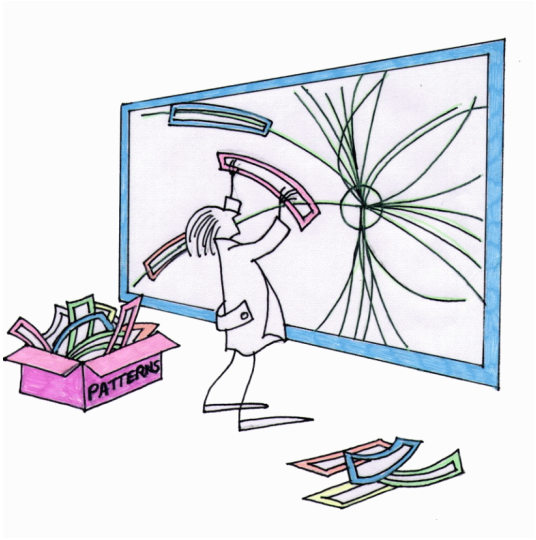


FTK

Fast TracKer for hadron collider experiments

324318, FP7-PEOPLE-2012-IAPP

Deliverable D5.1:

FTK Infrastructure for Imaging

Deliverable Version:	D5.1, v.1.0
Document Identifier:	FTK - D5.1_Imaging_Infrastructure.docx
Preparation Date:	March 30, 2015
Document Status:	Final
Author(s):	S. Citraro (UNIPI), P. Giannetti (FTK collaboration), C.-L. Sotiropoulou (UNIPI), P. Luciano (UNIPI) (UNIPI),
Dissemination Level:	RE - Restricted to a group specified by the consortium (including the Commission Services)

Project funded by the European Community in the 7th Framework Programme - People

Deliverable SUMMARY SHEET

Deliverable Details	
Type of Document:	Deliverable
Document Reference #:	D5.1
Title:	FTK Infrastructure for Imaging
Version Number:	1.0
Preparation Date:	March 30, 2015
Delivery Date:	May 30, 2015
Author(s):	S. Citraro (UNIPI), P. Giannetti (FTK collaboration), C.-L. Sotiropoulou (UNIPI), P. Luciano (UNIPI)
Document Identifier:	FTK - D5.1_Imaging _Infrastructures.docx
Document Status:	Final
Dissemination Level:	RE - Restricted to a group specified by the consortium (including the Commission Services)

Project Details	
Project Acronym:	FTK
Project Title:	Fast TracKer for hadron collider experiments
Project Number:	324318
Call Identifier:	FP7-PEOPLE-2012-IAPP
Call Theme:	Industry Academia Partnerships & Pathways (IAPP)
Project Coordinator:	Università di Pisa (UNIPI) – IT
Participating Partners:	AUTH, CAEN, CERN, LPNHE, PRIELE, UNIPI
Instrument:	CP-FP
Contract Start Date:	February 1, 2013
Duration:	48 months

Deliverable D5.1: Short Description

Keywords: Simulation; Test; Demonstrator;

Deliverable D5.1: Revision History

Version:	Date:	Status:	Comments
1.0	10/09/2013	Final	Author: S. Citraro (UNIPI), P. Giannetti (FTK collaboration), C.-L. Sotiropoulou (UNIPI), P. Luciano (UNIPI)

Table of Contents

List of Figures

:

Abbreviations and Acronyms

Executive Summary

We have decided the global architecture of the system we will use for our research in the "Imaging" sector, developing the WP5 work plan. Here we describe the algorithm, its planned implementation and the chosen hardware.

1 Introduction

The use of the Associative Memory (AM) [1] processor for brain studies is particularly fascinating. The most convincing models that try to validate brain functioning hypotheses are extremely similar to the real time architectures developed for HEP. A multilevel model seems appropriate also to describe the brain organization to perform a synthesis certainly much more impressive than what done in HEP triggers. The AM pattern matching has demonstrated to be able to play a key role in high rate filtering/reduction tasks. We can test the AM device capability as the first level of this process, dedicated to external stimuli pre-processing. We follow the conjecture of reference [2]: <<the brain works by dramatically reducing input information by selecting for higher-level processing and long-term storage only those input data that match a particular set of memorized patterns. The double constraint of finite computing power and finite output bandwidth determines to a large extent what type of information is found to be "meaningful" or "relevant" and becomes part of higher level processing and longer-term memory.>>

The AM-based processor will be used for a real-time hardware implementation of fast pattern selection/filtering of the type studied in these models of human vision and other brain functions.

