

results since the logical instruction is not fixed, i.e., bound, until execution time and at that time incorporates the data field descriptors which account for the encoding differences.

Once the control store unit has determined that the data types are compatible and that a transformation may occur, step 432 of the flow chart is executed. Step 432 is a general purpose purely logical instruction which may be an add, subtract, multiply, divide, move, compare and/or hash operation. This logical instruction is performed on the total information identified by the instruction, i.e., each field has a descriptor and there may be one to three fields depending on the type of instruction. When the instruction is executed, the data field descriptors automatically account for the variations in the data fields. Previously, this function had to be performed by a specialized subroutine written for each individual data type.

Upon completion of the instruction in step 432, the next instruction in the instruction fetch unit 308 is sequenced. This completes operation on the specific operand of the data base file.

The automatic accounting feature utilized by the logical instruction includes a translation to compatible formats of the data fields. This is easily accomplished through programmable read only memories (PROMS). In addition, these PROMS are able to be utilized such that the results are returned to the original representations. For particularized examples of the transformations of the data fields in accordance with the data field descriptors, reference should be made to the co-pending applications of Charles W. Bachman previously cited.

For ease of description, the above instruction has been shown to be utilized for a data base file and an application program. However, there are at least five applications which presently may utilize data field descriptors. The source field and/or destination field may be a data file from a communication facility, a data base of a user or the internal storage of the data processor itself. Thus, the five applications provided are (1) from a communication facility to the data processor, (2) from a data base to the data processor, (3) from the data processor to the data processor, (4) from the data processor to the communications facility and (5) from the data processor to the data base.

Although a data field descriptor finalizes the information that the instruction operates on, the actual instruction has great versatility. Thus, the instruction operates differently on the same data fields if another or different data field descriptor is accessed. In essence, this would be accomplished by changing the address syllable and offset in the instruction itself thus referencing a different data descriptor which through its displacement may reference the same or different data fields. In addition, the instruction can operate on different data fields with the same data field descriptors. This would be accomplished by other instructions (not shown) determining that more records are to be processed. The reference to the new operands would be made by changing the displacement in the base register. The next absolute address development would then provide the new data since it provides a different address. As a result, the same instruction would operate on a different data field. This flexibility of the instruction allowing different forms to be utilized is a primary feature in the invention since the same functionality provided by the instruction is able to be used with a great variety of differently encoded data fields thus obviating the limitations de-

scribed earlier. Moreover, by using data field descriptors not only may the same instruction be constantly used, but the data field descriptor may be changed to a currently more desirable form, thus enabling the evolution of the data base structure. This results since the data field descriptor which addresses and manipulates the data field is part of the control mechanism of the data processor and may be rewritten at any time to describe the new form of the data fields.

Moreover, the instruction can also be used in a hybrid situation wherein only one portion of the logical instruction uses the data field descriptor. As was explained earlier, the prior art in preparing the instruction required information concerning the attributes of the data fields. With this information, the instruction which could carry out the intended operation was then provided. Thus, in creating the instruction, the features of the data field were provided. A logical instruction utilizing a data field descriptor may be provided in combination with an instruction of this type. For example, the source data field and its attributes may be provided in the instruction whereas the destination field may be the logical portion of the instruction previously described and incorporating the data field descriptor. In similar manner, the source field may be the logical portion of the instruction and the destination field may have its features provided by the instruction. The third situation is the one described previously wherein both the source and destination fields are described by a logical instruction incorporating the data field descriptors. Thus, the limitation of preparing instructions based on the form of the data is effectively removed. Even if the form of the data field changes, the logical instruction and hence application program is not obsoleted since the instruction functions regardless of the data type or encoding given. This results since the data field descriptor is changed for the new data and the instruction does not operate on the data field descriptor until execution time. Thus, the nature of the data itself is, in every case, revealed only at execution time allowing the changes of the data fields through time to be made without having to reflect in the program the changes made in the data files. Economy is thus provided in not having to rewrite or retest the program. In the hybrid situation just described, the versatility is limited since the above changes would have to be made for the part of the instruction not dealing with the data field descriptor.

Moreover, the versatility of the instruction is enhanced by the ability to successively access data fields and perform its intended operation. For example, if it is required to move ten data fields, the same move instruction may be accessed for each field. Thus, by other means (not shown) the offset of the address syllable would be changed. However, the logical move instruction would account for the different encoding schemes at execution time via the data field descriptors accessed. Thus the ability to change the encoding schemes from execution to execution is another primary feature recognized by this invention. The well known loop used for moving data fields in the prior art would be applicable for moving successive fields but instead of requiring different move instructions based on each data field only one move instruction is necessary.

What is claimed is:

1. An apparatus for processing data fields having a plurality of different data structures, said apparatus comprising: