Griffith

PROJECT STRETCH
DELTA COMPUTER MEMO NO. 19

November 29, 1956

Subject: Code Translation Using Streaming

By: H. C. Montgomery

Summary:

A general problem of code conversion is discussed in terms of a somewhat new approach which assumes a machine having data processing facilities more sophisticated than machines currently in use.

Given:

A group of N characters, expressed in a model code M using $\underline{m}$ bits per character, is stored in consecutive memory words starting at R_r. The image portion of a conversion table relating M to another code I, which uses $\underline{n}$ bits per character, is stored in consecutive words starting at T. These image characters are stored using an eight bit byte for each character stored (for all image codes), with the $\underline{n}$ rightmost bit positions of the eight bit image group being used. The remaining bits of the eight bit byte are blanks or some other arbitrary array of zeros and ones.

The Problem is to convert these N characters of code M into N characters of Code I and store the results in consecutive words starting at S_s. (S_s may be the same as R_r if $N \leq m$).

An instance in which this type problem occurs is that in which one reads into the Delta computer data which are in the 705 code and wishes to convert these data into the code used internally by Delta. A specific example is treated in some detail in the appendix.

A Solution

1. The General Approach

The approach to be described consists of a table look-up procedure combined with the facility of streaming the data through the registers being used. The basic unit of data on which operations are performed is the byte. Peculiar to this approach, however, is the fact that the size of the bytes used varies from one register to another, but for a given register remains fixed throughout the problem. This is convenient because we can now choose byte sizes for the registers to be compatible with the codes of the data they use.

2. How It Works

The zero-one array of the model character is used to determine both the word address and byte location within the word of the image character to which it corresponds. The image character is then brought from memory to be inserted into a specified byte position of a register which develops the image data in word lengths and then sends such words to memory. The actual conversion, therefore, is quite simple. The somewhat more complex housekeeping operations involved will now be examined in more detail.

3. Locating a Byte Image

The image character corresponding to the model character being converted during a given byte cycle is stored in memory, as is given above, and is located in the following manner. The three low order bits of the model byte are sent to the controls of one of the data registers, say C. The remaining bits of the model byte are added to the constant T to obtain a sum which is the address of the word containing the image character being sought. (See Table 1). This word is sent to C, whose controls now use the three bits previously received to

No. of Bits Per Model Byte	No. of Bits For Byte Address	No. of Bits for Word Address
3	3	0
4	3	1
5	3	2
6	3	3
7	3	4
8	3	5

TABLE 1

How the Model Byte is Divided to Determine The Address of Its Image Byte

choose one of the eight bytes of the word it now contains. This byte, which is the image character we have been looking for, is then sent to another data register, say D, which trims off the (8-n) superfluous bits and enters it in the byte position specified by D's controls. The table entry now in register C remains there until the word required by the next model byte is determined, because it may be the same word.

Observe that this trimming facility of the controls of register D can be used to effect those conversions which consist only of eliminating a prefix from the byte of model data.

4. Streaming Control

Since the number of characters to be converted is known beforehand, a counter can be used to control the stopping of the program. The integer N, which is the number of model characters, is entered into a counter during initialization. Then as each character is processed this counter is reduced by one. When the count has been reduced to zero, the counter causes the conversion to end.

5. Problems of Disparity in the Byte Sizes of Codes

Cases arise in which the number of bits per character in the model code differs from the corresponding number for the image code. Since we have chosen the byte as the basic unit of data on which the conversion operations are to be performed, it may appear that byte size differences present considerable difficulty in devising a workable system.

Delta Memo No. 5 describes the controls to be associated with the data registers, and it is these controls which provide the means for overcoming the anticipated difficulties. These controls afford the degree of flexibility needed for operating with bytes of different size during the conversion process, because each register can be set independently to the byte size of the data it uses throughout the conversion. Hence the register through which the model data flow can be set to a byte size of m bits and the register in which the result data are developed can be set to a byte size of n bits.

The bit address counter specifies a particular byte of its parent register and is incremented by the number in the byte size register at sometime during each byte processing cycle. This incrementing continues to the end of the procedure.

