FR8002095

Wire chamber conference.
Vienna, Austria, february 27 - 29, 1980.
CEA - CONF 5226

MORPION : A FAST HARDWARE PROCESSOR FOR STRAIGHT LINE FINDING IN MWPC

M. Mur

Département de Physique des Particules Elémentaires,

CEN Saclay, France.

ABSTRACT

A fast hardware processor for straight line finding in MWPC has been built in Saclay and successfully operated in the NA3 experiment at CERN. We give the motivations to build this processor, and describe the hardware implementation of the line finding algorithm. Finally its use and performance in NA3 are described.

1. MOTIVATION

Achieving on line the pattern recognition process, or a part of it, is interesting in several ways :

- on line filtering : rejection of bad events, flag of good events, etc.
- time saving on off line computers for the kept events
- fast monitoring of the experiment, by immediately analysing off line a sample of everts selected on line according to physics criteria.

The NA3[1] experimental set-up (fig.1) allows the study of various physics fields. The pattern recognition process depends on the physics studied and on the trigger used, but always has to find straight lines in the chambers at the back of the magnet. With a dimuon trigger, one is able to recognize the muons among these back lines, and approximately calculate the vertex and the dimuon mass only from the back straight lines. At this point, events can be rejected if the muon reconstruction is not satisfying. The ratio of rejection is higher (90%) in the case of an open spectrometer than in a beam dump experiment (40%) where events are cleaner.

Taking benefit of the experience of A. Fucci (DD, CERN) in the field of on-line processors[2], we decided to build a hardware processor able to find straight lines in the MWPC. This task is time consuming in an off-line program , especially when the multiplicity is high in the chambers, since it involves many repeated loops on the same data. Doing it in a dedicated processor with a matched architecture is the fastest way, and is possible since a 16 bits integer data format can generally handle both the MWPC precision and size. It allows some more time to be spent in the host computer (or in a programmable processor) to evaluate the event, and to go further into the pattern recognition process for the good events in another processor, perhaps slower, but programmable and capable of more precise calculations.

2. ALGORITHM

We decided to implement the line finding in the MORPION using the same algorithm as in the off-line program[3] for testing and compatibility purposes. The basic features of this algorithm are the following :

Assume n planes of the same wire direction.

- choose two pivot planes F(first) and L(last) starting from the most external of the group.
- try all possible combinations (fl) of one coordinate (f) in F plane and one coordinate (l) in L plane, to initialize a line for each (fl) :
  . calculate for each inner plane the intersections of this line with the inner planes.
  . scan the inner planes to check their content against the predicted value with an arithmetic window ε.
  . validate the track if there are enough real points on the line
  . flag the points for further use if the track is valid.

Notes :

a) the value p of the intersection of the line (fl) with inner plane P can be expressed as

$$p = f + C_{flp} * (1-f)$$

where :

$$C_{flp} = \frac{Zp - Zf}{Zl - Zf}$$

depends on the Z position of corresponding planes and can be stored in advance in a look up table.

b) the inner loops are achieved in serie in the off-line program but are done in parallel in the processor

c) inefficiencies of chambers must be taken into account. If any of the first planes F or L has been inefficient, one must try a new FL combination with more internal planes. If 1 missing point per track is allowed, 3 combinations of pivot planes have to be done successively. If 2 missing points are allowed per track, 6 pivot planes combinations have to be done.

d) tagging : points are flagged, only when the track has not the minimum efficiency required, in order not to find a smaller segment of it later. Flags written in a combination of FL planes, are only considered in later FL combination, not to introduce geometrical bias in the rejection of points.

3. IMPLEMENTATION

3.1 General Organisation (fig.2)

The hardware architecture is matched to the algorithm. The central unit can search lines in up to 8 planes at a time. Seperate buffers for input (chamber data) and output (associated tracks) allow concurrent input and process operations. At input time, planes are stored in the input buffer in the order that they will be processed; wire numbers are changed in flight into space coordinates.

3.2 Central Unit (fig.3)

As soon as a plane to be used is completely input, it is loaded into a memory cell in the cental unit. These memory cells[4] have special facilities to control loops on their data : the top of the loop is indicated and can automatically reset the adress to the lst word of the plane. A counter loaded with the number of coordinates can be decremented each time a point is used in a track, and issues a useful "zerospark" condition when there are no more unused words. When all planes of a group are loaded into their corresponding working RAMS (MØ → M7) the line search process starts.

