31-March-82

J. Paradiso / ETH

A PROPOSED DESIGN FOR A FLASH DIGITIZATION SYSTEM TO BE USED WITH 3-WIRE TEC CONFIGURATIONS

a) Fast (100 Mhz) acquisition circuitry

The enclosed figure shows the digitizer structure. Only the component which handles the acquisition process is sketched; details of readout logic are left to imagination, although general guidlines and suggestions are given later. The shaped anode signal is buffered by a single-ended emitter follower before being input to the flash ADC chip (Siemens SDA5010 or similar here). It is assumed that all needed voltage gain is derived from the shaping amplifier. The zener diode at the input protects the ADC from analog transients greater than the allowed maximum (≈2.5 V). An analogous circuit buffers the shaped difference (ΔPW) signal, except here we have a bipolar signal, thus must use a double-ended class AB amplifier, and two zeners for transient protection. The ADC reference voltages are buffered by emitter followers; for the anode, the positive reference is fixed and the negative reference is variable (the dual diodes prevent the maximum of ≈3 volts from being exceeded). The ΔPW positive reference is set at the input of an emitter follower; this is inverted via a unity-gain operational amplifier and fed to the negative reference input, thus the "range-adjust" potentiometer controls both the positive and negative sensitivity, which are symmetric about ground.

Both internal and external clocking capabilities are desirous; by employing RC feedback around a compensated 100114 line receiver a very stable gated oscillator can be fabricated as the "internal" clock, which begins oscillating as soon as a "start" signal is received (ie. the gate ("busy") goes high). The external clock can be synchronized to the start signal via a flip-flop so that clocking pulses are output beginning with the first external clock edge following receipt of a start signal; in this case, the timing error will be within 1 clocking period (hence 10 nsec at 100 Mhz). There is essentially no such timing error with the internal "gated" clock, but the external clock bears the advantage that all ADC's being driven are at constant phase difference, and can be compared directly in an analysis. Both options can be supported with few additional components, and the Int./Ext. clock option can be switch selectable. Upon reaching the terminal count, the clock must be gated off; this is done by a "stop" signal (ie. gate ("busy") is brought low) derived from the address counter. A choice of several maximum sample counts can be made available by switch-selecting counter bits for this stop signal. This feature will prove useful, since different wire configurations subtend different maximum drift path lengths (depending upon thier position in the wedge-like chamber cells), thus maximum drift times will correspondingly differ.

The state of the acquisition logic is defined by two RS latches. The "Busy" latch is set upon the reciept of a start pulse, and is re-set when the terminal count is achieved, thus it defines a "digitizing" state. This output gates the clocking logic and is made available to CAMAC circuitry for readout-ready polling via F8. It also drives the "write-enable" of the memory arrays, thus clocking in data only during acquisition. The "Data-take" latch is set after a "start" is received and is re-set when the module is cleared or CAMAC readout is finished, thus it defines the state when the module is either acquiring or posesses valid data. To prevent additional triggering during readout, the "start" input is gated by this latch; no "starts" are accepted while it is set. The coincidence Data-Take·Busy defines a "data-present" state when acquisition is finished and un-read data still remains in memory.

This signal can be used to stop the memory readout, thus readout software need not be programmed with the terminal-count value.

The current address into memory is defined by the "address counter" (possibly an array of 10016's). It is advanced by the gated clock during acquisition and cleared at the end of the acquisition cycle by a monostable triggered when the Busy latch output falls. The address counter can be both advanced and preset by the readout logic.

The digital outputs of the flash encoder chips are first buffered (line recs., etc.) before being fed to the inputs of 6-bit ECL memory arrays (the maximum drift time expected in this chamber suggests a depth of $\geq$ 1K words). Fast 1K memories (such as the Hitachi HM2112-1) can be used, slower memories (address access < 20 nsecs.) can be double-buffered such as in the scheme developed at SIN around the Fairchild 10415A.

