DATA DRIVEN ARCHITECTURE

W. Sippach

Columbia University Nevis Labs
Irvington, NY 10533

Abstract

The concept of "data driven architecture" is presented and compared with instruction driven systems. A processor for BNL E766 is briefly described. An argument is made for using data driven principals for the data acquisition and trigger systems at the SSC.

Introduction

Given the estimated event rates for the SSC of 100 MHz, and a detector with 10^6 signals, it is important to review the architectural principals of the data acquisition and trigger systems. What is needed is an architecture that has no scale limits, is free of data path conflicts, and provides a maximum number of data operations/sec/$.

An instruction driven multiprocessing system requires conversational arbitration of requests for bus access by the various memory and processing units. As the complexity of the system increases, so does the potential for conflicts, resulting in severe loss of bandwidth. A data driven architecture[1-4] is free of these problems and meets the requirements stated above.

Data Driven Scheme

In a data driven system, all the information for directing the data resides in the data itself. There is a set of specialized modules that perform operations on the data, and a network of interconnection cables that correspond to the data streams. A relation is defined as the set of operators (with their pre-loaded fixed data), and the set of interconnection cables. An operator can accept one or more input cables and can range from a primitive device that adds two numbers each clock cycle, to a complex device that is internally instruction driven, such as a microprocessor.

Interconnections

The interconnecting cables define data streams, and are standardized so that the modules are generally interconnectable. Two fields are required: one for data, and one for control. The control field contains two kinds of information: first, a name or label that specifies membership of a word to a particular data set; second, special bits that bracket the data stream into words and blocks. An additional special bit called "hold" is asserted by "destination modules" that are unable to accept a cable transmission, and by "source modules" that have no data to send. "Holds" propagate against the data stream, but are annihilated whenever they intercept empty data elements. Together, these special bits align the data in the streams entering and leaving the modules.

A Track Finder As An Example

Figure 1 shows a relation for finding curved tracks in a MWPC spectrometer. Seven types of pipelined modules are required: the Index Generator (I) counts data elements on its inputs, and generates all possible index pairs on its output; the List (L) stores the data arriving at its write port, and retrieves data by index at the read port; the Normalizer (N) performs the operation aX+b; the Adder (+) aligns and adds words from its two input ports; the Map (M) provides a memory cell for each possible wire number at its write port, and allows retrieval of 32 neighbor cells from data at its read port; the Mask (MSK) compares data from two Maps according to a pre-loaded curvature relation; the Table (T) transforms the Mask output into a curvature parameter, and "names" the result of the test. The entire relation is performed on the four input streams each clock cycle (typically 25 ns). The execution time remains 1 clock cycle no matter how many other relations are added to the system, making the time of the computation independent of size. The system is conflict free, since each operator is only concerned about data on its input-output cables. The operators can be specialized to achieve high computation efficiency and to minimize the number of devices needed.

A Special Maintenance Bus

An additional bus is required for maintenance. This bus connects to all modules and allows direct addressing of

any module, and of any register or memory word inside a module. The bus is quite simple and typically supports bit serial data transfers, to minimize hardware.

Subroutines

Processing subroutines can be separated out by introducing control memory. The basic structure is shown in Fig. 2. A control stream is introduced that travels alongside the data stream(s). The control allows regulation and alignment of blocks in the system. The blocks are delivered to the subroutine structure by a set of buffers and received by a second set of buffers whose outputs connect to other subroutines. The buffers perform block operations such as dropping data, replicating data, renaming data, etc. The memory is constructed of first in-first out (FIFO) buffers, with control connections for controlling the input and output ports.

Figure 3 indicates how multiple data cables can be collected to represent a wide data stream, and how streams can be merged and branched. This architecture allows a large problem to be broken up into small subroutines imbedded in control memory. A large system of this type may have loops, and can contain many events (at any moment in time), distributed in both the subroutines and memory. The average processing rate will be limited by the slowest subroutine, so that the subroutines should be optimized. The buffers soak up short term fluctuations and regulate the system.

The BNL E766 Processor

This processor[5] performs complete track reconstruction, online, for a 12,000 wire multiparticle spectrometer. The details are described in the literature, and only the general properties are mentioned here.

The system uses about 400 modules (9 in. x 9 in. boards), with about 35 types of modules. The system is synchronous with a 40 MHz clock. A partial list of the hardware subroutine types is:

1. Multiplicity Logic Triggers,
2. Line Finders,
3. Line Matcher for 4 views,
4. Track Parametizer,
5. Least Square Fitter,
6. Track Cleanup,
7. Physics Triggers.

A limited part of the processor was used during the 1985 data run, and the entire system is expected to be operational in early 1986. The processing power of this system is estimated to be equivalent to 10^5 VAX's, based on knowledge of off-line tape processing on a VAX 11/780.

Applying These Principals to an SSC Detector

The data driven architecture is most obviously applicable to the front end data acquisition and trigger hardware, where a high degree of specialization is necessary to achieve adequate bandwidth. For the bottom, there seems

to be reasonable acceptance of computer farms[6] as a final trigger and data compression system. Careful examination will show, however, that these one-event one-processor farms are also data driven. For example, no conversational polling is required to direct incoming events to idle processors. It is only necessary for processors to send their names to a list of available processors when they finish their task, and for a name from this list to be attached to an incoming event to direct the data to an idle unit. The processing power is linear in the number of processors, and there are no conflicts.

Between the top and bottom of the system is a requirement to specially process the various detector components, at high speed, to reduce the work load of the farm (which is Fortran programmed). Rather than breaking the natural data flow by standardizing the data onto instruction driven buses (Fastbus, Camac, etc.), it seems reasonable to preserve the data driven architecture that already exists at the top and bottom of the structure. This choice removes any scale limitations, and provides a means of optimizing the different parts of the system.

Summary

The basic principals and properties of data driven architecture have been presented. We believe they are sufficiently interesting to serve at least as a guideline for designing a very high bandwidth data acquisition and trigger system, as required at SSC.

Bruce C. Knapp, who is building the E766 Processor, should be given credit for much of the concept.

References

1) "A Hardwire Architecture for Processing Detector Data in Real Time", B. Knapp, W. Sippach, Proceedings of 1978 Summer Workshop, Brookhaven National Laboratory.

2) "Data Driven Processing", B. Knapp, W. Sippach, IEEE Trans. Nucl. Sci. NS-27, 578 (1980).

3) "Study of Hadronic Production and Spectroscopy of Strange, Charm and Bottom Particles at the Tevatron", FERMILAB Proposal for Experiment E690. Columbia University/Fermilab/University of Massachusetts/ University of Mexico, Spokesman: B.C. Knapp.

4) "Use of a Parallel Pipelined Event Processor in a Massive Dimuon Experiment", Florida State University, FSU-HEP-851001, Y.B. Hsiung, J.A. Crittenden, H. Cunitz, W. Sippach, Columbia/D.M. Kaplan, FSU/K.B. Luk, Univ. of Washington (1985).

5) Status Report for E766, October (1985). Columbia Univ./ Massachusetts/Fermilab/Mexico. Spokesmen: B.C.Knapp & M.N. Kreisler.

6) Paul Kunz et al, IEEE Trans. Nucl. Sci. NS 27, 582 (1980).