An indicator of the bit address counter could be used to signal that the word address counter is to be incremented by one, thereby developing as it is needed the address of the next word with which the parent register will communicate.

6. Maintaining Continuous Flow

To avoid delays caused by the necessity of replenishing the supply of model data for some registers and removing result data from others, a streaming facility will be provided.

Consider the register through which the model data pass; call it B. Initially B receives a complete word from memory. As the bytes of model data are processed, the bit address counter of B advances m places for each byte. When this counter advances past 31, one knows that the first half of the contents of B has been translated. At this point, the first half of the next model word is brought in to fill bit positions 0 through 31 of B. Similarly, when the bit address counter of B passes the bit address of the last byte in B, one knows that the conversion of the second half of the original contents of B has been completed, so the second half of the next model word is inserted in bit positions 32 through 63 of B.

In both cases above, however, the streaming control counter determines whether the end of the model data stream lies within the next half word. When affirmative, the model data procuring procedure is stopped.

It is interesting to observe that this arrangement avoids the difficulty, in which some bytes begin at the end of one word and end in the next, which arises when the word length is not an integral multiple of the byte length being used.

But the outstanding characteristic of this arrangement is that the model data appear to come from a continuous source. That is, the operations proceed in uninterrupted fashion from byte to byte until the model data are exhausted, seldom having to hesitate while a new word is being brought from memory.

Although the model data register has been used to describe this feature, it is not limited to just one register. The same problem arises in the image data register, and it is to be understood that the situation is treated in an analogous manner.

7) Recapitulation: What Takes Place When a Byte is Translated

a) The word whose address is given by the word address control of B is entered in B.

b) From the byte of B specified by its bit address counter, the three low order bits are sent to the bit address counter of C. The remaining bits are added to T and the sum so obtained is sent to the memory address register.

c) The word whose address is in the memory address register is inserted in C.

d) The byte of C specified by its bit address counter is sent to the byte position of D specified by the bit address counter of D. The byte from C is entered in D after having any superfluous bits trimmed off.

3) If the streaming control counter displays a continue attitude, the bit address counters of B and D are increased by one, and the next byte is converted.

A special Stream Translate instruction will be provided to initiate this type of execution once the preparatory machine conditions have been set.

APPENDIX

Example

Beginning at bit position 23 of the word whose address is 536 there are stored 87 characters expressed in the 705 code. The first few characters of these 87 are 1111000, 1000010, 1100110,... . It is desired to translate these characters into the proposed Delta Computer internal code (see Delta Memo No. 12) and to store the results in consecutive memory words beginning at the 13th bit position of the word whose address is 231.

For convenience, the necessary part of the table relating the two codes involved is given here.

Familiar Character	C	B	A	8	4	2	1
⋮							
H	x	x	x y	x	y	y	y
⋮							
⊙	x	y	y		y	x	y
⋮							
2	x	x y			x	x	y

Read x as a 1 in the 705 code and read y as a 1 in the Delta code. In both cases, read blanks as zeros. The Delta code portion of this table is stored in a block of memory starting at location 175.

Assume that there are four data registers; call them A, B, C and D.

Initialization

The following preparatory settings are made.

1. The byte size control of B is set to 7 bits.
   The byte size control of C is set to 8 bits.
   The byte size control of D is set to 6 bits.

2. The field length counter of B is set to 87 bytes.

3. The word address control of B is set to 536.
   The word address control of D is set to 231.

4. The bit address counter of B is set to take the first byte starting at bit position 23.
   The bit address counter of D is set to take the first byte starting at bit position 13.

5. The constant 175 is put in register A.

<u>Execution</u>

When the Stream Translate instruction is given, the following operations take place automatically:

1. The word in location 536 is put in register B.

2. A 7 bit byte of B is selected, starting at bit position 23. This byte is 1111000.

3. a) The three low order bits of this byte are sent to the bit address counter of register C.
   b) The remaining bits (1111) are added to the contents of register A and the sum sent to the memory address register.

4. The word whose address is $\left[(175)_{10}+(1111)\right]$ is put in register C.

5. The byte of C specified by its bit address counter is sent to the bit positions of D specified by D's bit address counter. The controls of D trim off the two leftmost bits of this byte before it is entered in D.