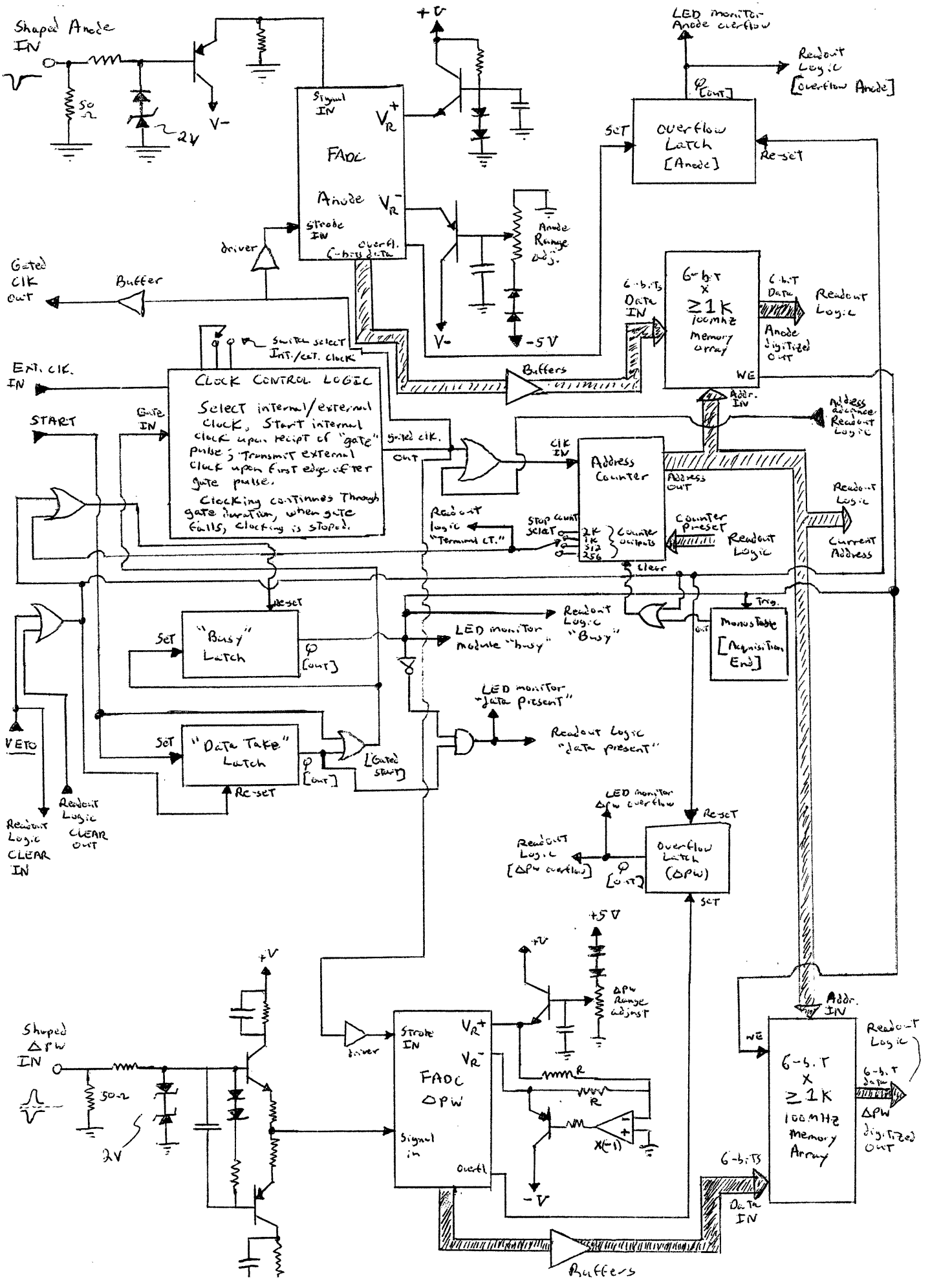
The overflow bit output from the flash ADC's sets fast RS latches. If an overflow occurs during an acquisition cycle, the appropriate latch is set and is not re-set until CAMAC readout is completed, or the module is cleared. The latch outputs are available to CAMAC for overflow flagging in the termination word (see ahead), and power LED's which give an obvious indication of overdriving and are invaluable as a rapid saturation diagnostic.

All latches and counters are re-set and the module is returned to a quiescent state when a pulse is applied at the "VETO" input, or a CAMAC clear is received. The "VETO" is also transmitted to the readout logic to halt any progressing read sequence.

The analog and ADC reference circuitry is simplified and contoured for the standard anode (unipolar) and ΔPW (bipolar) signals. In charge division applications, one unipolar ($V_{close}+V_{far}$) and one bipolar ($V_{close}-V_{far}$) signal are also recorded, thus the same circuit can be employed.

