July 1972

B72-10350

NASA TECH BRIEF

Marshall Space Flight Center

Memory Reduction Through Higher Level Language Hardware

The problem:

The increasing demands on computer capabilities and the general trend in reduction of their size and manufacturing cost require a frequent re-evaluation of the basic logic/memory function assignments. The development of large-scale integration (LSI) produced marked reductions in hardware size and manufacturing cost and, hence, significant improvements in the logic function. However, only a fraction of this has been accomplished in the memory function.

The solution:

Memory savings can be realized with a token increase in logic hardware and by use of the higher level languages such as FORTRAN.

How it's done:

FORTRAN IV is used in this method as the higher level language, although other higher level languages may be used. In addition, compact coding is used within each specific instruction word. The allocation of functions to the preprocessor and the FORTRAN language processor (FLP), has been governed by the following considerations:

1. Maximum memory saving should be achieved with minimum additional logic hardware cost.
2. The execution speed should not be less than that obtained with a conventional compiler approach.
3. The speed-improvement possibilities uncovered should be implemented only with minor hardware penalties.

To eliminate a substantial increase in logic hardware, the preprocessor is assigned the functions of an assembler and a compiler, thus relieving the FLP from all functions not absolutely necessary for the primary goal of code compression. A simplified flow diagram of the FLP is shown in the figure. Instruction execution begins as soon as one memory word (containing two instruction words) has been obtained from memory and loaded in a 32-bit instruction register with a capacity of two memory words (four instruction words). A fetch overlap feature loads the next two instruction words into this register while the first two instruction words are decoded and executed. A memory address register specifies the next (double) instruction location in memory in the usual way, while a pointer locates the instruction to be executed within the instruction register. The memory address register and the pointer act together as an instruction location counter. The multiple instruction register allows multiple word instructions to be executed without further access to memory. For example, with the four instruction word register, DO loops of four instructions or less can be executed directly without requiring access to instruction memory until the loop is completed.

To illustrate the effectiveness of this method, memory requirements for a given mission were estimated to be 260,000 38-bit words. Sixty-thousand words were allocated for storing the executive program, data, and work area leaving 200,000 words available for instruction storage. The instruction storage compression ratio of 4:1 expected with the FLP approach indicates a memory savings of 150,000 words or 5.7 million bits.

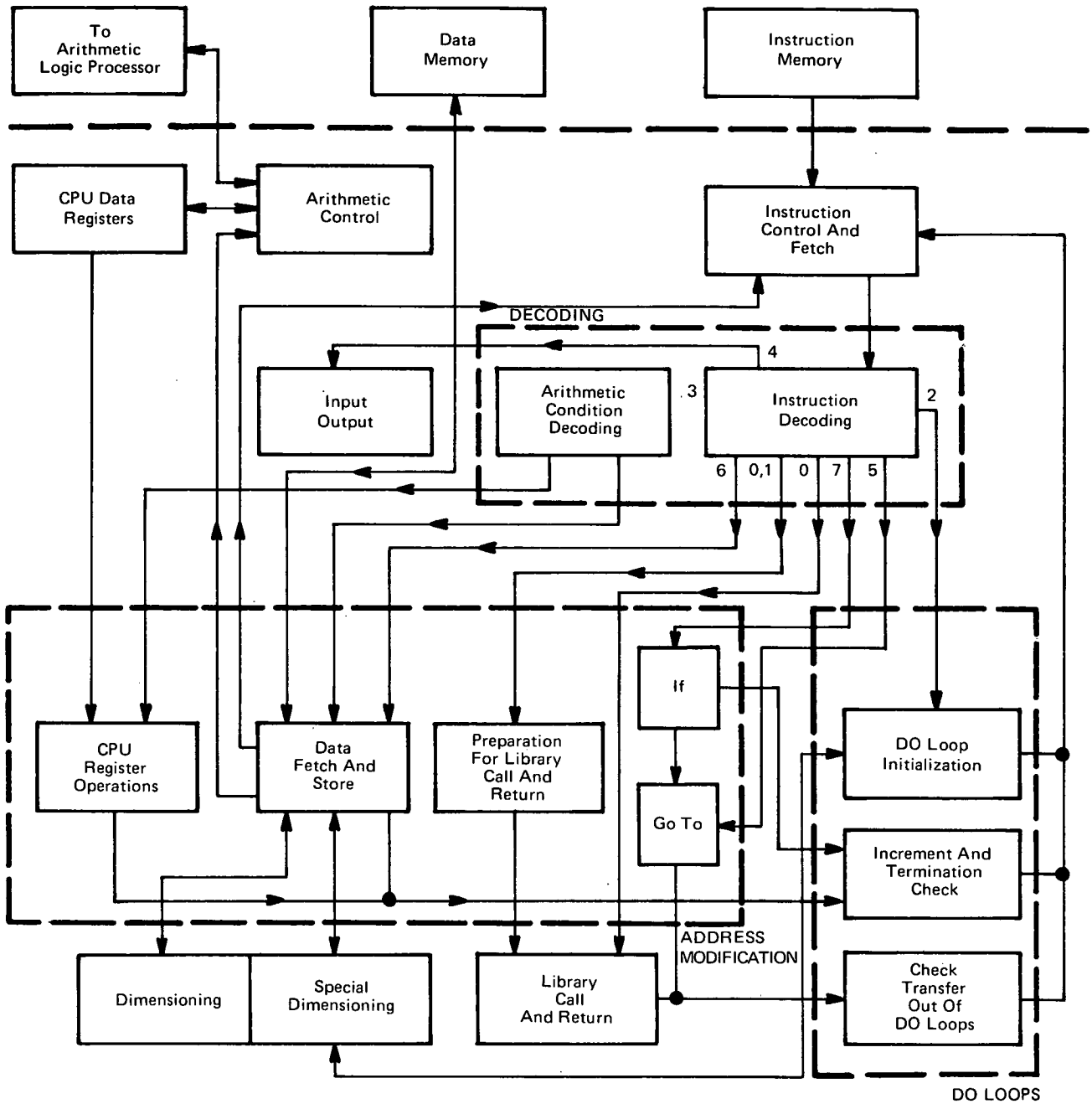
Note:

Requests for further information may be directed to:

Technology Utilization Officer
Marshall Space Flight Center
Code A&TS-TU
Huntsville, Alabama 35812
Reference: B72-10350

Source: H. Kerner and L. Gellman
Marshall Space Flight Center
(MFS-21128)

(continued overleaf)



FLP Flow Diagram