Harvard

To : Distribution January 4, 1957

From' : William J. Worlton

Subject : Addressing considerations for STRETCH

Symbol : T-1

PURPOSE

There has been some discussion around Los Alamos that the number of bits in the address and tag fields is excessive; however, no concrete proposal has been made for using these bits for other purposes. If, for normal addressing, the number of bits proposed is excessive and no other valid use is proposed, it may be that efficient use of these bits may be made if a more generalized form of addressing is employed. The following paper assumes that there will be forty five bits in the address and tag fields, and investigates the possibility of using them for different addressing forms.

ASSUMPTIONS

1. Assume three tags, T_1, T_2, T_3; these tags to be used for the following purposes:
   a) An address, A, of an operand to operate on the accumulator.
   b) An address, I, of an index register.
   c) An address, P, of an operand to be pre-loaded.
   d) An address, Pr, of an operand to be pre-stored.
   e) An address, Po, where a result will be post-stored.
2. Assume the following instruction format:

	OP	α	ADDRESS & TAGS
1	12	6	45

   where the quantity "α" specifies the use of the address field.
3. Initially it will be assumed that any one of the addresses may be direct or indirect.

4. At least two levels of indexing will be assumed.

ADDRESSING COMBINATIONS

Although there are actually 216 ways in which samples of size three can be drawn from a population of 6 (besides the five addresses indicated, "*" will be used to indicate that the field is not used), the non-repetitious and meaningful combinations number only 20.

The first thing which becomes evident in these addressing combinations is that P, Pr, and Po, all assume that an address A exists; I may operate on an address other than A, but then the assumption still exists that an A address exists, and only combinations in which an A occurs will be considered.

TABLE I

		T_3 →	A	I	P	Pr	Po	*
T_1	T_2							
A	A		i	i	i	i	i	i
A	I		r	i	i	i	i	i
A	P		r	r	x	i	i	i
A	Pr		r	r	r	i	i	i
A	Po		r	r	r	r	i	i
A	*		r	r	r	r	r	i

where: i indicates that this combination will be investigated.
r indicates that this combination is repetitious.
x indicates that this combination is not really meaningful.

SET I: MULTIPLE ADDRESS OPERATIONS

Inherent in the P, Pr and Po combinations above is the assumption that some two and possibly even three address work will be available; this

assumption has furthermore been stated by both IBM and Los Alamos; however, the extent to which this can profitably be carried needs some discussion. Consider, for example the following set of address combinations.

Set I:
1. A A A
2. A A I
3. A A P
4. A A Pr
5. A A Po
6. A A *

All of these combinations imply multiple address operations. Operations of this type have both advantages and disadvantages, and it may be well to list some of these. Advantages are:

a) speed: the arithmetic unit will operate with a greater effieiency because more operands become available for each instruction processed. Timing diagrams verify this; arithmetic efficiency is sometimes promoted to 100%.

b) decreased number of instructions: keypunching for input is thereby decreased; the memory occupied by the program is decreased; the automatic programming routine will probably benefit because of the more generalized instructions (i.e., it will probably be easier to write an efficient automatic programming routine).

Disadvantages of multiple address operations include:

a) hardware: the decoder would be more complex because (1) the number of operations would increase, (2) the number of addresses to be processed, and therefore the number of registers, would increase.

b) complexity of use: the optimum programming spectre raises its head, i.e., for a multiple operation instruction to take full advantage of its possibilities, the memory references would need to be appropriately arranged. However, the automatic programming routine may be of real value in keeping this problem to a minimum.

Some background exists at Los Alamos for judging the merits of three address programming because of the experience with the CPC #1 Board and the 701 SHACO system. With such a large addressing field available on STRETCH, serious consideration should be given to multiple address work.

Bit allocations for the multiple address work might be (15,15,15) in the case of the (AAA) combination, and perhaps (18,18,9) in the (AAI) or other cases where temporary storage was involved. In the combination (AA*), where the T_3 field is not used, the two fields used could be expanded to allow for more directly addressable memory, say (21,21,-).

SET II: MULTIPLE INDEXING

The (AII) combination has obvious value in two-dimensional work and should definitely be included on the machine. Further dimensional indexing may be allowed for by allowing T_2 to specify the location of the tags involved, and T_3 to specify which tags are operative. In this fashion one obtains maximum versatility with a minimum number of tag words. Since the T_3 field in this case needs only the same number of bits as the maximum number of tags in a single word, the bit allocation might be (20,20,5), with five 12-bit tags in the tag word. With six 10-bit tags, one could have (20,19,6) and with seven 9-bit tags (20,18,7) or (19,19,7).

SET III: PRE-LOADING, PRE- AND POST-STORING

Combinations involving P, Pr and Po have a great value in eliminating operations which store intermediate results in temporary. A preliminary investigation indicated that something like 25% of the numerical work in formula calculations could be saved if operations were available which automatically used the P, Pr and Po concepts. There is a speed and efficiency advantage similar to that mentioned above for the multiple address operations.

SET IV: UNUSED FIELDS

The (A**) combination, or any other combination where a field is not used, provides the possibility that this field might be used to advantage by the programmer for such things as Complex Arithmetic, Double Precision Arithmetic, etc.

Check-out of a problem might be facilitated by having a "debugging mode" in which otherwise unused fields would contain a transfer address. In normal operation, this field would be inoperative, but during checkout, the field could specify the location of a check routine to which an automatic transfer would be made.