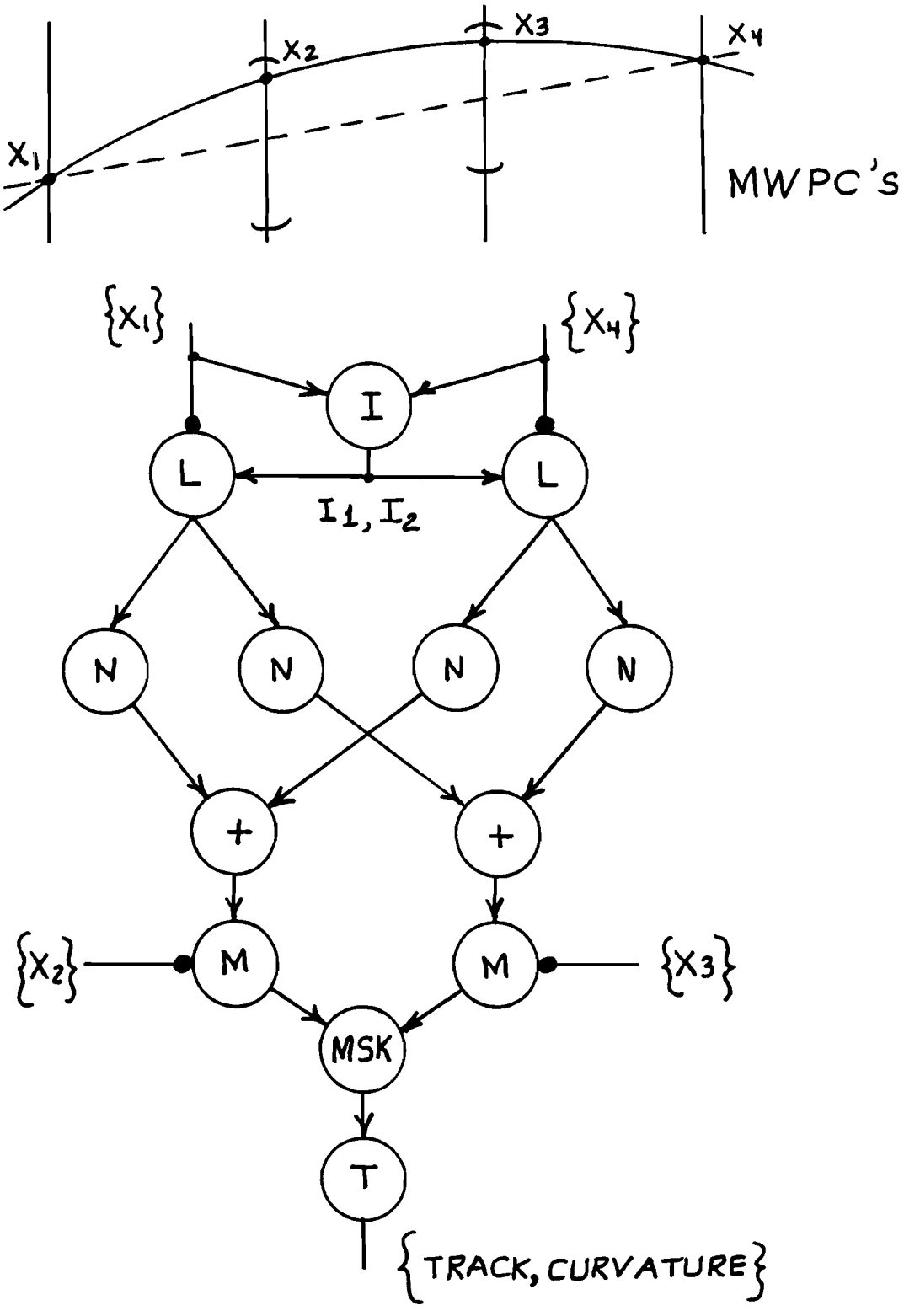


FIG. 1 TRACK FINDER

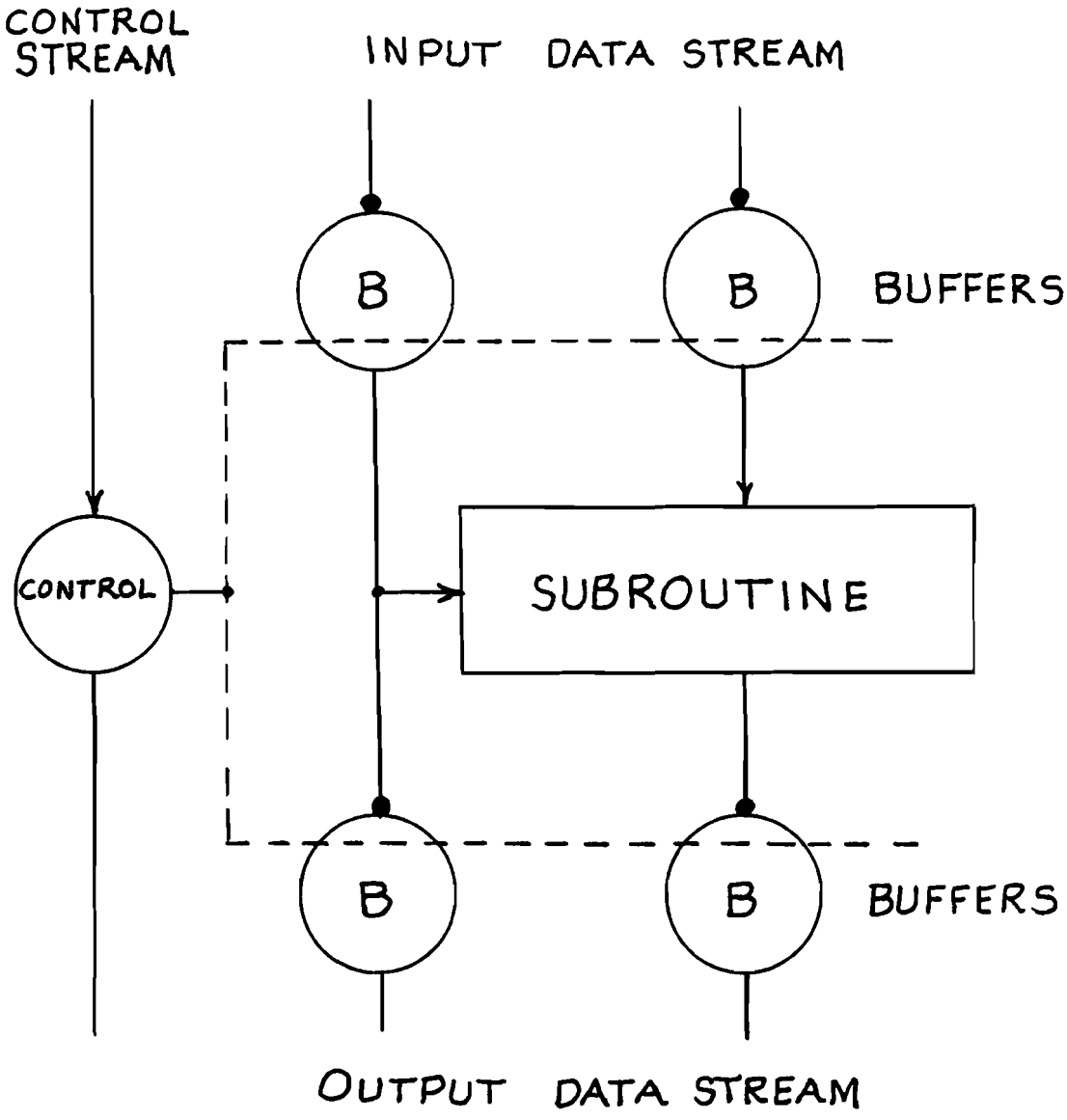
