A Class of Software-Hardware Processors for Fingerprint Matching on the Fourier Domain

Theja Tulabandhula[1], Samuel Antão[2], and Leonel Sousa[2]

[1] Indian Institute of Technology, Kharagpur, India
t.theja@iitkgp.ac.in
[2] Instituto Superior Técnico/INESC-ID, Lisboa, Portugal
{sfan,las}@sips.inesc-id.pt

Abstract. In this paper we propose and evaluate different architectures to perform on FPGAs the Phase Only Correlation (POC) operation for fingerprint matching. In particular, we are interested in a class of architectures that differ on the extent to which the tasks have been split between (dedicated) hardware and software (MicroBlaze soft-processor). We compare the performance of using a dedicated 2D FFT hardware with the one achieved by using hardware arrays of 1D FFT, both of which are used to accelerate the computation of a POC operation. We insert for the first time reconfigurable and scalability capabilities to a real-time POC processing system.

Key words: Fingerprint Matching, Phase Only Correlation, Reconfigurable Processor

1 Introduction

Fingerprint based biometric methods are one of the most applied for a variety of applications, such as personal identification and classification [1]. Local and global features, filterbanks, parameter clustering, and phase only correlation (POC) are some of the methods that have been used for matching [1, 2]. While most of these methods rely on preprocessing the fingerprint data, in order to achieve the maximum performance, the POC is a method that does not require this preprocessing step and has been shown to give highly accurate results [3]. This method which has also been successfully employed in many computer vision tasks, can be implemented by computing the direct and inverse Discrete Fourier Transform. Hardware implementations of POC reported in the past are not reconfigurable/scalable [4, 5]. Also, no analysis on software-hardware partitioning of the entire task has been presented.

This paper proposes software-hardware architectures suitable for implementing a real-time fingerprint matching class of processors designed on Field Programmable Gate Arrays (FPGAs). This architecture uses a soft-core and hardware accelerators for efficiently perform fingerprint matching on the Fourier domain. Hardware and software-hardware accelerators were designed to compute the 2D Fast Fourier Transform (FFT) and its inverse transform (IFFT). The former approach requires more hardware but is optimized to compute the 2D FFT, while in the later approach the 2D transform is computed by controlling the 1D FFT hardware cores by software.

The 2D core requires minimal software support whereas the 1D array will require different degrees of data handling and manipulation/processing by software to achieve the whole POC operation. The 2D FFT/IFFT algorithm requires a large amount of resources to be implemented in a hardware core. We design four different hardware-software processors and give insight into how speed/area trade-off can be achieved. The presented solution allow to use the reconfigurable capabilities of FPGAs for computing POC in real-time.

This paper is organized as follows. Section 2 presents the method for fingerprints matching. Section 3 details the design of a class of software-hardware processors, and Section 4 lists the results. Finally, some concluding remarks are presented in Section 5.

2 Fingerprint Identification Method

The POC method uses the fact that the phase information of an image is more important and varies much more between images than the magnitude of the frequency components. Hence, a phase correlation can reject two similar looking images which would have given similar 2D magnitude spectra in the Fourier domain. Images which are of the same object, which have the same *features* have a high probability to correlate and give a match, making the POC method very robust against false positives. If we subtract the phase spectra of two given images and take an IFFT, then the resulting magnitude spectra (which is the phase correlation result) will be an impulse depending on whether the two images were same or not. Also, this impulse may be shifted from the center depending on the translation between the two images. The phase extraction and its subtraction step is the same as multiplying and normalizing $(F_1 \times F_2^*/(\mid F_1 \mid\mid F_2^* \mid))$ in the Fourier domain. Figure 1 depicts some POC examples.

It can be shown that the POC is invariant to illumination, rotation and scaling [3]. Rotation and scaling effects operations have in general a negative impact in the results of the POC operation. This effect can be eliminated by rotating one of the images back and rescaling it to the same size of the first image as shown in [6]. Taking the 2D FFT of two input images and extracting the magnitude phase spectra removes the translation effects. Then, the LPT can be used to map angular rotations and scaling to translations. An intermediate POC can then determine the degree of rotation and scaling based on the location of the impulse at the output of the POC operation (this impulse will be seen only in the case of same images), and thus rotation and scaling effects can be removed from the image.