The control section choose F and L planes, points the address of their memory onto the correct f and 1, and calculates sequentially the intersections of the line (fl) with the inner planes, in a pipe line mode. When all intersections have been calculated, the loops over inner planes are started in parallel. If a correct straight line is found (enough HITS in inner planes), values of the coordinates (real or predicted) are routed to the output buffer through the three state bus.

3.3 Logic Control

In the central unit, the hardware is the same for the 8 possible planes (except the most external, that can never be inner, and don't need comparators). Each of these eight planes can be -First, -Last, -Inner, - External or unused. For each of these 4 cases a 2 bits status is attached to the plane.

The central loops control is done in a array of PROMS. Each plane has an idividual PROM; two address bits of this PROM are the plane status, another address bit is time dependent and indicates if we are in the parallel inner loops sequence or in the (fl) line initialisation sequence. The other

address bits of the PROMS are the address control, zerosparks, arithmetic hit and flag, individual for each plane. The PROMS outputs are combinated to generate the control signals for the loops,

SKIP fl (for l has a flag)

SKIPFL1 (zerosparks in either F or L)

- END of inner loops

Another PROM indicates

VALID TRACK (enough hits in inner planes)

SKIPFL2 (enough ZEROSPARKS in inner planes).

The inner loop control is synchronous. A main clock is sent to increment all inner address counters. New data are compared between each clock pulse, changing PROMS addresses. At the next clock pulse, all necessary control signals from PROMS are stable and are strobed by the clock to decide what to do next. Fig.4 shows how the control signals described control the flow of the process sequence.

4. FLEXIBILITY

Various degrees of flexibility dealing with detectors and algorithm options have been implemented.

Algorithm options

On each event one can do up to 4 times the basic straight line algorithm on different groups of up to 8 planes.

- for each track, in case one deals with less than 8 planes the extrapolation of the track in virtual planes, of which one can program the Z position, can be output. This is especially useful for saving time on the computer (or processor) that uses output track to go further in the on-line program.

- for each pass, one can allow one or 2 missing points per track.

- a pass can use a subset of the planes used in the previous pass, with different options, but taking into account the flags written in the previous pass (DUAL option).

- the arithmetic comparison window can be programmed for each plane from 1 to 255 mm.

Detector options

- all plane positions in X, Y, Z are programmable. Wire space can be 1, 2 or 3 mm.

Input options

Any order of planes at input is allowed, although the input of planes in the order that they will be processed (ex. X planes then Y planes) is more efficient, since there can be a time overlap between input (of Y planes) and process (of X planes).

All these options can be changed at any time in a Fortran program run on the host computer. This program builds a formatted block of control words that is then transfered to the processor through a Camac control channel.

5. TIMING

5.1 Estimation

One can roughly estimate an upper limit of the processing time for a typical case.

Assume a pass with 8 planes, 6 points required on each line.

$t_{max} = 6 \quad N^2 \quad (6 \; tc + 2tc + N \; tc)$

- 6 FL combinations
- N^2: nested loops
- $6\,tc$: F, L
- $2tc$: pipeline offset
- $N\,tc$: serial pipeline calculation of 6 inner values
- parallel inner loops

where N = number of clusters per plane
tc = main clock period (240 ns)

In this case we gain a factor 6 on the coefficient of the highest power of N by doing the inner loops in parallel.
for N = 4, $t_{max} \simeq 300$ µs

5.2 Measurement

The total processing time has been measured on true events (lines in 8 X planes and then in 6 Y planes).

For a clean event with one track t = 150 μs.
For a very big event with high multiplicity resulting in 6 X lines and 6 Y lines : t = 5 ms
The processing time is about 10 times faster than the same routine on a 7600 computer.

6. USE IN NA3

6.1 The system has been used in NA3 since Nov.79. It introduces no dead-time, since all I/O operations to and from the MORPION are performed between bursts (event data are stored in buffer memories during the bursts). 3 passes are programmed :

a) straight lines in 8 planes at the back
b) dual pass for X straight lines in CP3-4 only
c) straight lines in 6 Y planes at the back

The tracks are then used to calculate the dimuon mass on line in the PDP 11-45 host computer :

a) find which lines are μ lines using trigger "damier" chambers[5] for X-Y association
b) find Z vertex from Y lines
c) from Z vertex calculate momenta of the μ's (P_1 and P_2)
d) calculate the dimuon mass, using
$M^2 = 2P_1P_2(1-\cos\theta)$, θ = angle between the two tracks.

