AMS TRACKER

Test DPNC August 2005

Tracker Data Reduction

Software Version 4

Montpellier, September the 2th
Claude.Zurbach@lpta.in2p3.fr

Summary of the data reduction module version 4

Name: version 4 release 0 (V04_R0).

General description:

Step 1 – readout, **pedestal substraction and reordering** for each amplitude (1024 channels)

Step 2 – **Common Noise** calculation VA by VA, excluding channels with flag noisy or dead.
Backtrack procedure:

- back to the first channel on VA
- CN subtraction
- search for cluster candidates with high threshold

After backtrack procedure: selected channels are memorized in a table as "candidate" for cluster (at this step, the program does not discard noisy channels.

Step 3 – **cluster identification**, exploiting the table of candidates from the second step; start address and length of clusters (one or more) are memorized in a new table; for each cluster, the program includes systematically the right and left channels, so each cluster is built of at least three channels.
The noisy channels do not trigger the cluster identification, but may contribute to the cluster.

Step 4 – **cluster writing** on output, from table of clusters produced in step three.

Main differences with Version 1

Optimization: the program respects the constraint of less than 14000 DSP cycles (conditionned by the trigger rate of 2 KHZ) for the event processing (less than 280 µs): exploiting in best way the DSP capacity to manage two or sometimes three instructions in the same cycle, and with a better design of the module

Input: apart tables with pedestals, and flags, the calibration module (V4_R0) produces two tables on output fed to the reduction module: low and high thresholds table for the cluster identification; therefore, the thresholds are no more computed in TDR

Reordering data: the program subtracts the pedestals values during the reordering phase, and no more in CN calculation as in the first version; that allows to gain ~1000 cycles

CN calculation: the program don't exclude any more the channels with a high threshold, but only the channels with a flag dead or noisy

Candidates identification: in version 1, candidates could be forgotten in case of a negative VA: this problem was fixed in the fourth version with a systematic research for candidates but after CN subtraction (Cf the backtrack mechanism); in this step the program uses the High Threshold table

Cluster identification: in version 1, the candidates were memorized after CN calculation with a shift of one position; this was not a problem in the case of a cluster with more than one significant channel, but in opposite case a cluster could have been lost; this problem was fixed in the fourth version; here, the program uses the High and Low Threshold tables