A dedicated hardware FFT core can accelerate the POC implementation, by performing two 2D FFTs for the two input (fingerprint) images and one final IFFT of the phase-subtraction (or multiply-normalize) result [5]. As it is known, the complexity of computing a $N \times N$ 2D FFT based on radix-2 Decimation-In-

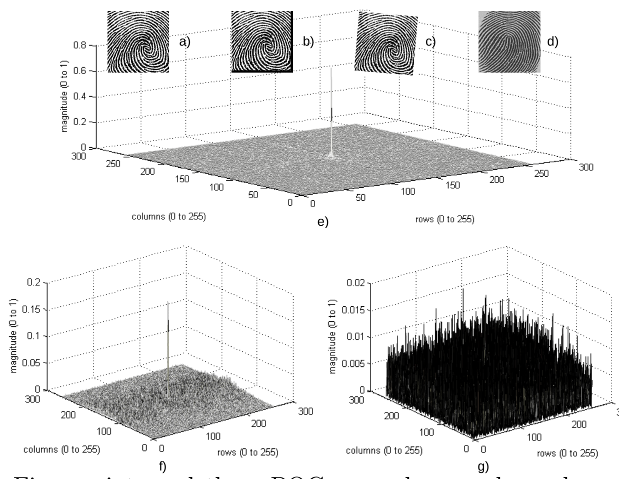

Fig. 1. Four Fingerprints and three POC examples are shown here. The fingerprint b) is slightly shifted w.r.t the a) fingerprint, and yet the POC operation results in a match (peak in e)). The POC of the fingerprint a) with a scaled and rotated version c) is shown in f) and with some other fingerprint d) in g).

Time (DIT) is $N^2log(N)$ time. The phase-subtraction (or multiply-normalize) operation is much easier than the 2D FFT and can be accomplished by software. Thus, having a dedicated hardware 2D FFT core apart in the registration/matching processor seems to be an attractive solution. We propose hardware acellerations for computing 2D FFT by using 1D FFT arrays or a dedicated 2D FFT core. These different cores have also different characteristics, namely in terms of reconfigurability and speed. With a 1D FFT array one can perform 1D FFT operations and compute the 2D FFT algorithm by software (such as matrix transpose in radix-2 Decimation In Time FFT algorithm). This is a valuable advantage as one can perform a 2D FFT of sequences with different size making use of the same 1D FFT arrays. On the other hand, if a dedicated 2D FFT core of fixed size is implemented in a system, it can provide faster execution times. However, the software cannot postprocess inexpensively to perform a 2D FFT greater than the $N \times N$ point available in the dedicated core.

3 A Class of Hardware/Software Architectures and Processors

Four architectures were designed and implemented for the POC based fingerprint matching. These architectures are based on the MicroBlaze, a soft-processor optimized to be implemented in Xilinx FPGAs [7]. The soft-processor computes the general purpose operations and controls the communication with a host system. Each system's operation is accelerated by dedicated hardware cores which compute Fast Fourier Transforms (direct and inverse). Between the soft-processor

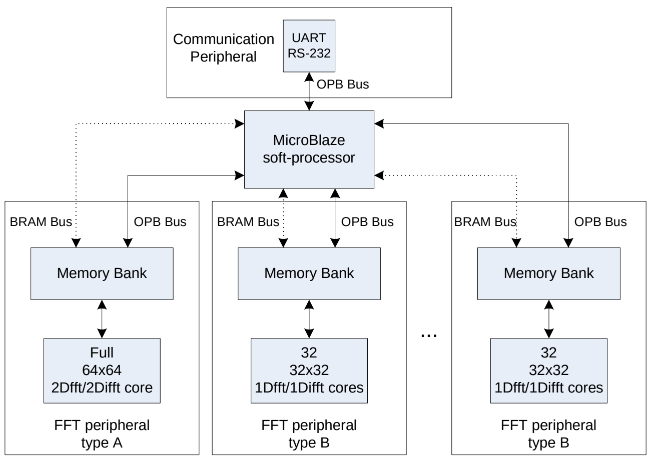

Fig. 2. Architecture schematic.

and the hardware core is a memory bank which stores the necessary data for the hardware core operation, including the twiddle factors of the FFT. A block diagram of this architecture is presented in Figure 2.

3.1 MicroBlaze and Hardware Accelerators

The MicroBlaze soft-processor is a 32 bit general purpose processor which can be used with several peripherals (extending its functionality) in various applications. Different buses are provided, namely the On-chip Peripheral Bus (OPB) and the Local Memory Bus (LMB). The former, which allows the co-existance of several slave devices with different properties that can be accessed independently, is used here to transfer data and control commands to the dedicated Fast Fourier Transform (FFT) peripheral. The LMB is used to connect the MicroBlaze to memory controllers, namely one memory controller for data and other for instructions. Two alternatives are considered in this paper for data transfer between the MicroBlaze and the peripheral: one using only the OPB and another one using a memory controller attached to the LMB. The bus that connects the memory to the controller is called a BRAM bus. Two of the four implemented systems/architectures use 32 sized 1D FFT arrays and the MicroBlaze for merging the results from these peripherals to compute the full 64×64 2D FFT. The normalization step of the Phase-Only Correlation (POC) algorithm is also performed by the MicroBlaze.

3.2 FFT Cores

A 2D FFT core of 64×64 points and several arrays of 32 1D FFT cores are designed for computing the 2D FFT and IFFT in the POC algorithm. The dedicated 2D core and the 32 sized 1D FFT array are each one used in two of the four systems implemented.

2D FFT core: Multiple (64 point complex) 1D FFT sub-cores are used to exploit the parallelism in the 2D computation. Let $N = qp$, where $N \times N$ is the size of the input 2D data, p is the number of 1D FFT sub-cores (32 here) and q is a positive integer. For the first step of the 2D FFT computation, each of the p sub-cores computes 1D FFT on q rows of input data assigned to it. Therefore, the processing time is reduced by a factor N/q. Next, by taking advantage of the distributed memory (multiple independent dual port BRAMs) the matrix transpose is accomplished in reduced time. Finally, another set of 1D FFTs are carried out. In [5], a fully hardware solution is presented for FPGA, with data management and I/O controlled by the computing hardware unit, unlike in our case, where we use the soft-processor.

Each of the 64 point sub-cores has a 16 $bit \times 16$ bit optimized multiplier from the DSP blocks of the FPGA to compute a butterfly structure. Each sub-core is connected to a BRAM, with the overall core and memories forming a MicroBlaze peripheral. Each butterfly operation requires 8 cycles to generate $A \pm W * B$ from inputs A, B and W (where W represents the twiddle factor $e^{-j\frac{2\pi kn}{N}}$ and A, B represent the input complex data). Since $\frac{64}{2}$ butterfly operations need to be performed for each of the $log_2(64)$ stages in this DIT scheme, the sub-cores take $8 \times \frac{64}{2} \times log_2(64)$ cycles to process one 64 point data set. A 16 bit word length is chosen taking into account the SQNR (Signal to Quantization Ratio $\frac{\sigma_x^2}{\sigma_q^2} = \frac{1}{2N}\delta^{-2}$, $N = 64$ and $16bits = -log_2\delta$) requirements. The corner turning (matrix transpose) is addressed by making use of the simultaneous read and write accesses to multiple BRAMs. The operation completes in approximately $\sim O(N)$ time. The 32 1D sub-cores are scheduled twice for 64×64 operation.

1D FFT Array: The 32 point 1D FFT Array is an alternative to the above 64 point dedicated core in the software-hardware partitioning and has been used in two of the four systems implemented in this work. To perform the 2D FFT operation, 1 to 4 arrays can be used with the corner turning handled by the MicroBlaze. Each array has thirty two 1D FFT sub-cores. The 64 point data in each row of input is first split into two sets of odd points and even points and fed to one or more arrays. The result is then worked upon by the MicroBlaze to give the 64 point 1D FFT (radix-2 DIT approach is used). When compared to the 2D FFT core, some portion of the full 2D operation now has to be performed by the MicroBlaze (e.g. matrix transpose and postprocessing of two 32 point 1D FFT outputs). A significant advantage in using arrays instead of a complete 2D FFT core is the ability of performing a 2D FFTs of variable size using the same 32 point 1D FFT atomic operations.