The on-line mass distribution (fig.5) and vertex position are plotted online.

We sample events of masses higher than 3.8 GeV/c^2 on a seperate tape. A mass spectrum of these sampled events (dotted line) compared with the total mass spectrum (continued line) is shown in fig.6, for about 50 000 events. Only 4% of events are kept on the tape of selected events, without loosing high mass events, while having a 97% rejection ratio at the mass of the ψ. This filter allows to speed up the off line analysis of high mass Drell-Yan events and allows a fast monitoring of the experiment.

REFERENCES

(1) J. Badier et al., A large acceptance spectrometer to study high-mass muon pairs, Submitted to Nucl. Inst. and Methods, March 1980.

(2) A. Fucci et al., Design of hardware processors for use in high energy physics, Proc. of 1974 IEE Conf. on Computer Systems and Technology, London. Oct. 1974 pp 42-48.

(3) M. Hansroul, C. Verkerk and P. Zanella; Hardware processors for pattern recognition tasks in wire chamber data; Int. Conf.Instrumentation for high energy physics, Frascati, May 1973, pp 497-499.

(4) A. Fucci et al., Memory Module 4271C; CERN DD Internal Report.

(5) J. Boucrot et al., A trigger system using cathode read-out chambers and a coincidence matrix, paper to be published in this volume. Presented at the Wire Chamber Conf., Vienna 1980.

FIGURE CAPTIONS

Fig. 1 NA3 experimental set-up.

Fig. 2 General organisation of the processor.

Fig. 3 Central unit.

Fig. 4 Process flow diagram (1 pass).

Fig. 5 On-line dimuon mass distribution.

Fig. 6 Off-line dimuon mass distribution.