Hardwired analog division might be possible by driving the reference chain of the flash ADC by the divisor signal (Ref. Hallgren and Verweij, CERN EP/79-133), however this method entails identical shaping of both signals, and is quite inflexable and sensitive to offset and gain variations. The flexability offered by software division may be manditory to properly normalize the signals.

Another option valuable for test purposes is a storage scope simulation, where the memory may be cycled repeatedly after accumulating data, and the analog signal re-constructed by a D$\rightarrow$A converter. Such "freezing" of acquired events in this way for stable oscilloscope viewing may be of high value in test proceedures, but is not necessary in a running experiment (such could be done via computer display).


Shaped Anode IN
50 Ω
2V
V-
+V
Signal IN
FADC
Anode
V_R^+
V_R^-
Strobe IN
Overfl. Gated Data
driver
Anode Range adj.
-5V
V-
LED monitor Anode overflow
Readout Logic [Overflow Anode]
φ [out]
SET
Overflow Latch [Anode]
Re-set
Gated CLK out
Buffer
6-bits Data IN
6-bit x ≥1K 100mHz memory array
6-bit Data
Readout Logic
Anode digitized out
WE
Switch select Int./ext. clock
EXT. CLK. IN
CLOCK CONTROL LOGIC
Select internal/external clock, Start internal clock upon receipt of "gate" pulse; Transmit external clock upon first edge after gate pulse.
Clocking continues through gate duration, when gate falls, clocking is stopped.
Buffers
START
Gate IN
gated clk. out
CLK IN
Address Counter
Addr. IN
Address advance Readout Logic
Address out
Readout Logic
Current Address
Readout logic "Terminal CT."
Stop count select
2K 1K 512 256
Counter outputs
Counter preset
Readout Logic
Clear
Reset
SET
"Busy" Latch
φ [out]
LED monitor module "busy"
Readout Logic "Busy"
Trig.
Monostable [Acquisition End]
LED monitor "data present"
Readout Logic "data present"
VETO
SET
"Data Take" Latch
φ [out]
[Gated start]
Re-set
Readout Logic CLEAR IN
Readout Logic CLEAR out
LED monitor ΔPW overflow
Reset
Readout Logic [ΔPW overflow]
φ [out]
Overflow Latch (ΔPW)
SET
+5V
+V
ΔPW Range adjust
Shaped ΔPW IN
50Ω
2V
driver
Strobe IN
V_R^+
V_R^-
FADC ΔPW
Signal in
Overfl
R
R
X(-1)
-V
Addr. IN
WE
6-bit x ≥1K 100mHz memory Array
Readout Logic
6-bit data
ΔPW digitized out
6-bits
Data IN
Buffers