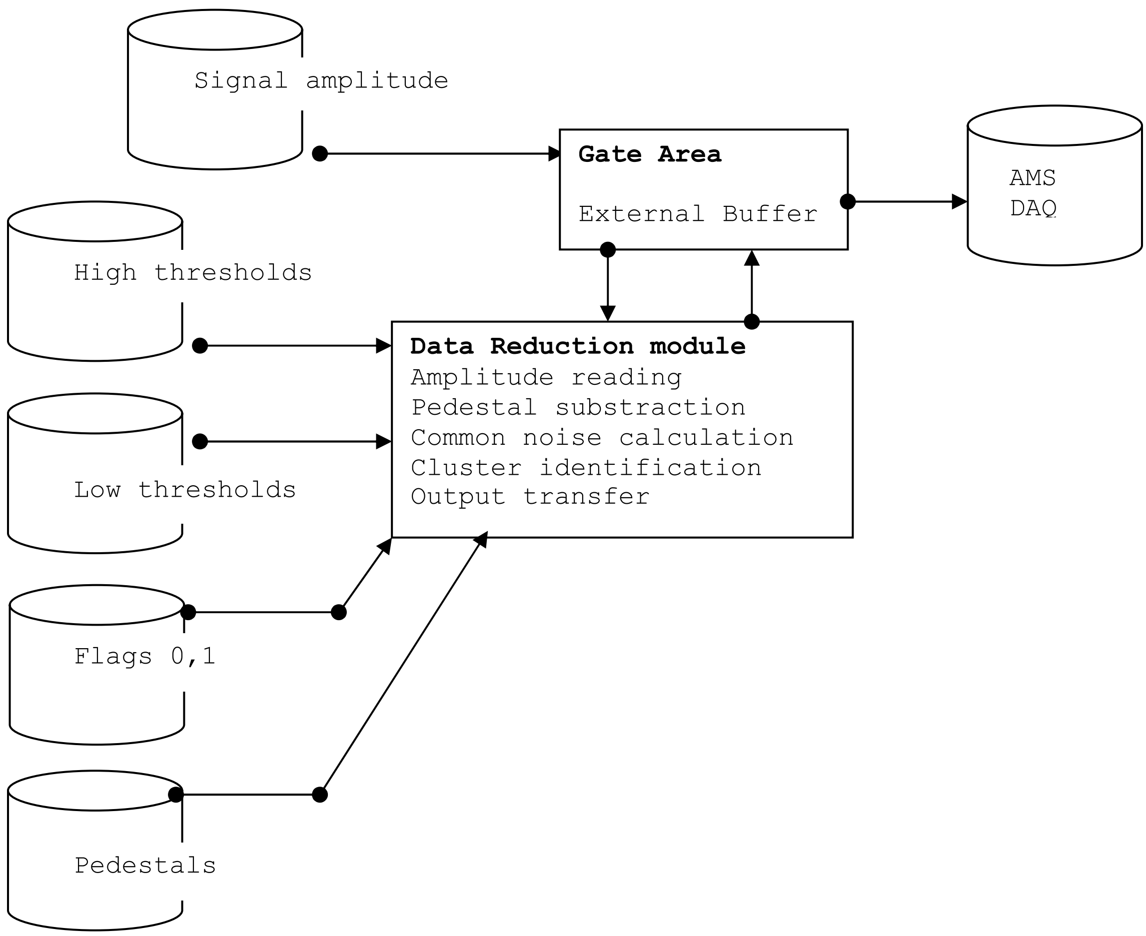
Input and output data

Input data are coming from:

- Calibration module – High and Low Thresholds, Pedestals, Flag tables
- Gate Area memory – Amplitude in which we search clusters

Output data:

- sequentially for N clusters: the length of the cluster, the address of the first channel, the channel values
- temporarily, the program adds to its output the CN values of 16 VA
- the total length of buffer



Precision, flags and reordering Data