Figure 1: natural images and corresponding filtered images

The second row of figure 1 shows the quality of the image when filtered accepting only the "relevant patterns" to be stored in the AM. The associative memory works as an edge detector algorithm able to extract the salient features. The extracted features can be processed with fast but complex reconstruction algorithms implemented on FPGA. The reduction of execution time of image reconstruction to be applied after the AM filtering function, would exploit the computing power of parallel arrays of Field Programmable Gate Arrays (FPGAs) as we do in the FTK [3] project to find clusters of contiguous pixels above a certain programmable threshold [4].

2 The filtering Algorithm

We plan to use the AM to extract object contours. With the use of an FPGA we can build a small coprocessor which will make a huge improvement in this kind of computation, by achieving both good efficiency and low power consumption.

Different computing phases are necessary.

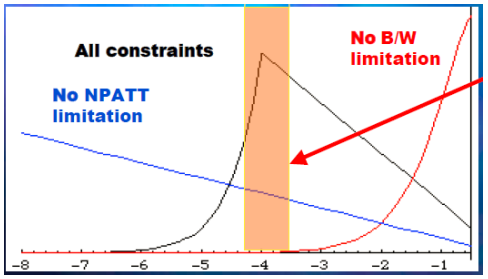
The first important task is the **Training Phase.** It is subdivided into many steps:

1. **Accumulating pattern appearance frequencies:** The embedded system receives the image bit-streams (e.g., data from a PC or a video camera). The FPGA partitions/reorganizes the input data into small patterns (3×3 square pixel matrices in black-and-white (B/W, 1-bit depth) for static images, 3×3×2 to use four levels of grey in static images, and 3×3×3 cubic pixel matrices, 3 black-and-white frames taken at 3 subsequent instants, for movies. Then, for each pattern, the FPGA calculates the occurrence of the analysed pattern in the processed images/frames. This calculation is iterated for all possible patterns in a large set of training images. The total number of possible patterns is $2^9 = 512$ for static image and $2^{27} = 134$ Mpatterns for movies, if B/W is used. 2^{18} is the total number of patterns for static images using 4 levels of grey. In this way, different Probability Density Histograms (PDHs) are computed for different training image sets. Of course, PDHs are different for different type of images. Certainly medical images have different PDHs than natural images.

2. **Pattern selection:** In this step, the system must decide which set of patterns has to be selected for memory storage (in the AM bank). To maximize the capability to recognize shapes (both human-brain recognition and artificial recognition), we adopt the hypothesis described in Del Viva's paper [2], i.e., that the principle of maximum entropy is a measure of optimization. The set of patterns that produces the largest amount of entropy allowed by system limitations is the best set of patterns that we can select to filter our images or videos. System limitation can be summarized in two main parameters: N maximum number of storable patterns, and W maximum bandwidth. From Del Viva's paper, the entropy yield per unit cost and for each pattern is given by:

$$f(p) = \frac{-p\log(p)}{\max(1/N, p/W)}$$

p is the probability that a given portion of the input data matches a specific pattern, N is the maximum number of storable patterns and W is the maximum allowed total rate of pattern acceptance. Fig. 2 shows three entropy yield functions for different values of W and N. Infinite memory space (N=∞) favors low probability patterns, while infinite output bandwidth favors high probability patterns. Real constraints (N and W finite) produce a peak that indicate in which interval are clustered the "relevant patterns".



Patterns that are efficient carriers of information given the bandwidth (W) &memory limits (N),

Figure 2: Entropy yield per unit cost, as a function of the pattern probability. Blue curve: limited bandwidth and unlimited pattern storage capacity (W=0.001, N=∞); green curve: limited storage and unlimited bandwidth (N = 100, W=∞); red curve: limited bandwidth and storage (N=100, W=0.001)

Fig. 3 shows an example of pattern selection performed on a natural digitized image representing a butterfly. It can be seen that the filtered images (b) and (c) obtained with the selection of "relevant patterns" are recognizable, while in image (d) filtered with low probability patterns, the important features are lost.