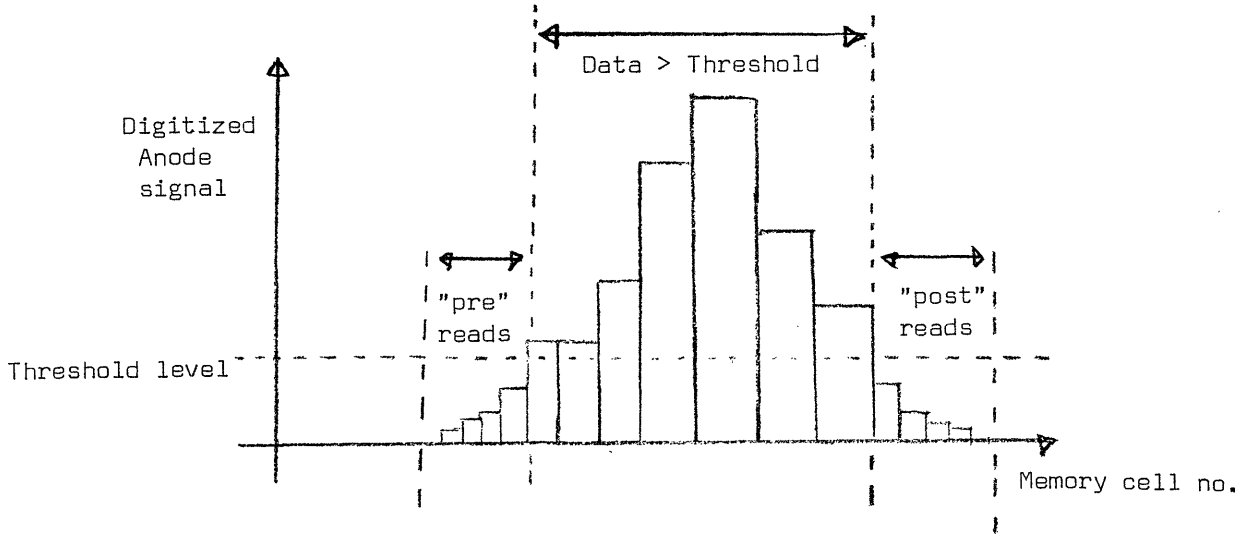
b) Readout Control Circuitry

Because of the high speed requirement, the acquisition circuitry sketched previously must be based around ECL logic. The readout control circuit described here may employ conventional TTL (thus benefit from its simplicity) and run at ≈20 Mhz (zero-skipping processing, ie. memory scan processor), thus require a minimum of 50 nsec. to scan 1K of memory (not counting CAMAC cycle time required for transfer of selected data, which may be done asynchromously). A faster rate (approaching 50 Mhz or better) can be obtained by also constructing the memory scan processor out of ECL.

The most straightforward and necessary hardware post-processing is the elimination of "zeroes" in the data. If the total memory is scanned and read over CAMAC, the time required (at $\leq$ 1 Mhz) will be considerable. Since the majority of memory locations should contain "zeroes", "clusters" with anode amplitude over a pre-programmed threshold can be seperated and transferred through CAMAC, appreciably reducing data transfer time.

It has been suggested that other post-processing tasks may be accomplished in hardware; division of ΔPW by the anode, "zero-crossing" fits, etc. These items probably require a μ-processor (the only "division" chips readily locatable seem to be CMOS APU's, which are slow and certainly overkill; ROM lookup tables are possible, but may become too sizable to dedicate 1 per wire configuration, to say nothing of the necessity of offset subtraction before taking the quotient, etc... the required hardware increases greatly, and the loss of software flexibility is disadvantageous). μ-processors also suffer from speed limitations; higher speeds are available with bit-slicing configurations, but the circuit size is certainly too large for 1 μp/wire configuration to be realistic. A more practical scenerio is suggested here: the basic flash ADC CAMAC module will consist of 2 flash ADC channels with ECL controller (as sketched earlier), and hardwired zero-skipping readout (ADC values from both channels with memory address are to be packed into one 24-bit word). The compacted data may be transferred through CAMAC to a dedicated central processor array, which can perform further analysis.

Two registers will contain values needed for the zero-skipping logic. One ("threshold" register) holds the threshold on the anode signal below which data is considered "zero". The other ("pre/post" register) contains a value which indicates the number of memory cells before the first $\geq$ Threshold value to start readout transfer and the number of memory cells after the last value $\leq$ Threshold to cease readout transfer. To be more detailed, the data should consist of a series of pseudogaussian "clusters". The threshold may be set to detect the "clusters", but in order to record the rise and tail of this "cluster" data must be output both before and after the threshold is exceeded; the pre/post value determines the number of cells to output before and after threshold detection. (see sketch below)



Of course the anode signal may generally consist of a series of "clusters", thus overlap logic is necessary. If other cells > Threshold are detected while transferring "post" cells after a previous "cluster", the "post" count is re-set, and "post" cells are counted from the point where the data again falls below threshold. When subtracting the "pre" value from the address where a cluster rises above threshold, a check must be performed to insure that the starting address of data transfer does not overlap with data output from a previous "cluster". If this overlap is detected, data transferral will start after the last address output.

Using this method, only one pass is needed through the ADC memory (threshold comparison is done only on the anode signal); memory backspacing when subtracting the "pre" value from the address of the > Threshold data may be accomplished via the counter preset bits.

The proceedure described above may be implemented with fast, conventional binary adders and magnitude comparators; no μ-processor is needed. To liberate the memory scan logic from the slower CAMAC read speed, a buffer memory may be employed. As soon as data acquisition is completed (ie. "Data-present" goes high), the memory search is begun, and after the first word is transferred into buffer memory the CAMAC controller can be queued (via F8 Q-test) to begin readout (via F2). Since the memory scan logic should be much faster than CAMAC transfer, a wait-for-valid-data during readout should not generally occur with a sufficent buffer size (≈10 24-bit words),however if the CAMAC read does beat the memory scanner (as perhaps with an excessive memory gap between "clusters"), the "scanner busy" state can be acknowledged via responding Q=0 to an F2 read request. If the Q bit is reserved for a "stop-on-Q" read, an X=0 could be used to signal "scanner busy".