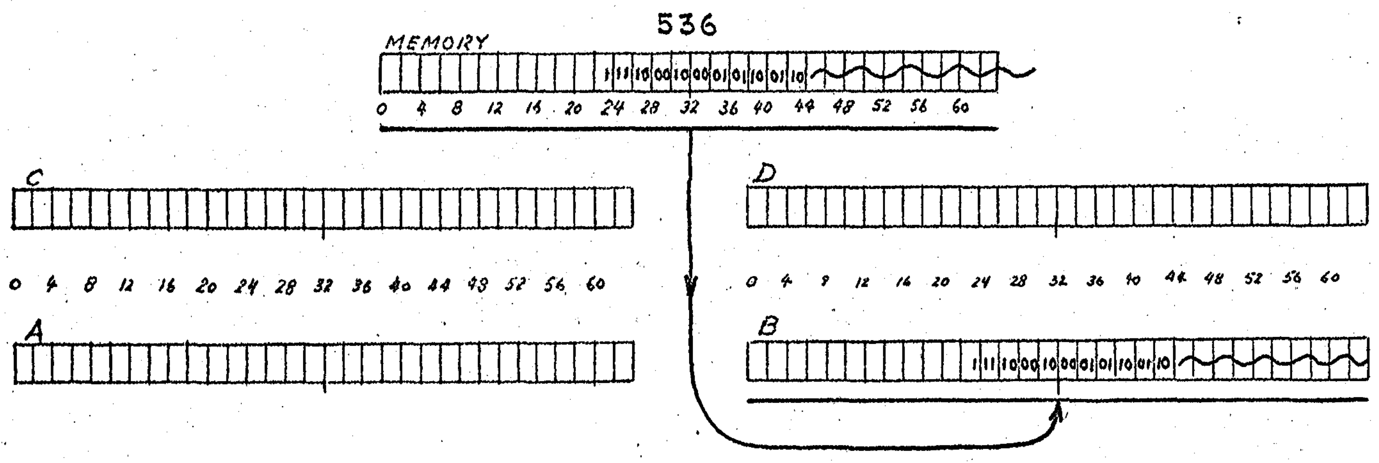
6. The field length counter of B is reduced by one.

7. If B's field length counter is now zero, the next instruction is executed. If this counter is not zero, the bit address counters of B and D are advanced to specify their next byte positions and the preceding operations are repeated.

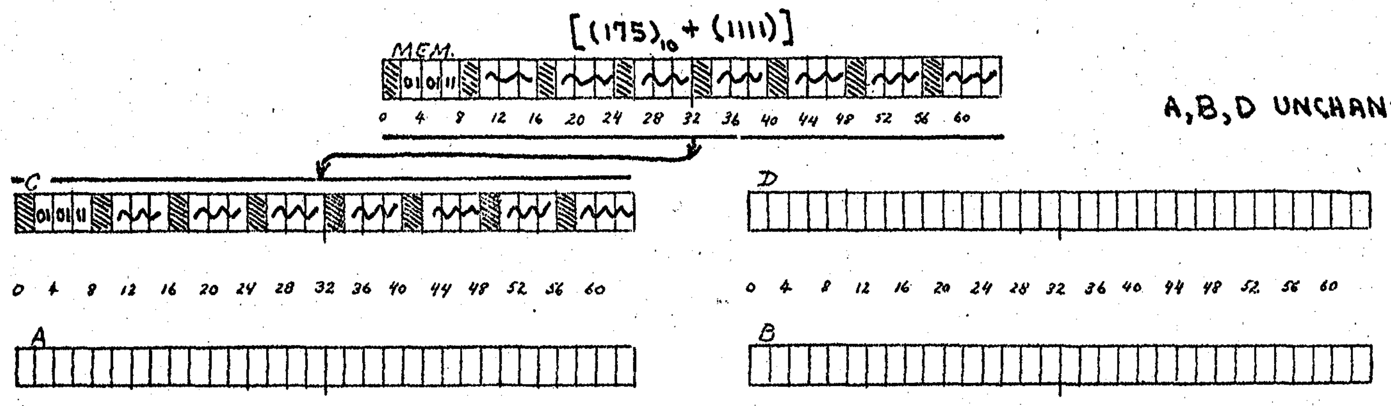
(A sheet is attached which displays the contents of the data registers in the key steps given above).

EXECUTION STEP

NOTE: ∿∿ MEANS PERTINENT DATA NOT OF IMMEDIATE INTEREST
▨ MEANS NON-SIGNIFICANT DATA

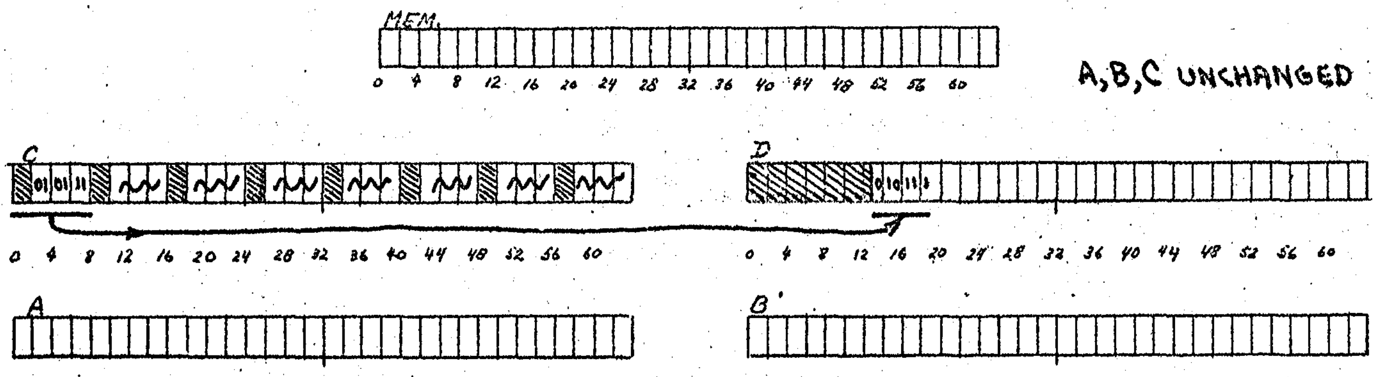


4.

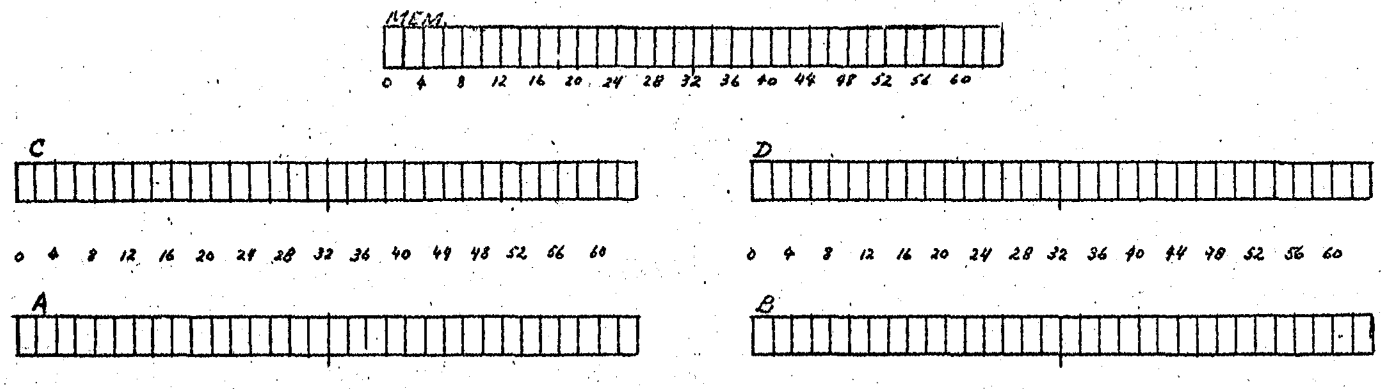


A,B,D UNCHANGED

5.



A,B,C UNCHANGED



11/28/56