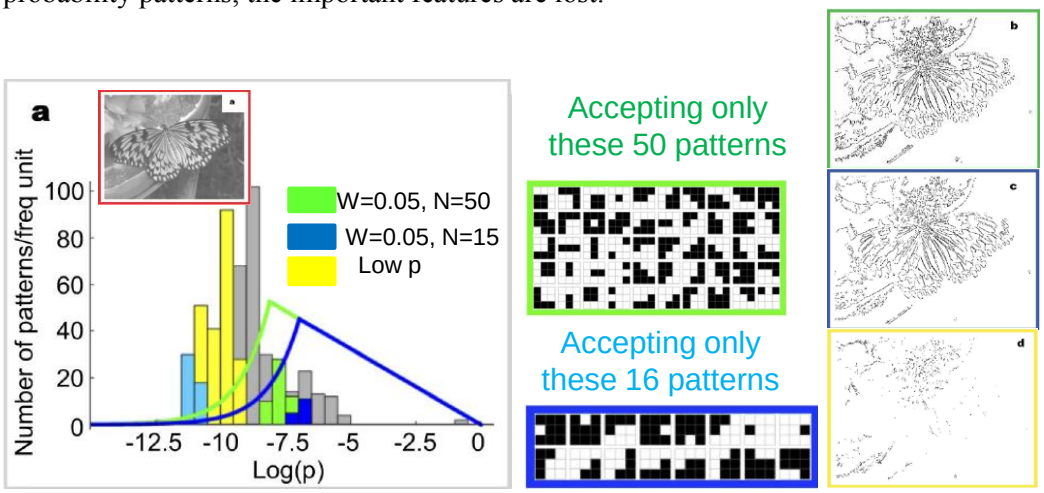


Figure 3: On the left (plot a) is the probability distribution of the $N_{tot}=2^9$ possible 3×3 square pixel matrices in black-and-white (1-bit depth) for natural images. Blue curve and histograms is obtained setting W=0.05, N=15; green curve and histograms with W = 0.05, N=50; on the center is the visualization of the pattern sets shown in (a), in green and blue respectively (resembling to edges, bars, or end-stops in several orientation); on the right are the results of the filtering applied to the butterfly image shown in (a) filtered with green patterns (b), blue patterns (c) and yellow not relevant, low probability patterns (d).

3. **Writing operation:** The relevant pattern (selected in the second step) is written in the AMchip bank. The writing operation is made via JTAG by means of a system controller.

Real-time pattern recognition: After training, selection and writing operations, the system is able to work in real-time at the maximum working frequency and is able to perform:

1. **Parallel recognition of patterns** in the data stream. The filtered pattern is recognized by the AM chip (through a bit-wise comparison) and the pattern address is transferred at the output from the AM chip.
2. **Output formatting operation**: The resulting patterns are reorganized into a new compressed image, to produce the filtered images/videos, called "sketches", where only the boundaries of the relevant objects are kept.

3 The Hardware for the Implementation

Our parallel processing architecture will be based on hardware currently used in High Energy Physics (HEP), in the FTK processor [3] approved for a trigger upgrade by the ATLAS experiment at the Large Hadron Collider (LHC) at CERN. FTK reconstructs the events produced at LHC, extremely complex images, in few tens of microseconds and therefore is extremely suitable to tackle the Big Data challenge. The key role in the novel technology is played by the custom multicore AM system that will intensively be used to filter out the relevant information of the data to be further processed by Field Programmable Gate Arrays (FPGAs) executing higher level algorithms. The AM implements the maximum parallelism and offers the best timing performances, since it solves its task in the time data are loaded on it. Additionally our architecture benefits from the FPGA computing power. The FPGA complements the AM task with its flexibility and configurability, adapting its logic to perform any necessary "refining processing" of the AM result. In addition the AM has a specific architecture (the pattern matching can be successful on a subset of the tested data) that helps to identify not only perfect features, but also partial, or noisy versions of them.

A first approach is to adapt exactly the same hardware developed for FTK in ATLAS to be used for generic imaging. The HW is briefly described below. To simplify input/output operations to/from the AM board, in FTK the AM chips are grouped into AM units composed of 16 chips each, called Little Associative Memory Boards (LAMB, Figure 4). The ASIC prototypes in the figure are AM05 developed in 65 nm VLSI-technologies (Very Large Scale Integrated Technologies) and containing 2048 patterns each one. It has serial link buses for all data paths. The newly developed chip, AM06, will allow the storage in the memory bank of 128 Kpatterns per die.

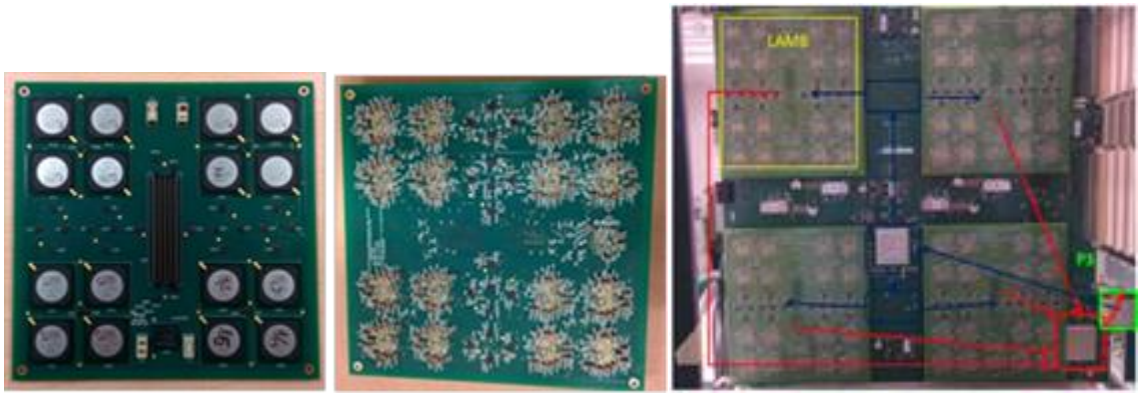


Figure 4: LAMB (left & center) and AMB with 4 LAMBs (right) inserted in the VME crate.

A 9U-VME motherboard has been implemented to hold 4 such units (see figure 4 on the right). The LAMB and the motherboard communicate through a high frequency and high pin-count connector placed in the center of the LAMB. A network of high speed serial links made of ~750 point-to-point connections handles the data distribution to the 64 AM chips (blue arrows) and collects the output (red arrow). The data traffic is handled by 2 Artix-7 Xilinx FPGAs with 16 Gigabit Transceivers (GTP), each providing ultra-fast data transmission. Two separate Xilinx Spartan-6 FPGAs implement the data control logic.

The VME solution is very powerful and offers a lot of computational power but it is large (not portable) and requires a specific interface (VME crate). It is very little user friendly. We have decided to also try a more modern, compact solution. One of our key goals is the miniaturization of the system in new modern standards with the objective to make it suitable for an open range of applications in which massive and parallel data processing is key.

The new Control Board will have the following characteristics:

- powerful FPGA (Field Programmable Gate Arrays) with large on-board memory,
- Ethernet & PCI Express I/O,
- handling (distribution and collection) of all AM chip serial links,
- configuration and control of the AM pattern bank,
- provision of extra functionality to complete the AM functions in real time.

While the AM chip needs challenging developments, one of the advantages of the FPGA imaging task is that boards already available on the market are powerful enough to cover the above listed specifications. New generation commercial FPGAs are already available (e.g. Xilinx Ultrascale FPGAs) and will allow us to develop the high performance embedded system required in parallel with the development of the new generation AMchip.

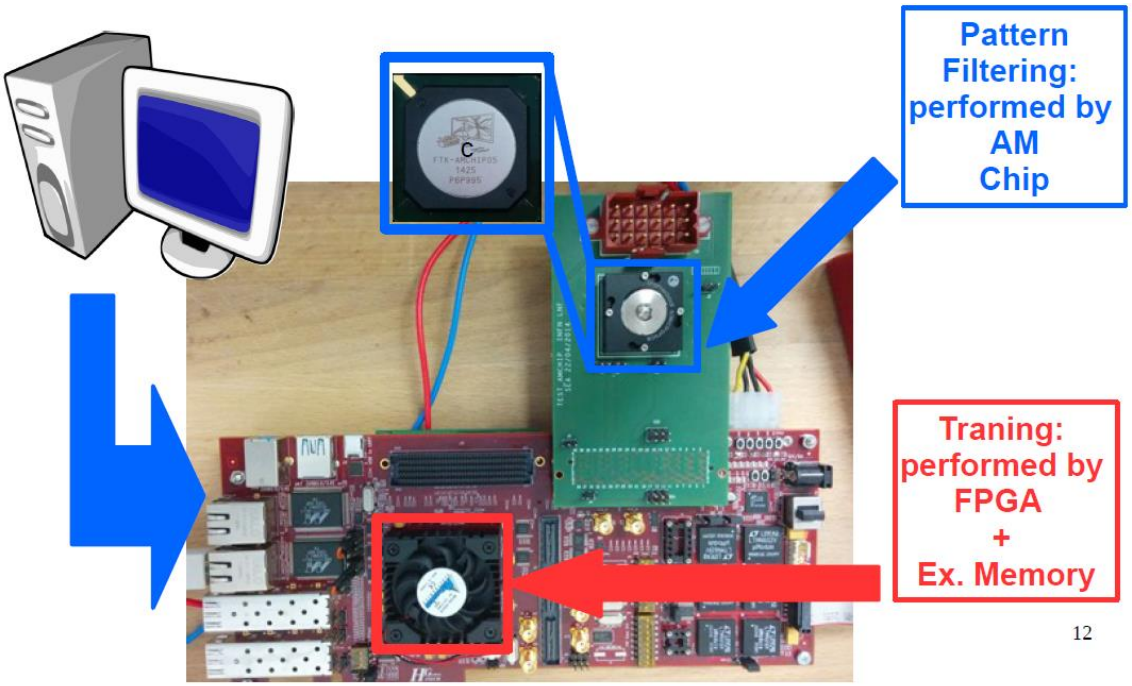


Figure 5: the hardware setup for static images processing

Figure 5 shows the chosen computing unit, based on a Xilinx Virtex™ 6 PCI Express Gen 2 / SFP+ / USB 3.0 Development Board. A new LAMB-like mezzanine can be connected to the large connector on the top of the board, but for the moment we start with a single chip mezzanine as shown in the figure.

All the functions described in section 2 will be executed by the FPGA with the only exception of **Parallel recognition of patterns.** As described in section 2 the total number of possible patterns for static image is 2^9 = 512 if B/W is used and 2^{18}= 256 k if 4 levels of grey are used. The simulation of the algorithm has shown that for the B/W case a very little number of patterns need to be stored in the memory (between 50 and 15 in figure 3) while for the 4 levels of grey less than 1% of the patterns (2000 relevant patterns) are enough to obtain very good results.

In conclusion a single AM05 is enough powerful to analyze static images.

For the movies study, for B/W (2^{27} = 134 M patterns) and few levels of grey (2^{54}), we need the second generation ASIC that will hold 128 k patterns per chip a will allow to build AM banks of ~1 Million of patterns with ~ 10 chips that can be easily assembled in a small mezzanine.

4 The Logic Description

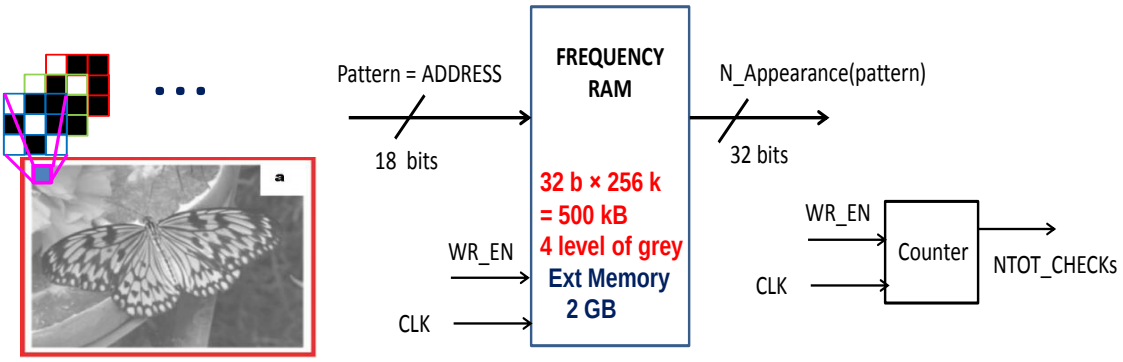


Figure 6: Image scanning and frequency of appearance calculation.

1. **Accumulating pattern appearance frequencies.** The image is scanned starting from the left top corner by the little 3×3 square that is moved in step of one pixel toward the right. When the row is finished, the little square is moved one pixel down to scan again the raw from the left to the right.
   Each square is converted in a 9 bit sequence (each bit is 1 for a black pixel and zero for a white one in the case of B/W) or a 18 bit sequence in case of 4 levels greys. The bit sequence is used to identify the pattern and as an address for the large memory shown in the center of the figure, called *Frequency RAM*, where each pattern has a 32 bit word allocated. The RAM word contains for each pattern the number of times the pattern appeared in the scanned pictures, so for each appearance the content of the RAM is read, incremented and written back. In parallel, for each access in the *Frequency RAM* a counter is incremented to count the total number of accesses called NTOT_CHECKs in figure 6.
   The scanning of a row can start when the picture three frames belonging to the 3×3 squares are available in the FPGA, so a latency is necessary for the first raw, after that everything can be executed in pipeline.
2. **Pattern selection.** For each pattern we have to calculate the entropy and apply a threshold. Only the patterns that maximize the output information entropy will be selected. In addition we have to check that the system constraints of available memory and bandwidth are satisfied: the selected patterns are less than N, the available space in memory and the $\Sigma p_i <$ W where the sum is executed over all the selected patterns.

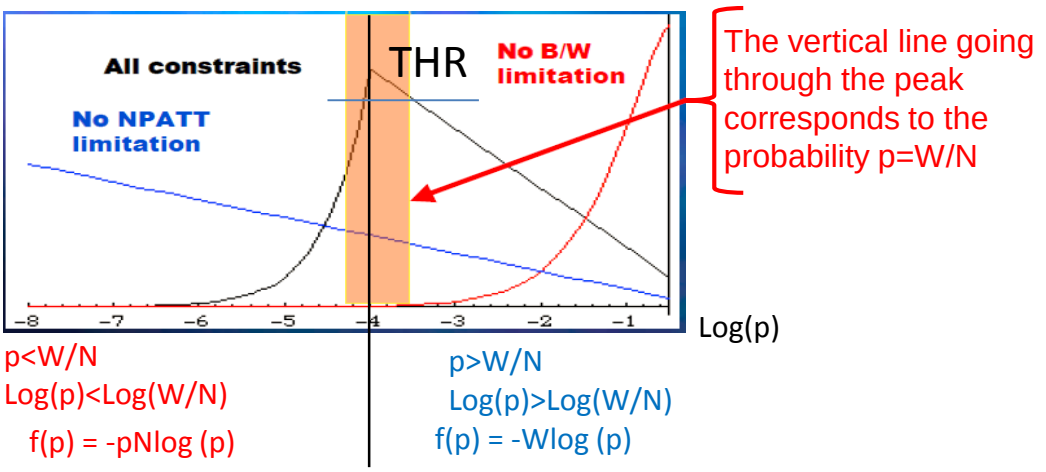


Figure 7: the entropy function calculated in the two regions separate by the line p=W/N

Figure 7 reminds the entropy function reported in section 2 and introduced in reference [2], simplified by the separated application of the formula in the two regions where the pattern probability p is less or more than the value W/N set by the two constraints of the system. The transition is given by the term MAX(1/N, p/W) and obtained when the two terms are equal: 1/N=p/W → p=W/N.

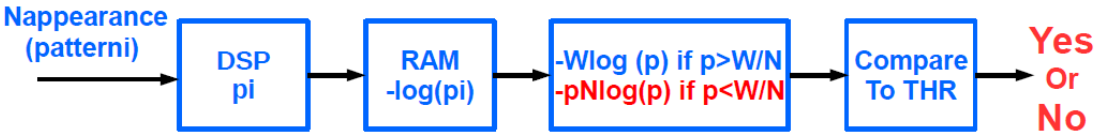


Figure 8:logic that performs the entropy calculation for each pattern and the comparison with the threshold

As shown in figure 8, when the accumulation of appearances is finished, we scan the Frequency RAM and for each pattern (location) we first calculate the probability p_i dividing the number of appearances by the NTOT_CHECKs, than we calculate the $-\log(p_i)$ using a look up table with the pre-calculated values, finally we find the value of the entropy with the right function depending on the value of p_i and we compare it with the threshold. If the entropy is above THR, the pattern is selected tp be written in the AM bank, otherwise it is rejected.

At the end of the selection process we have to check that the constraints are satisfied, that is no more than N patterns have been selected and Σp_i of the selected patterns is less than W. If the constraints are not satisfied, the threshold is increased and the procedure is repeated.

To be faster, during the first scan of the whole Frequency RAM, we can store a subset of the patterns that are in the interesting region around p=W/N in a temporary memory, so that the following scan to adjust the threshold can be much faster.

3. **Writing operation.** The selected patterns will be downloaded in the AM chip through JTAG controlled by a FSM.
4. **Parallel recognition of patterns.** Figure 9 shows the architecture of the AM chip.