All the tables in input contain 1024 values (Cf documentation of the calibration module), and are calculated with a precision of 15 bits.

<u>Precision</u>: events are usually numerized on 12 bits by the ADC (Analog to Digital Converter) in output of the VA; this value is multiplied by 8 to exploit the full capacity of the TDR registers (15 bits and 1 bit for the sign). This gain of precision is acquired before sending the data to the Gate Area, and the reduction module uses data only with 15 bit precision.

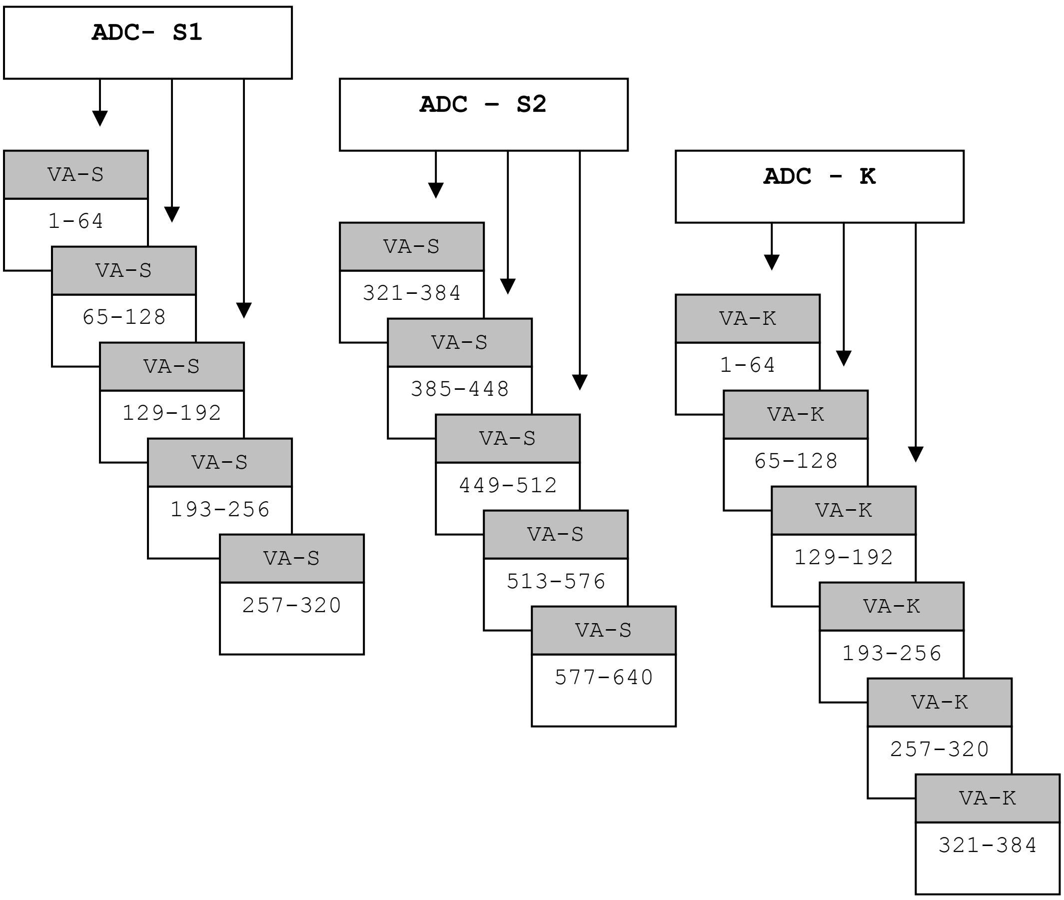
The <u>Flag table</u> contains the quality flags of the channels:

Bit 0 to 1 (0x1): dead channel on the sigma raw base
Bit 1 to 1 (0x2): noisy channel on the sigma raw base
Bit 3 to 1 (0x4): dead channel on the final sigma base
Bit 3 to 1 (0x8): noisy channel on the final sigma base

Nevertheless, the program verifies during the common noise calculation if the value for the channel is 0 or not (0 means valid).

<u>Reordering</u> problem of data from the Gate Area:

As we can see in the next scheme, data arrive in a desorganized way, in the opposite of the tables coming from the calibration module. For coherence, we need in a first step to reorganize data from channel 1 to 1024.



ADC provides: S1, S2, K, S1, S2, K, ……, K,K,K
For example: channel 1, channel 321, channel 641, channel 2, channel 322, channel 642, channel 3, channel 323, channel 643 etc…

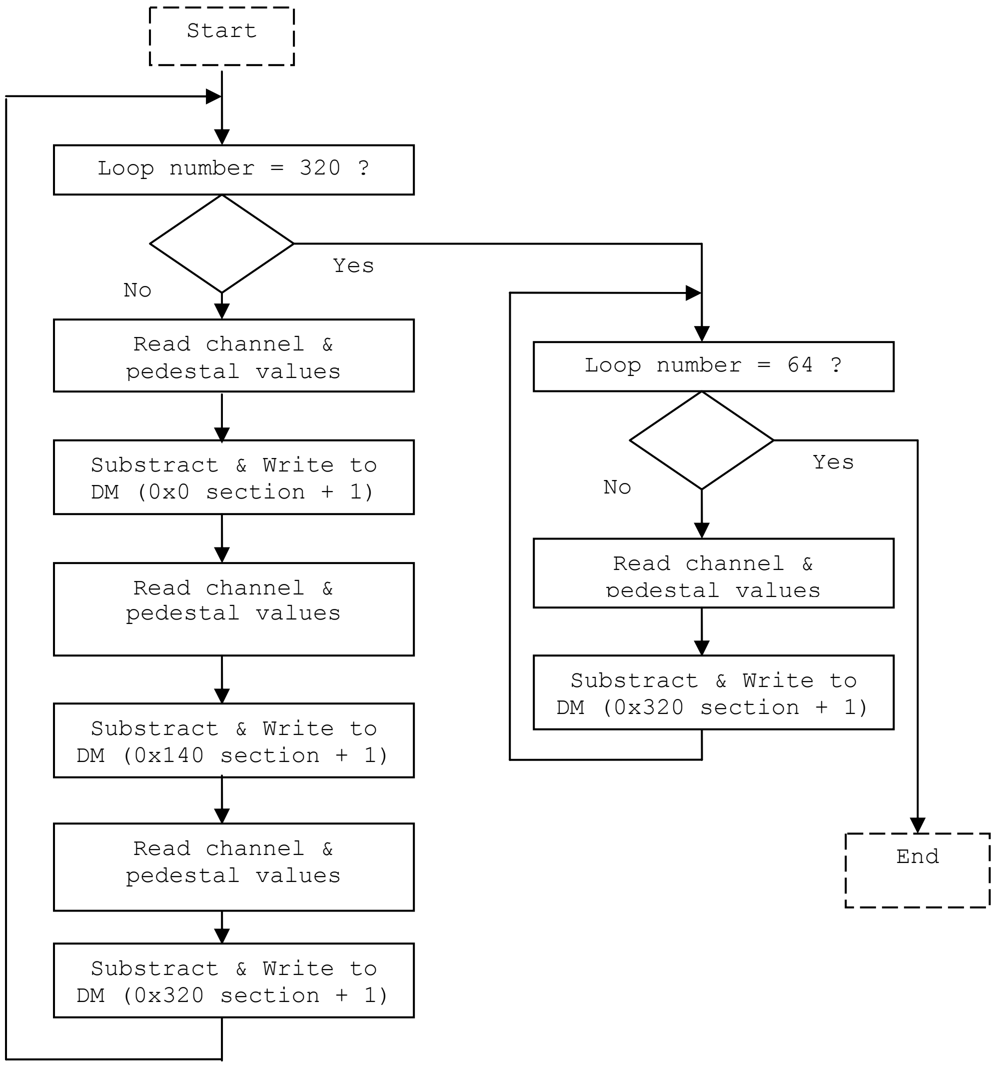
Module description

First step: reading and reordering