3.3 MicroBlaze Implementation Details

Taking into account the size of the images and the required general purpose computation, a 32 Kbytes internal memory was used in MicroBlaze. The system clock was set to 100 MHz (maximum allowed). Figure 3 presents the two bus

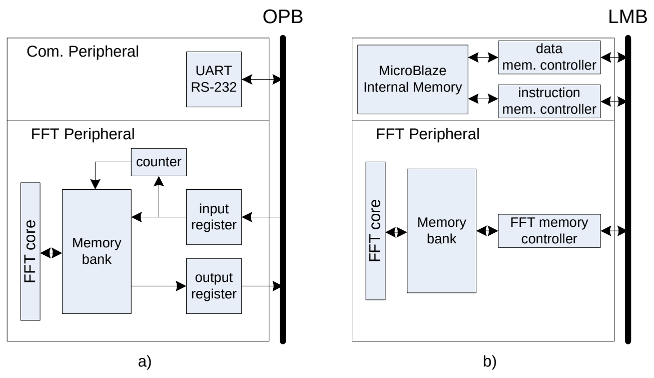

Fig. 3. OPB and LMB peripheral communication solutions.

schemes explored in this work for connecting the peripheral designed to accelerate the FFT computation. In Figure 3 a), the data transfer is made exclusively through the OPB. There are two registers: one stores the incoming data and other keeps the data to be read by software. In order to avoid sending memory storage addresses there is a counter to generate the sequential addresses, which is incremented for each data read and write. Thus, it is only possible to write in sequential memory positions using sequential write requests. This implementation requires the data to be written/read in block (complete image), i.e. after a block has been written the core starts computing, and then the completed result is read.

The alternative structure, in Figure 3 b), involves a memory controller. This allows for each memory location in the peripheral to be in the addressing space of MicroBlaze, allowing random data memory access. Furthermore, this allows memory addresses in a peripheral to be used for general purpose processing if required. In this solution, the OPB connection is also present, but only to communicate control signals, which set the peripheral in write or read mode and allow the MicroBlaze to know when the core finished the computation. When the peripheral starts computing the data is expected to be row-wise with bit reversed addresses, and when it finishes computing, the data is organized column-wise. A convention was adopted to communicate data always in rows, which requires the read/write addresses to be adapted, depending on the write or read mode.

3.4 Memory Bank Implementation Details

The memory bank (several block RAMs) contains the necessary data (incl. twiddles) for the FFT computation. Its characteristics do not significantly change with the configuration of the FFT processing block (2D or 1D), because both need 32 dual port connections. Nevertheless, the amount of data is smaller when computing the 1D FFT, which means that the addressing space can be reduced in this case. As discussed in Section 3.3, different communication methods between the MicroBlaze and this peripheral were implemented and accessed. When

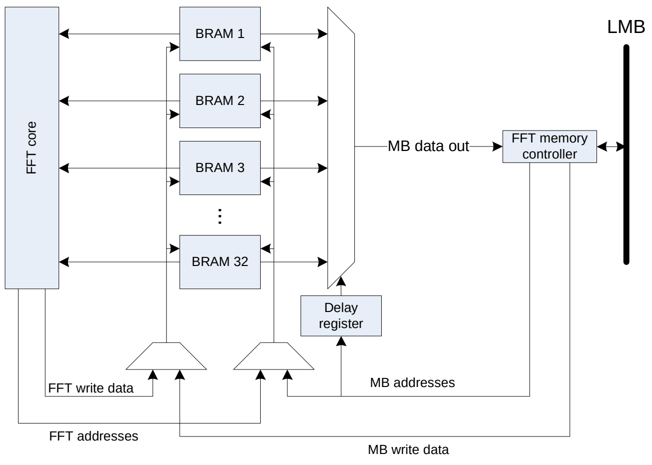

Fig. 4. LMB connected peripheral.

using the BRAM bus, the peripheral behaves as a Block RAM for the addresses, data, and control signals provided by the memory controller. The multiplexing of data and addresses for all the Block RAMs is performed appropriately in order for all the BRAMs to behave as a whole block memory. Each one of the 1D FFT cores in the peripheral requires to simultaneously access different memory position to read or write. Thus, each processing unit is attached to a dual port BRAM, available in the Xilinx FPGA devices, which means that 32 BRAMs are required.