CERN NA 3 SPECTROMETER

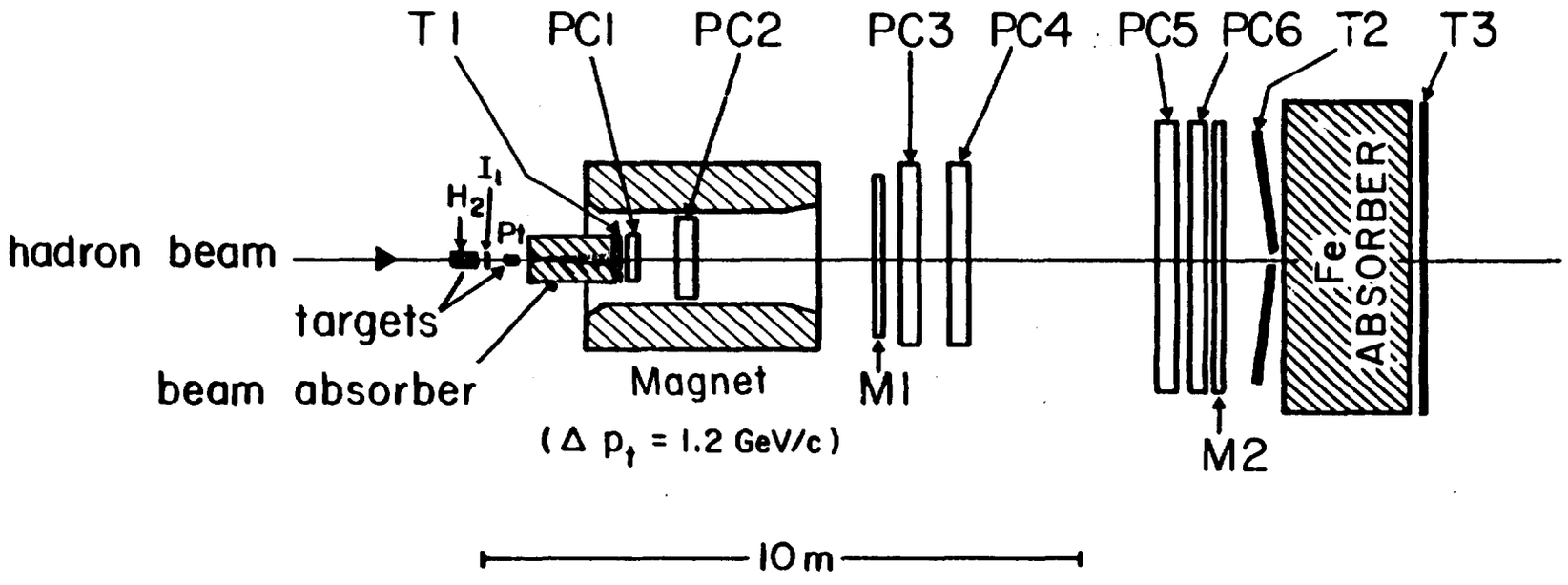

FIG. 1

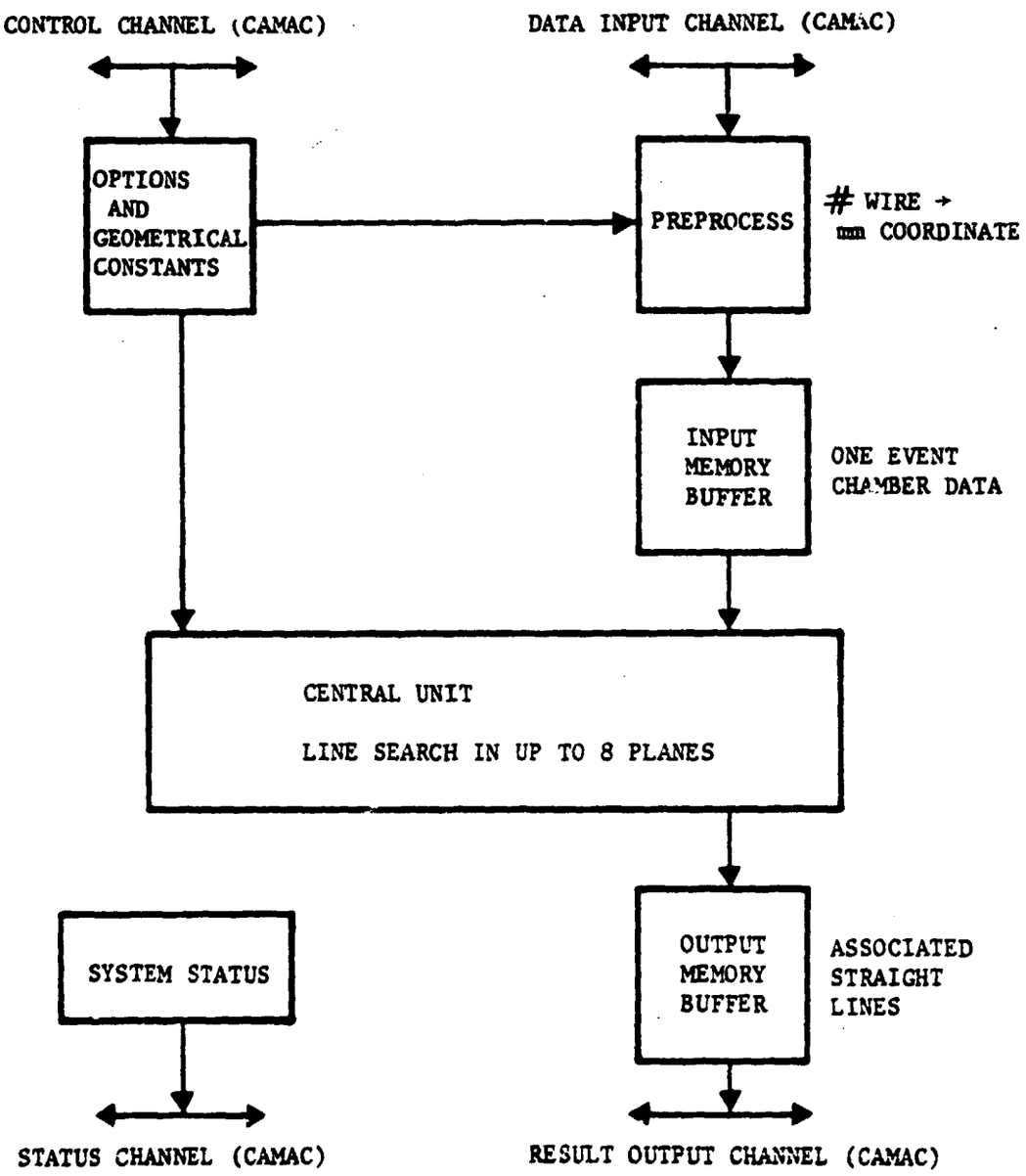
CONTROL CHANNEL (CAMAC)
DATA INPUT CHANNEL (CAMAC)
OPTIONS AND GEOMETRICAL CONSTANTS
PREPROCESS
WIRE → mm COORDINATE
INPUT MEMORY BUFFER
ONE EVENT CHAMBER DATA
CENTRAL UNIT
LINE SEARCH IN UP TO 8 PLANES
SYSTEM STATUS
OUTPUT MEMORY BUFFER
ASSOCIATED STRAIGHT LINES
STATUS CHANNEL (CAMAC)
RESULT OUTPUT CHANNEL (CAMAC)

