

DATA PROCESSING SYSTEM UTILIZING DATA FIELD DESCRIPTORS FOR PROCESSING DATA FILES

BACKGROUND OF THE INVENTION

Related Applications

1. "Address Development Technique Utilizing a Content Addressable Memory", invented by James L. Brown and Richard P. Wilder, Jr., filed on Aug. 24, 1972, having Ser. No. 283,617 and assigned to the same assignee as the instant invention.

2. "Segment Address Development", invented by Bienvenu and filed on May 16, 1974, having Ser. No. 470,496 and assigned to the same assignee as the instant invention.

3. "Data Processing System Incorporating a Logical Move Instruction", invented by Charles W. Bachman, filed on Dec. 13, 1973, having Ser. No. 424,381 and assigned to the same assignee as the instant invention.

4. "Data Processing System Incorporating a Logical Compare-Instruction", invented by Charles W. Bachman, filed on Dec. 13, 1973, having Ser. No. 424,406 and assigned to the same assignee as the instant invention.

5. "Data Processing System Utilizing a Hash Instruction for Record Identification", invented by Charles W. Bachman, filed on Dec. 13, 1973, having Ser. No. 424,391 and assigned to the same assignee as the instant invention.

Field of the Invention

This invention relates generally to data processing system and more particularly to an apparatus and utilizing a data field description.

Description of the Prior Art

Because of the advances in data base systems, it is possible for a broad class of programs to process files of the same data base. In general, these programs have been developed over a period of months and years and represent a large investment in both human and computer resources to bring them to a point of useful productivity.

Under conventional practice, these programs must recognize the exact representations of the data as stored in the data base file or must have the data reformatted. For the latter situation, before the program can operate on the data files, the data is reformatted to a compatible form each time the data is accessed from the data file, i.e., the data is transformed from the format as stored in the data file to the format which the program recognizes. This reformatting is accomplished by subroutines and hence is slow and expensive.

When the records and files used in the data base are defined, they are done so in a form which depends primarily upon the then existing need. Since the art of data handling is not static, additional needs or requirements surface with the problem that the data files may not be in a form which is particularly suited for the new purpose. For example, it is often highly desirable to add new fields to a record, or to change the size or recording mode of some fields. Since the format of the data determines the design of the subroutines, the more versatility desired in the handling of the different kinds of data, the more complicated the subroutine becomes. The main result of these developments is to effectively freeze the original format of the data base files and the

programs for extended periods of time. As a consequence, the evolution of data base structures that would normally occur is inhibited. This evolution is highly desirable to permit the data base files to better support the information system activities of the enterprise which they represent.

Historically, instructions have been defined for specific data types. By this is meant that under conventional practice, the attributes of the data field, i.e., the location of the data field, its length, encoding, etc. are determined and at compile time the description of the data is used to create the special instruction. In addition, it may be necessary during the execution of a program to select a combination of instructions for processing data whose characteristics cannot be given beforehand, but are given as a result of some preceding instruction. Thus, it is apparent that such instructions were created to include information based not only on specific data types but also for the data processor which would implement each instruction. If any of the attributes of the data field are required or desired to be changed, or if a different data processor is to be used, the instruction provided is rendered obsolete since the instruction is not capable of handling the new data. As a result, the binding of an instruction by a particular data type has also not allowed evolution of data base structures. What is desired is a system capable of incorporating developments without the attendant problems shown above.

OBJECTS OF THE INVENTION

It is a primary object of this invention to provide a data processing system which overcomes the above recited limitations.

It is a further object of this invention to provide an improved data processing system which is able to operate with a variety of data types from a plurality of sources.

It is another object of this invention to provide a data processing system which can quickly and efficiently utilize data fields in data base files.

It is yet a further object of this invention to provide data independence for data fields utilized in a data processing system by insuring independence of the program being used and independence of the data base file containing the record.

It is yet a further object of this invention to provide a data processing system which allows the separate changing of data files and/or programs with no modification of alteration of the other being required.

SUMMARY OF THE INVENTION

The foregoing objects are achieved according to one embodiment of the invention and according to one mode of operation thereof by providing in a data processing system an instruction format which incorporates data field descriptors describing the attributes and location of the data fields to be processed. In accordance with the instruction, the individual features of the data field descriptor are analyzed and compared by an arithmetic control unit in combination with a control store unit and a control interface adapter. Based on the determination made thereon, the arithmetic control unit carries out the operation specified by the instruction in accordance with the information provided by the data field descriptor. The same instruction may be used for a plurality of different data fields by incorporating different data field descriptors.