Figure 4 depicts an overview of the memory bank implementation for the LMB connection approach. Since both BRAMs' ports are being used by the FFT core, the control signals for each BRAM need to be multiplexed, because these signals can be applied by MicroBlaze or the FFT core. There is also a multiplexing of the output data to be communicated to MicroBlaze, because the memory controller expects the 32 BRAMs to be a single memory block. This last multiplexing is based on the read address delayed by one clock cycle. This delay is motivated by the BRAMs one clock cycle latency when reading, since it is not assured that the memory controller maintains the address stable while the two clock cycles of an individual reading procedure.

4 Experimental Results

The POC fingerprint matching was accomplished by using four reconfigurable processors in Xilinx Virtex 4 FPGA technology (XC4VSX55). This device has dedicated multipliers (in the DSP blocks) and optimized block RAMs, which were efficiently made use of, enhancing the capabilities of the reconfigurable hardware.

The FFT peripherals were synthesized using Synplify Premier tool (version 9.2) and the MicroBlaze logic was synthesized using Xilinx ISE tool (version

Table 1. Virtex 4 (xc4vsx55) Implementation Results.

Description	DSPs	BRAMs	Slices	Op. frequency
conf. 1: OPB com. solution with 2D FFT core	67 (13%)	48 (15%)	14,870 (60%)	89 MHz
conf. 2: LMB com. solution with 2D FFT core	67 (13%)	48 (15%)	14,682 (59%)	88 MHz
conf. 3: LMB com. solution with 1D FFT array	67 (13%)	48 (15%)	8,662 (35%)	102 MHz
conf. 4: LMB com. solution with 2×1D FFT array	131 (25%)	80 (25%)	15,900 (64%)	86 MHz

Table 2. Performance Results (×10 *ns*); n.a - not applicable.

Task	conf. 1	conf. 2	conf. 3	conf. 4
2D FFT/iFFT computing	8,496	8,598	884,926	635,448
1D FFT computing	n.a	n.a	638	751
write a 64×64 image to the peripheral	73,649	32,747	n.a	n.a
read a 64×64 image from the peripheral	78,318	37,432	n.a	n.a
write 1024 points to 1D array peripheral	n.a	n.a	9,085	n.a
read 1024 points from 1D array peripheral	n.a	n.a	8,152	n.a
read 1024 points from 1D array peripheral with data merging in software	n.a	n.a	193,838	n.a
write 1024 points to 2×1D array peripheral	n.a	n.a	n.a	16,637
read 1024 points from 2×1D array peripheral with data merging in software	n.a	n.a	n.a	142,602
1 point normalization	427	433	374	439
64×64 image normalization	2,169,281	2,202,266	1,899,037	2,233,322
POC total time	2,490,249	2,528,114	3,668,890	3,504,218

9.1.03i). Finally, the P&R (Place & Route) procedure was carried out by the Xilinx ISE tools (version 9.2.04i).

Table 1 presents the post P&R results for the four configurations of the 64×64 POC computing system:

- *conf. 1)* 2D FFT core acceleration core with all communication via the OPB;
- *conf. 2)* 2D FFT core acceleration core with data transfers managed by a memory controller connected to the LMB;
- *conf. 3)* 32 sized 1D FFT acceleration array using the LMB;
- *conf. 4)* two 32 sized 1D FFT acceleration arrays using the LMB;

The time performance of these four implementations are presented in Table 2. The results are expanded in Table 2 to show the performance of major tasks done by hardware and software. These tasks include dedicated hardware computation of FFT in 2D and 1D, write and read procedures and the normalized correlation computing step. In the solutions that use the 1D FFT array, the read procedure includes an associated software processing routine to merge partial results to obtain the final 64×64 computation.

It can be inferred from Table 1 that conf. 1 and conf. 2 metrics are similar, both in terms of time as well as area. Thus, the insertion of an LMB memory controller has not decreased the performance. In fact, from Table 2, the utilization of conf. 2 (based on LMB) allows for $\sim$2.28 times data latency reduction proving that conf. 2 is a better option than conf. 1. The decision to use LMB based communication in the conf. 3 and conf. 4 was based on this conclusion. Even in conf. 2, the communication overhead was 89% of the total time to compute the 2D FFT, making communication optimization crucial in improving the overall system performance. From these results, considering 64×64 images, a fingerprint matching can be accomplished in $25ms$.