FIG. 2

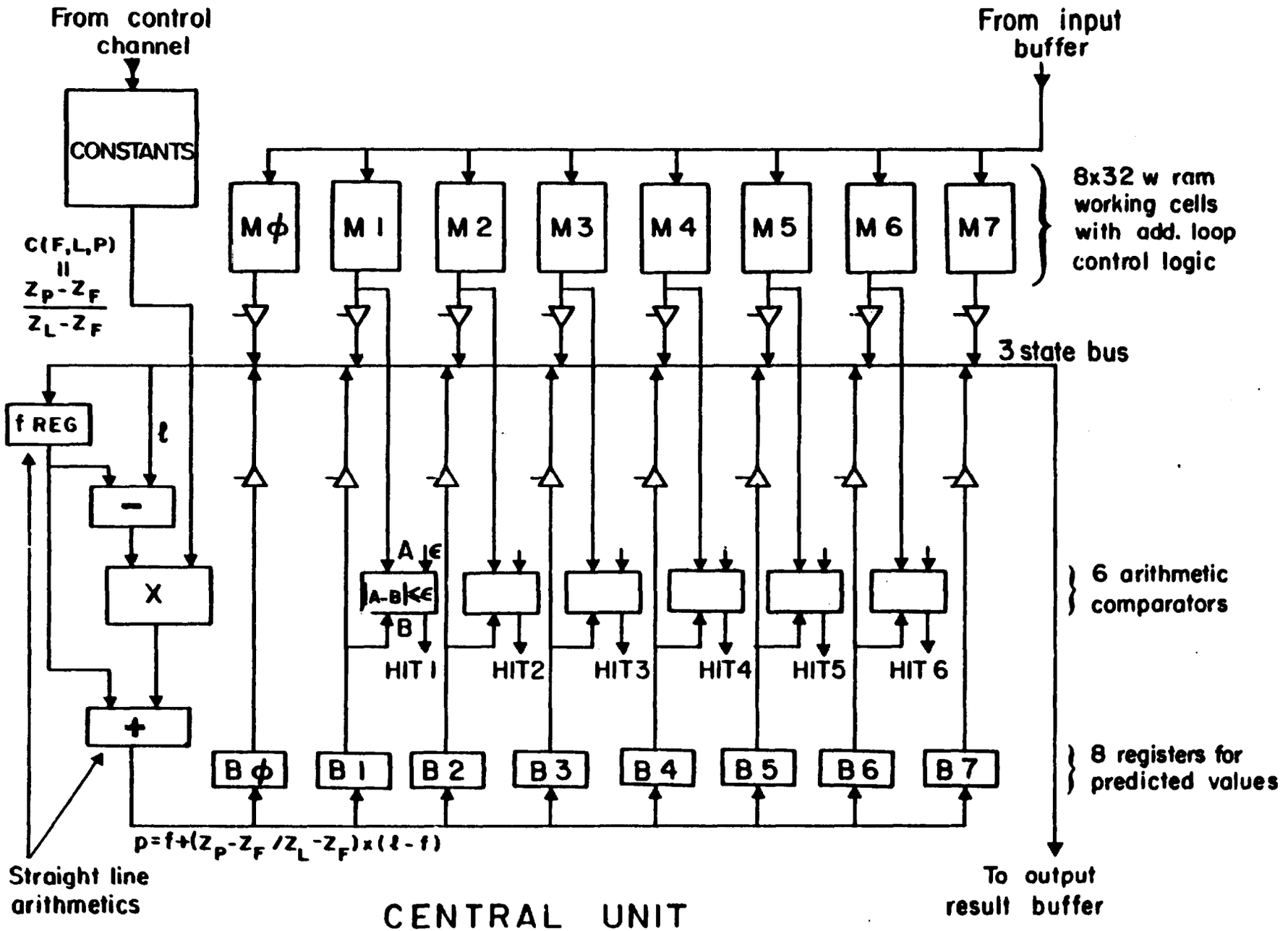
From control channel
From input buffer
CONSTANTS
C(F,L,P) = (Zp-ZF)/(ZL-ZF)
Mφ
M1
M2
M3
M4
M5
M6
M7
8x32 w ram working cells with add. loop control logic
3 state bus
f REG
ℓ
−
X
+
A
ε
|A-B|<ε
B
HIT 1
HIT2
HIT3
HIT4
HIT5
HIT 6
6 arithmetic comparators
Bφ
B1
B2
B3
B4
B5
B6
B7
8 registers for predicted values
p=f+(Zp-ZF/ZL-ZF)x(ℓ-f)
Straight line arithmetics
To output result buffer
CENTRAL UNIT