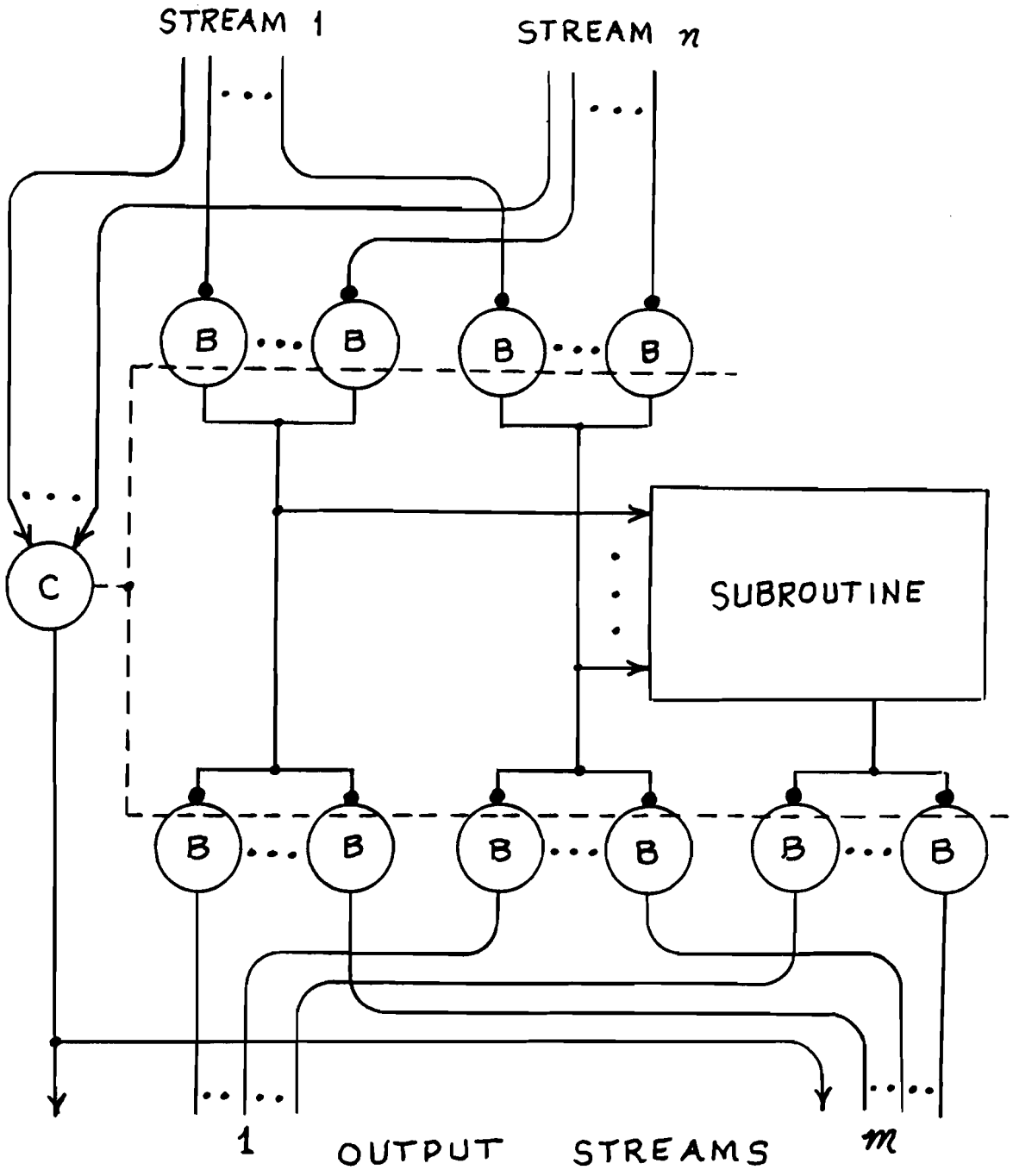


FIG. 2 BASIC MEMORY SUBROUTINE STRUCTURE



MERGING AND BRANCHING MULTIPLE DATA STREAMS

FIG. 3