When the end of ADC memory is encountered, the scanner constructs a "termination" word (consisting of the 2 "overflow" bits set during readout, and flagging a "1" at bit 24) and loads it into the CAMAC buffer; thereupon the acquisition logic is cleared and further data can be taken. The readout proceeds until the "termination" word is read; if the acqusition system has taken another event while the buffer is being read to CAMAC, the next event will be be loaded in "after" the termination word of the previous event (a rotating buffer can be used with two pointers, one showing the location of the current write from acquisition memory, the other showing the location of the next CAMAC read; overwrite protection is necessary). If the CAMAC buffer is empty, further F2's respond Q=0.

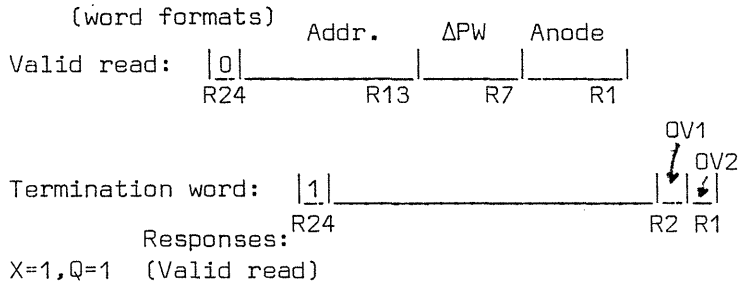
If a CAMAC F9 or VETO signal is applied, the memory scanner is halted and reset (a special termination word can be constructed flagging "VETO during readout" to seperate data written from such events in the CAMAC buffer).

SUMMARY:

Required CAMAC commands:

F2 : Read next CAMAC buffer word

(word formats)



Responses:

X=1,Q=1 (Valid read)

X=1,Q=0 (Buffer empty, no memory scan in progress)

X=0,Q=1 (Buffer empty, scan in progress)

F8: Test Acqusition Busy
Responses:

X=1,Q=1 (Data present, acqusition not in progress)
X=1,Q=0 (Acquisition in progress)

F9·A(0): Clear and re-set entire module (incl. CAMAC buffer)
Response:

X=1,Q=1

F9·A(1): Clear and re-set acqusition logic and memory scanner. Leave CAMAC buffer intact.
Response:

X=1,Q=1

F16·A(0): Write anode threshold register (6 bits)
Response:

X=1,Q=1

F16·A(1): Write pre/post register (only ≃4 bits necessary, one value can be used for both "pre" and "post" counts).
Response:

X=1,Q=1

COMMENTS:

The readout scheme described above entails the minimum hardware necessary for a fast transfer of non-ambiguous data. If further processing is included (divide, etc.), the hardware will increase considerably, taking up vast amounts of CAMAC space. A system such as this only compacts data, making rapid transferral to an external array of dedicated processors possible. Additional CAMAC options (LAM, etc.) can be easily designed into this system, but are not required for the envisioned operation. A two-slot CAMAC module should be able to hold all required components for readout and acquisition circuitry. By employing nearly inhuman cramming ability, it might be possible to fit it all into a single width module.
The 11-bit address used here enables a maximum of 2K words acquisition memory.

C) Cost Breakdown

We assume costs of ≃SF250./piece for 100 Mhz Siemens SDA5010 flash ADC's in quantities of ≃100 (≃150./piece for the SDA6020 50 Mhz). Hitachi HM2112-1 1 X 1K ECL memory (8 nsecs. address access) runs SF50./piece for ≃100 quantities. Below a systematic breakdown is given for 1 CAMAC module as described, handling 1 anode and 1 ΔPW signal. The 6-wire configuration considered for the LEP and DESY detectors will require 2 of these modules.

ITEM		COST IN SF:
Flash ADC (SDA5010):	2 X 250.	500.
Memory (HM2112-1):	12 X 50.	600.
ECL Logic and analog ckts. for acquisition:		300.
Readout and memory scan system:		500.
G10 PC board(s) through plated:		100.
CAMAC module, mechanics:		50.
TOTAL		SF 2050.

Thus approx. SF 2000./module, not including CAMAC crates or μ-processor support. If the Siemens SDA6020 is used rather than the SDA5010 (it has been clocked at 100 Mhz, Ref. V. Farr, Heidelberg), only ≃SF200., or 10% of the total is saved. This estimate is extremely approximate (in general ECL and TTL costs), and depends heavily upon actual design, but it most probably will go up rather than down. Labor costs are not included.