FIG. 3

PROCESS FLOW CHART (1 PASS)

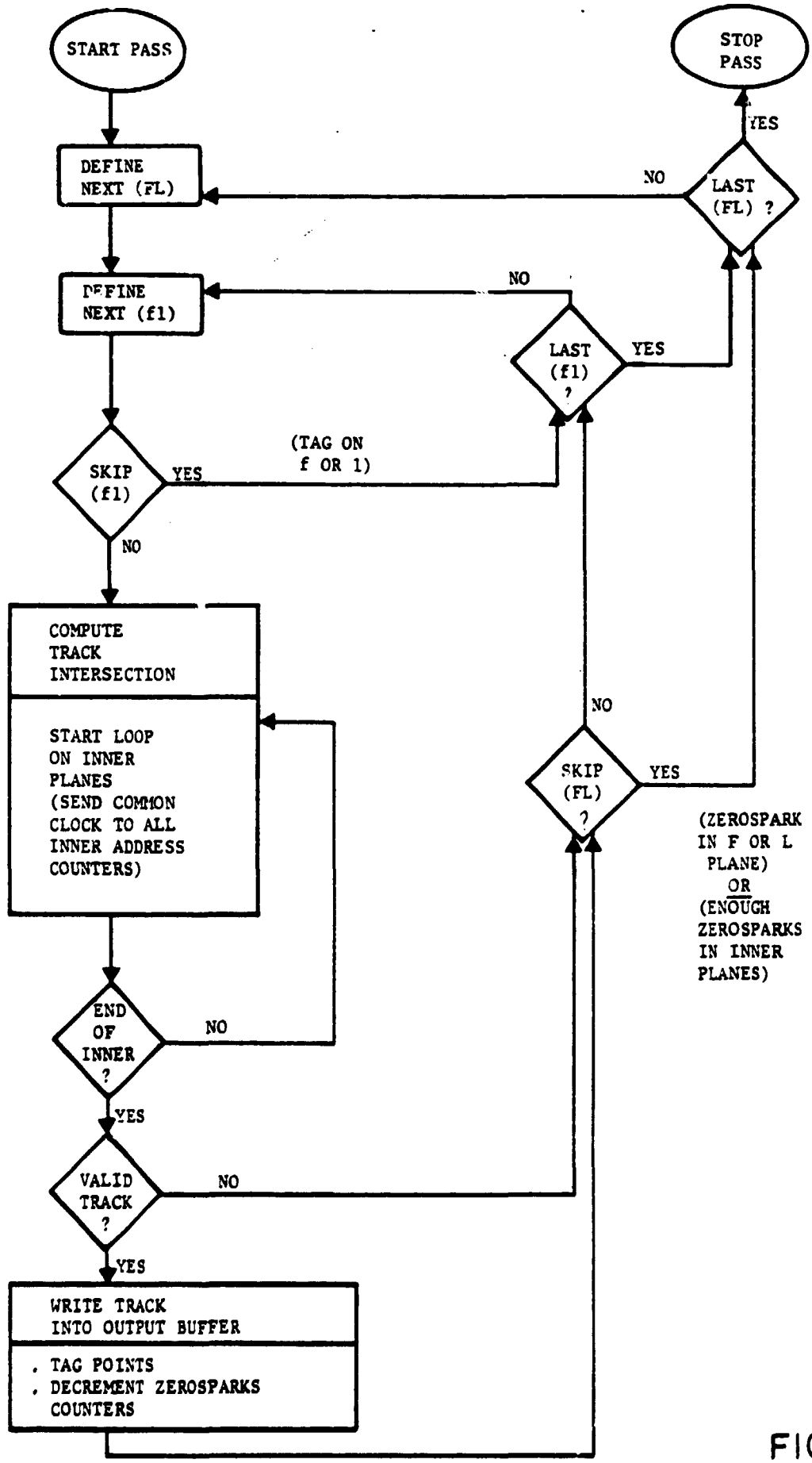

FIG. 4