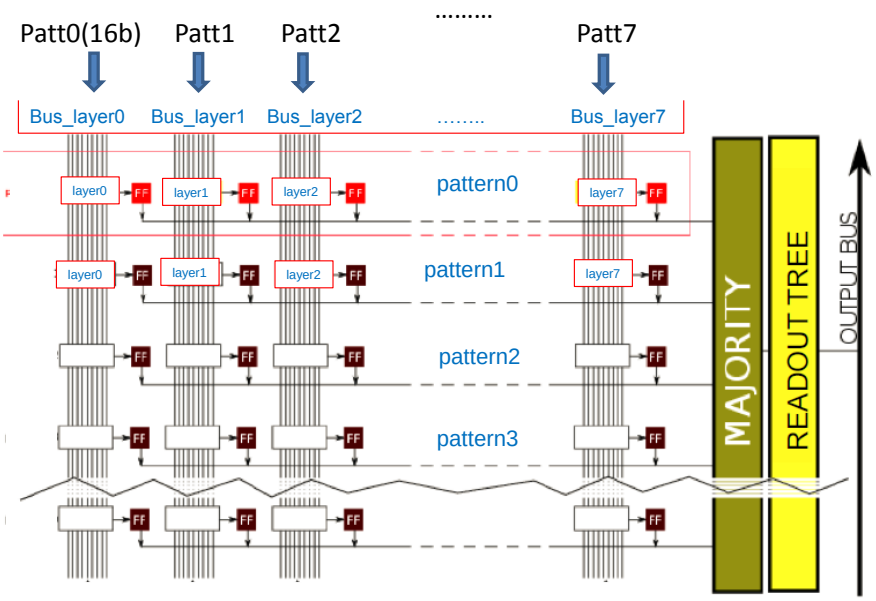


Figure 9: the AM chip architecture

In the AM each pattern is stored in a single memory location like in the commercial CAM, but it consists of 8 independent words of 16 bits each. Each word refers to a particular item to be identified in a flux of data that is private of the words that occupy that position in the pattern. In fact data are sent on N parallel buses, one for each word of the pattern. Each word is provided

with reserved hardware comparators and a match flip-flop. All words in the AM can make independent and simultaneous comparisons with the data serially presented to its own bus. Any time a match is found, the match flip-flop is set. A pattern matches when a majority of its flip-flops are set.
In conclusion AM05 has 2048 patterns, and each pattern has 8 16-bits words.
For B/W image processing each pattern is 9 bit large, so it fits well inside a 16 bit word, and we can repeat the same 9 bit word in all the 8 layers, so that each clock cycle we can test up to 8 patterns. For 4 levels of gray, the pattern is 18 bit long, so two words are used for each pattern and only 4 patterns can be compared to the bank each clock cycle. The AM bank works with a 100 MHz clock cycle, and needs few clock cycles to perform a comparison (see below). The FPGA has to provide 4 or 8 patterns at each comparison.
For each comparison the FPGA has to perform the following operations:

- the patterns are presented to the bank in the same clock cycle;
- the matches are immediately checked reading fired patterns and checking the bitmap, a 8 bit word, one bit per layer or column in the chip architecture (see figure 9), whose bits set to one indicate which layers fired. If a pattern is split into two layers, both layers have to match simultaneously to have the pattern fired.
- The matched patterns will go back to the filtered image, in the same position, while the not matched patterns will be rejected.
- After the comparison the bank must be reset before the next comparison can be performed.

References

[1] M. Dell'Orso et L. Ristori, "VLSI Structures Track Finding", Nuclear Instruments and Methods A 278, pp. 436-440, 1989

[2] Del Viva MM, Punzi G, Benedetti D (2013) Information and Perception of Meaningful Patterns. PLoS ONE 8(7): e69154. doi:10.1371/journal.pone.0069154

[3] A. Annovi et al. "The Fast Tracker Real-Time Processor and its Impact on Muon Isolation", IEEE Trans, Nucl. Sci. 59, pp 348-357, 2012

[4] C.-L. Sotiropoulou, S. Gkaitatzis, A. Annovi, M. Beretta, P. Giannetti, K. Kordas, P. Luciano, S. Nikolaidis, C. Petridou, and G. Volpi, "A Multi-Core FPGA-based 2D-Clustering Implementation for Real-Time Image Processing", Nuclear Science, IEEE Transactions on, 61 no. 6, (2014) 3599–3606.