Input: each channel value, on 16 bits (15 bits for value and 1 bit for sign), from Gate Area

Subtraction: substraction of the corresponding pedestal values

Output: in TDR data memory DM, on address 0 to 1023 (0x0 to 0x3FF in hexadecimal); first writing at DM address 0x0 (0), second writing at DM address 0x140 (320), third writing at DM address 0x320 (640)



Algorithm

Read 3 channels values 320 times and the corresponding pedestal values, then, after substraction of the pedestal value for each channel, write result in first, second and third section of Data Memory, then add 1 to the index registers for writing. Channels values are qualified as V2 (values after pedestals subtraction).
At the end of the 320 loops, we get 960 reorganized values in DM. It’s necessary to read at the end last VA on the K side.

Second step: Common Noise calculation and candidates identification

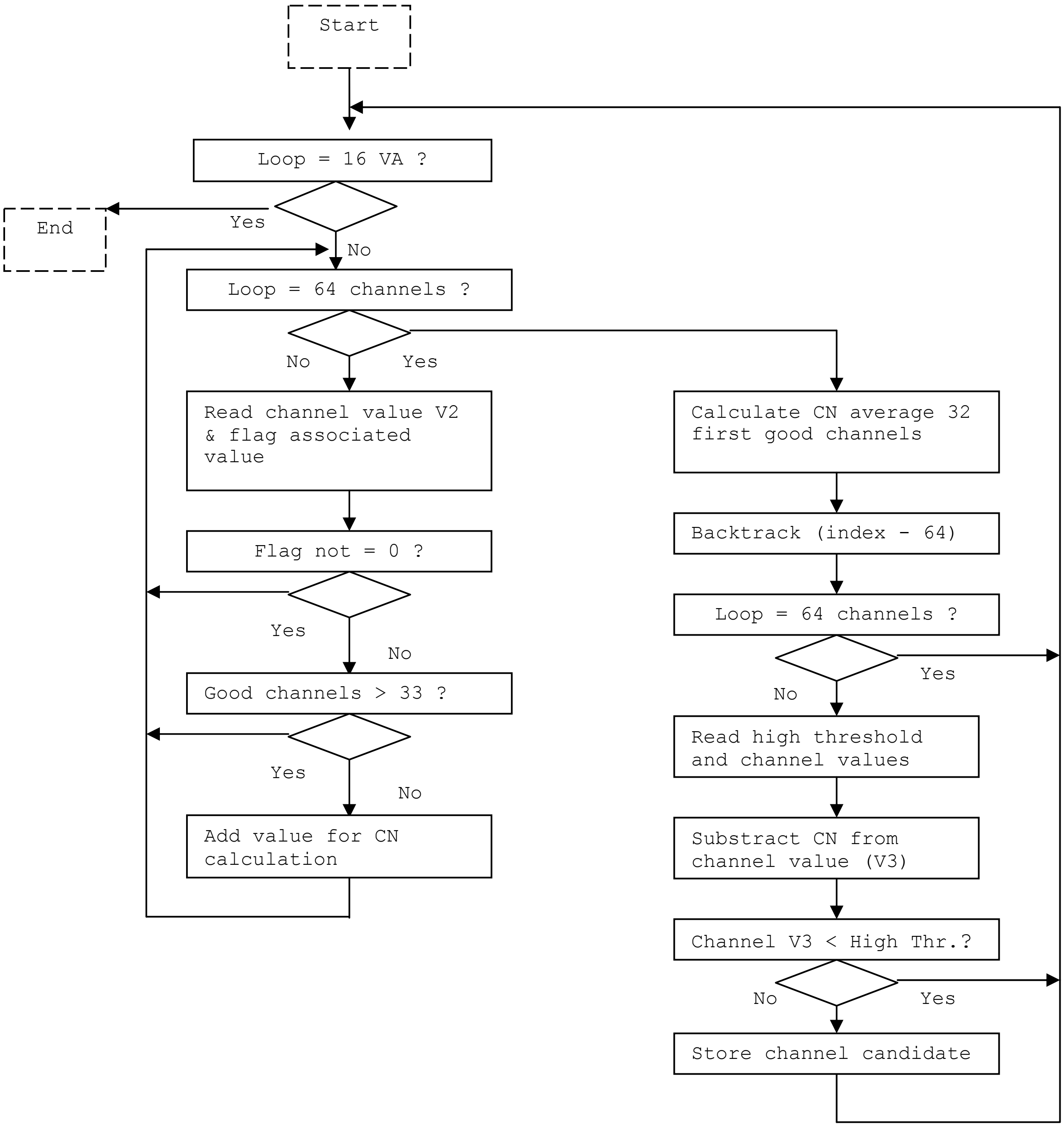
Input:

- channels values V2 in TDR data memory (DM)
- flag table
- high threshold table

Output: in TDR data memory (DM)

- common noise table
- candidate table

Recall: for time of CPU optimization, the average of a VA is calculated on a base of 32 good channels.



Algorithm

Startpoint

Test if end 16 VA.
If not, calculate common noise on a VA.

Channel readout

If end of VA go to average calculation.
Read channel value (V2) and corresponding flag.

If corresponding flag not = 0 (dead or noisy) jump this channel and go to a new channel lecture. If flag = 0, go to next test.

Channel values addition

If cumulated good channel number exceeds 32, go to channel readout. If not, add channel value to the sum and go to a new channel lecture.

Calculation of the average(common noise for the VA)

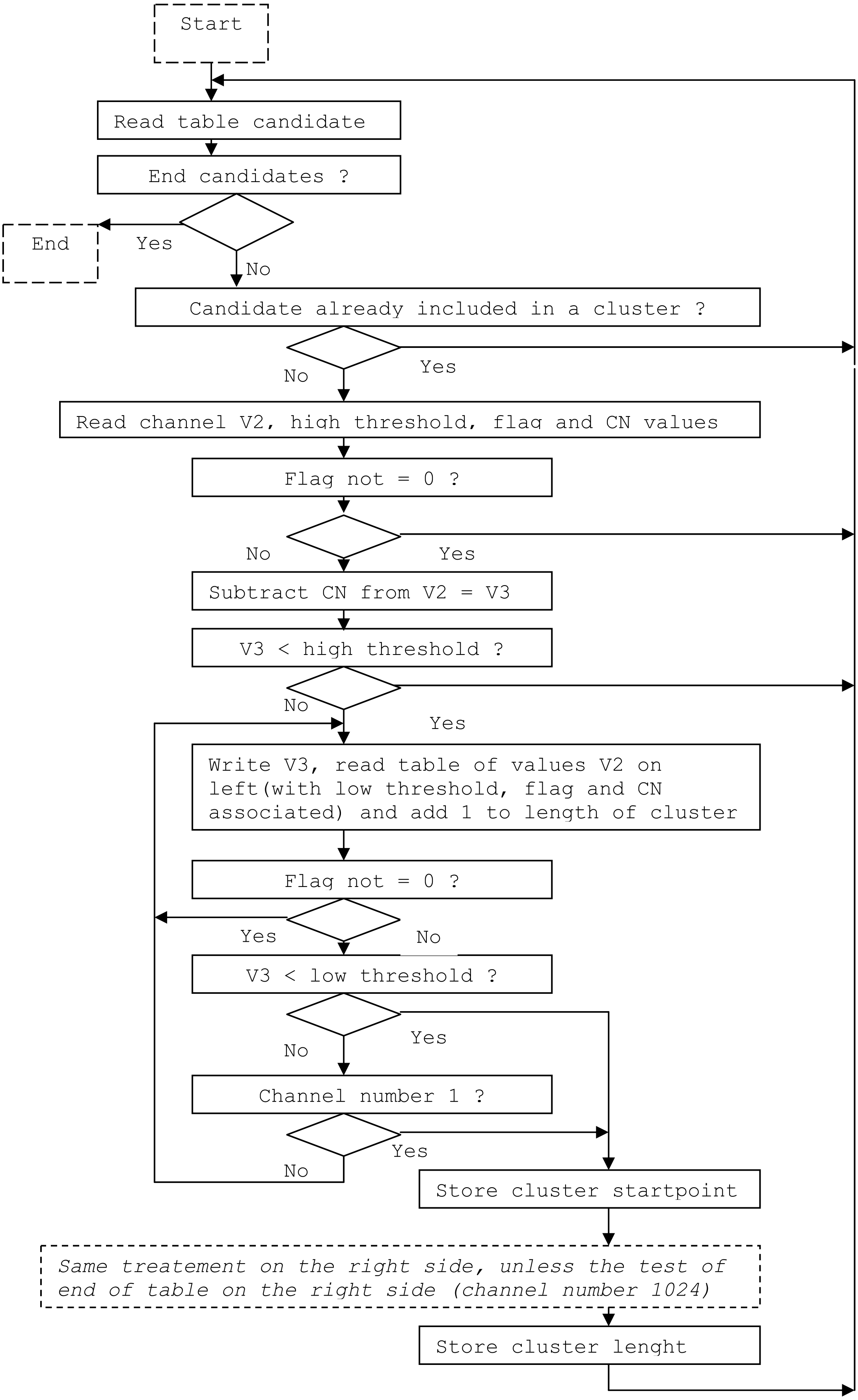
CN = Sum divided by 32.

Candidate identification (Start channel reading with a shift of 64)

Read channel and corresponding high threshold value.
Substract CN from channel V2 (giving V3).
If high threshold less or equal to channel V3, store the channel address in the table of candidates.

At the end of VA go to Startpoint.

Third step: Cluster identification



Reading:

- candidate table in TDR data memory (DM)
- channels values V2 (in DM)
- flag table
- sigma table

Writing: in TDR data memory (DM)

cluster table (address cluster 1 + length, address cluster 2 + length etc…)

Algorithm

Input table of candidates

Verify if the candidate is already included in a cluster (exploiding the address and the length of the last cluster). If yes, go to an other candidate (end of table: value 0x2000).

Read high threshold, flag, CN and channel value V2 associated with the candidate.

If flag candidate not = 0, go to Input table of candidates

If V3 (V3 = V2 minus CN) less than high threshold go to a new candidate.

Backward scan

Write V3 value in place of V2. Add 1 to the length of the cluster.
Read next channel value V2, low threshold, flag and CN associated with the channel. If flag not = 0, go to Backward scan.
If channel number 1 (left end), then store the startpoint of cluster and go to the reading of the right side.

If V3 (V3 = V2 minus CN) equal or greater than low threshold, add 1 to the length of the cluster and go back to read an other channel on the left side.

Store the address of the candidate and go one step to the right side.

Forward scan

Add 1 to the length of the cluster.
Read channel value V2, low threshold, flag and CN associated with the channel. If flag not = 0, go to Forward scan.
Write V3 value in place of V2. If channel number 1024 (right end), then store the lenght of the cluster and go to read a next candidate.

If V3 equal or greater than low threshold, add 1 to the length, and go back to read a next channel on the right side. If not, store the length of the cluster and go back to read an other candidate.

Fourth step: Cluster output

Input:

- cluster table in TDR data memory (DM)
- channels values V2 and V3 (in DM)

Output: in Gate Area

- sequentially for N clusters: the length of the cluster, the address of the first channel, the channel values
- CN values of 16 VA,
- the total length of buffer