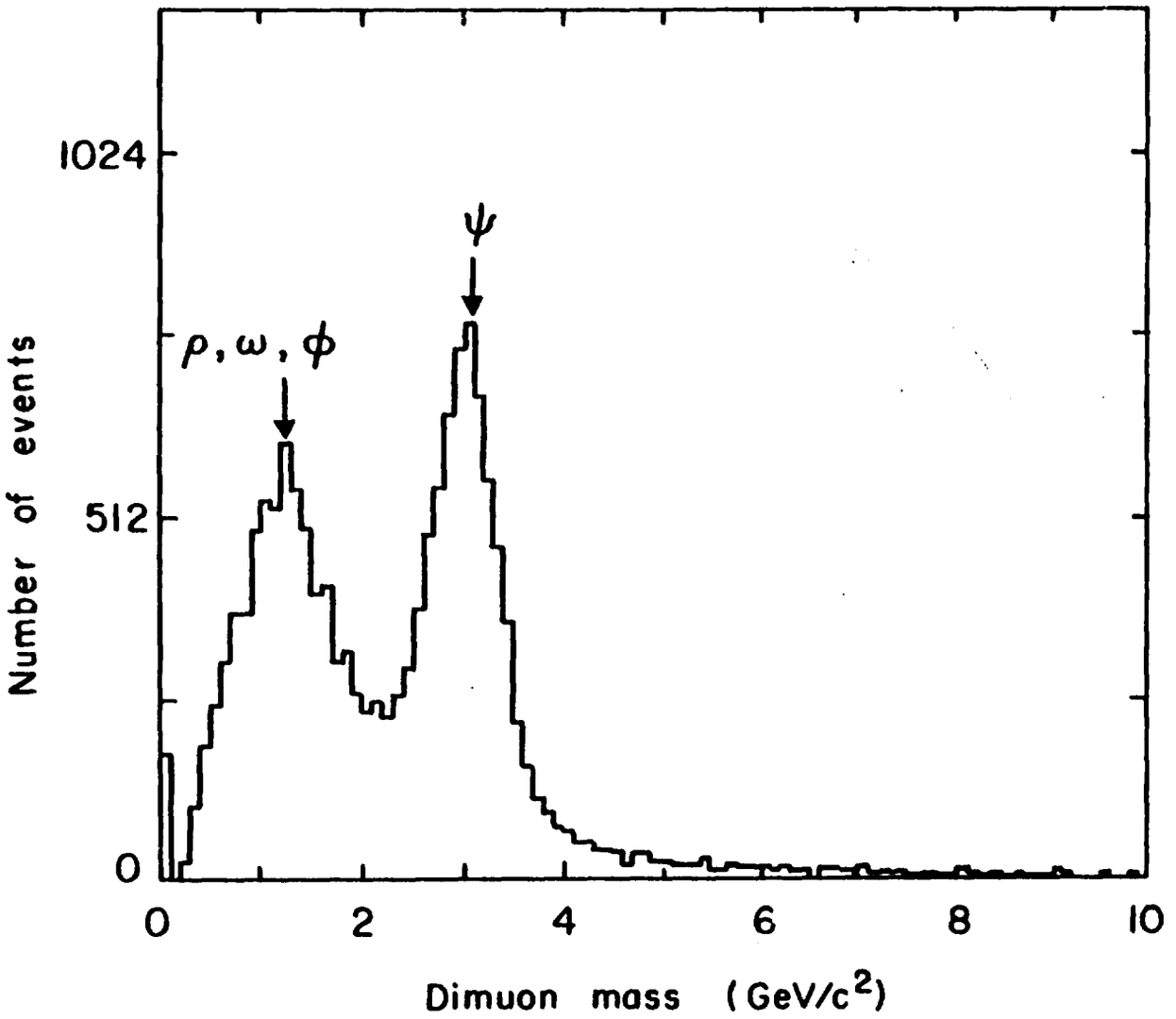

FIG. 5

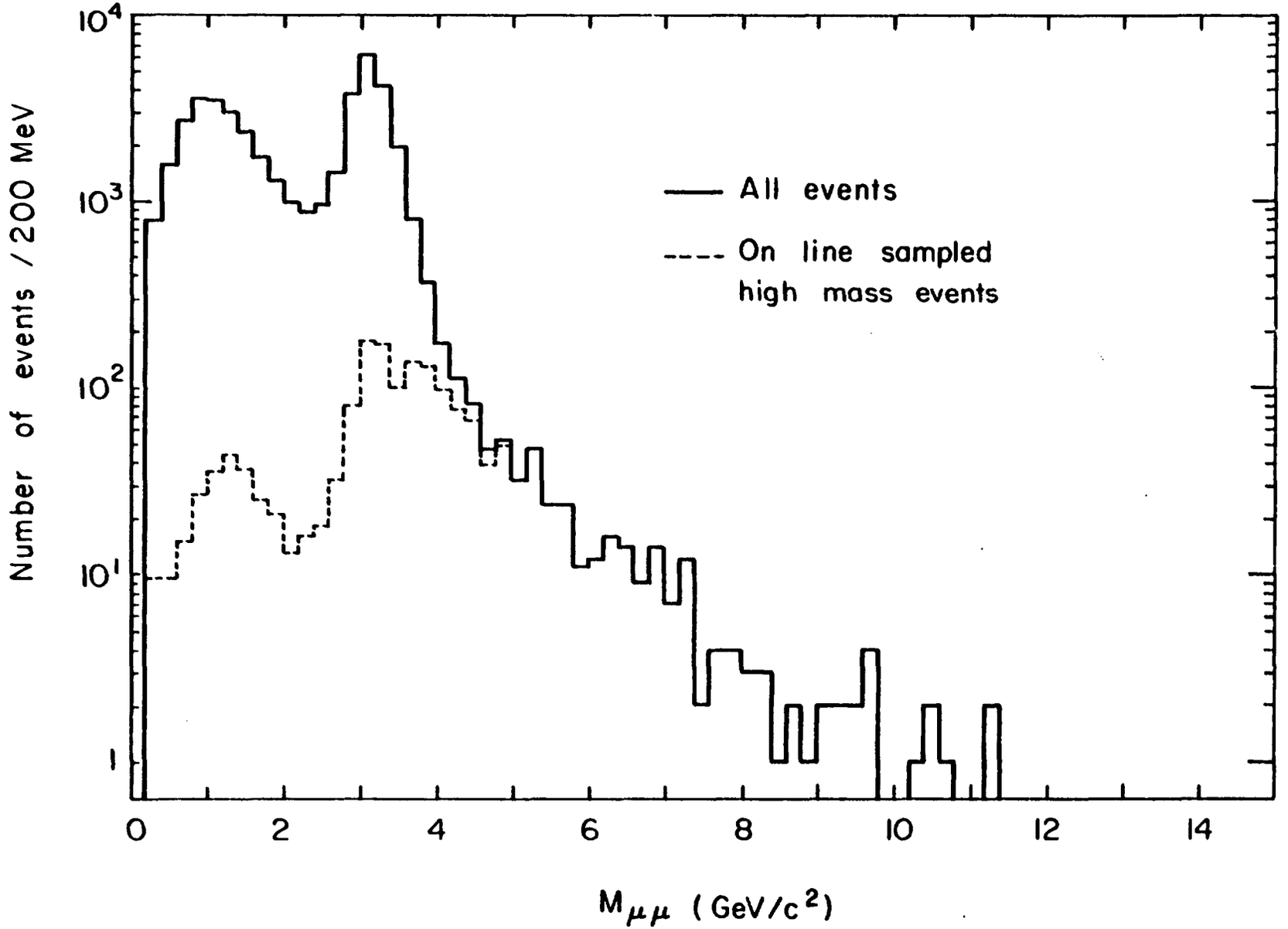
Number of events / 200 MeV
10^4
10^3
10^2
10^1
1
0
2
4
6
8
10
12
14
$M_{\mu\mu}$ (GeV/c^2)
All events
On line sampled high mass events

FIG. 6