during the frame in which the error occurred.
- **CORRECT**: for codes which can detect and correct errors. The LED flashes when an error is detected and corrected. There is a TTL high, one bit wide, at the adjacent socket, during the frame in which the error occurred.

The DETECT and CORRECT outputs (LED and bit-wide TTL HI) are mutually exclusive.

- FS SELECT - EXT: frame synchronization may be attained by accepting a 'stolen' FS signal from the transmitter, patched to the FS input socket.
- FS SELECT - EMBED: frame synchronization may be achieved automatically, using the embedded information in the LSB of the frame itself. For verification the recovered FS signal is available at the FS output socket. When a stolen FS signal is used there is *no output* from this socket.

EXPERIMENT

This experiment is intended to help familiarize you with some aspects of the operation of the BLOCK CODE ENCODER. It will also confirm the decoding process performed by the BLOCK CODE DECODER module. It is a necessary preliminary to the experiment entitled *Block coding and coding gain* of this Volume.

encoding

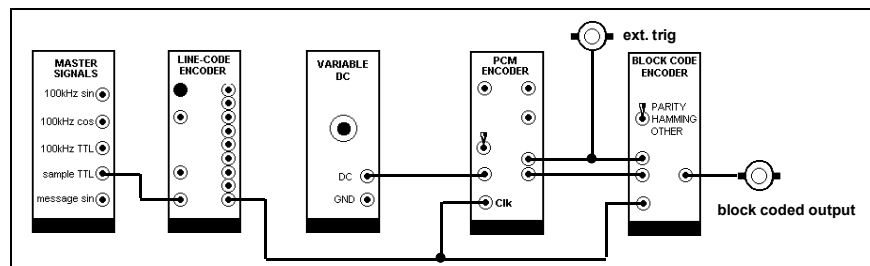


Figure 5: block code encoding

The BLOCK CODE ENCODER requires a TTL clock near 2 kHz. The *TIMS Advanced Modules User Manual* says it must be operated at a clock speed below 8 kHz.

You may have noticed that it is customary TIMS practice (but not mandatory) to use a clock locked to the MASTER 100 kHz source. Typically this has been the 8.333 kHz TTL signal from the MASTER SIGNALS module. Since the BLOCK CODE ENCODER requires something lower than this, a convenient source is obtained by dividing this by 4. The LINE-CODE ENCODER module has just such a divider² (and typically forms part of a data transmission system). The model of Figure 5 above illustrates this method.

For stable oscilloscope displays from the PCM source a DC message is used, together with a suitable source of external triggering signal.

T1 patch up the model of Figure 5.

² the DIGITAL UTILITIES module, and the BIT CLOCK REGEN module also have divide-by-4 sub systems

T2 set up simultaneous displays of the PCM input, and the block coded output, of the BLOCK CODE ENCODER, over two or three frames. Spend some time investigating different methods of oscilloscope synchronization. Accepting jittering displays is unprofessional! See Tutorial Question Q2.

T3 verify, where possible, that each of the codes has been implemented correctly.

decoding

Having successfully block encoded a PCM signal, it is time to demonstrate its decoding. For this purpose transmission will be via a direct connection.

You will have noticed the ERROR INDICATION front panel LEDs on the decoder. These will be useful when transmission via a noisy, bandlimited channel, with the inclusion of line encoding, is examined in a later experiment. There the benefits of block coding will be demonstrated and evaluated.

Patching for the decoding process is shown in Figure 6.

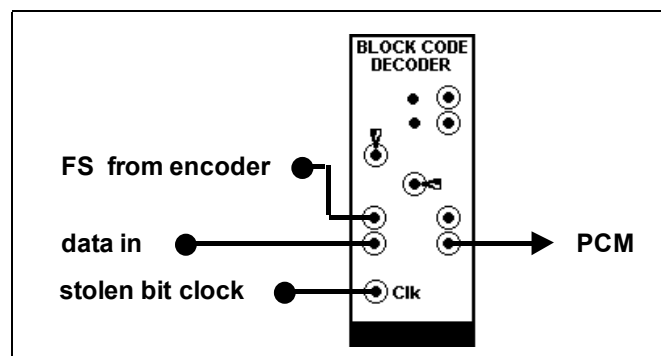


Figure 6: block code decoding

Note that a stolen bit clock is used.

Frame synchronization can be achieved by either a 'stolen' FS signal from the encoder, or by internal decoding of the alternating pattern of 1 - 0 - 1 - 0 - 1 embedded as the LSB of the PCM code word (in location '0' of the frame). This scheme was introduced in the experiment entitled *PCM decoding*.

T4 patch up the BLOCK CODE DECODER according to Figure 6. This uses the 'stolen' frame synchronization signal FS from the transmitter, connected to the EXT. FS input, and selected with the front panel toggle switch FS SELECT.

T5 verify that successful decoding back to the original PCM is possible for all codes.

T6 switch the front panel toggle switch FS SELECT to EMBED. Confirm that the internal circuitry for extracting the frame synchronization signal FS from the PCM signal itself is operating correctly. Refer to the Appendix to this experiment for more information.

conclusion

You are now in a position to include block coding in a more complex transmission system (noisy, bandlimited) and to demonstrate its effectiveness in improving the bit error rate. This is the subject of the experiment entitled *Block coding and coding gain*.

During this experiment you should have developed techniques for obtaining oscilloscope displays which show you what you want, without need to constantly adjust and re-adjust the oscilloscope controls. Choice of the appropriate trigger signal for each display is important.

Although, for a DC message, each 4-bit word and added code bits are the same, the alternating pattern of 0-1-0-1- for the FS signal make alternate frames *different*. It would be preferred if the synchronisation technique adopted would always put the same frame first, no matter what the sweep speed.

This is a simple matter to implement. See Tutorial Question Q2.

TUTORIAL QUESTIONS

Q1 when adding check bits for parity checking, the bits of the alternating pattern 1 - 0 - 1 - 0 - for frame synchronization in the LSB position were ignored. Explain.

Q2 explain how dividing the frame synchronization signal by two is often a help in obtaining and maintaining stable, and repeatable, oscilloscope displays in the context of this experiment.

APPENDIX

automatic frame synchronization

The BLOCK CODE DECODER module has built-in circuitry for locating the position of each frame in the serial data stream. The circuitry looks for the embedded and alternating '0' and '1' every 8 bits (which occur in the LSB position of each frame).

The search is made by examining a section of data whose length is a multiple of eight bits.

The length of this section can be changed by the on-board switch SW3. Under noisy conditions it is advantageous to use longer lengths.

The switch settings are listed in Table A-1 below.

left toggle	right toggle	groups of eight bits
UP	UP	4
UP	DOWN	8
DOWN	UP	16
DOWN	DOWN	32

Table A-1: synchronization search length options