The read procedures took longer than write procedures because of the BRAMs access latency. There was an exception in conf. 3 where the read time was found to be smaller than write time, but this is due to the latency incurred for computing addresses by MicroBlaze. This overhead is expected because the transpose step is performed by appropriately controlling the addresses during data transfers by software.

Comparing conf. 2 with conf. 3, conf. 2 is 102 times faster while computing the FFT, but requires 69% more area. Thus, conf. 3 suits applications that would require many hardware slices for realizing other computation cores, as it alleviates area shortage. This reduction in area is related with the inexistence of tranposing related logic, because the MicroBlaze handles the tranposing procedure with appropriate addressing. Another advantage of this solution is that the memory bank in the peripheral has more free space (3 times more), which can be used for general purpose computing or by other peripherals.

Conf. 4 tries to extract more computation capabilities by using 2 times more dedicated multipliers than conf. 3, achieving 1.39 times FFT computing speedup. However, it is 74 times slower than conf. 2, along with a slight area increase. The communication and software processing overhead neutralizes the advantage gained from the use of more dedicated multipliers. Nevertheless, conf. 3 and conf. 4 provide the possibility to scale the design while conf. 2 doesn't.

The normalization step in the POC algorithm retain most of the processing time (87% in conf. 2). Conf. 3 and conf. 4 are 80% and 46% slower than conf. 2 in the overall algorithm. Particularly for conf. 1, the trade-off time/area (80%/69%) compared to that of conf. 2 could be attractive for some applications.

From the authors knowledge, there are no similar approaches in literature as the one presented here for sotware-hardware processing analysis of POC. Moreover, there are no POC computation processors which exploit reconfigurability and adaptation to different application requirements. In [8] is presented a 128×64 POC implementation using an Altera Stratix II device with an occupation of nearly 17,940 ALM (Altera basic cell). It is not possible to do an accurate comparison with this, since different technologies are involved. However, comparing the number of logic cells (ALMs and slices), conf. 2 and conf. 3 use 18% and 52% less area respectively. An appropriate corrective factor resulting from the different resolutions used in the compared implementations should be inserted, depending on the effect of the scalling. This device computes POC in $590\mu s$, 42 and 62 times faster than conf. 2 and conf.3, respectively. An LSI solution is reported in [4] which computes 256×256 POC in 10.06 ms. It is difficult to extract comparison metrics since it is not an implementation suitable for a reconfigurable device. Unlike our solutions, both [8] and [4] implementations cannot provide reconfigurable options for different applications.

5 Conclusions

Four processors for the POC based fingerprint matching task are accomplished using FFT cores and the MicroBlaze soft-processor on a Xilinx Virtex 4 FPGA. Data communication between the cores and the processor has been accomplished by both OPB and memory controller based on the LMB. The soft-processor works with a dedicated 2D FFT core in one case, as well as a 32 sized 1D FFT array in the other, in order to accelerate the most computational heavy steps in the POC computation. The relative performance of the different POC processors with 2DFFT/1DFFT array and peripheral logic OPB/LMB have been evaluated. Two of the four processors are reconfigurable with respect to scaling and trade-off between area and time. In the presented design a fingerprint matching can be accomplish in 25 *ms*.

References

1. Maltoni, D., Maio, D., Jain, A., Prabhakar, S.: Handbook of Fingerprint Recognition. Springer (2003)
2. Maio, D., Maltoni, D.: Direct gray-scale minutiae detection in fingerprints. IEEE Transactions on Pattern Analysis and Machine Intelligence **19**(1) (1997) 27–40
3. Ito, K., Nakajima, H., Kobayashi, K., Aoki, T., Higuchi, T.: A Fingerprint Matching Algorithm Using Phase-Only Correlation. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **87**(3) (2004) 682–691
4. Morikawa, M., Katsumata, A., Kobayashi, K.: Pixel-and-Column Pipeline Architecture for FFT-based Image Processor. IEEE International Symposium on Circuits and Systems, ISCAS 2002 **3** (2002) 687–690
5. Tulabandhula, T., Patra, A., Chakrabarti, N.: Design of a Two dimensional PRSI Image Processor. 11th Euromicro Conference on Digital System Design, DSD 2008 (2008) 197–205
6. Reddy, B., Chatterji, B.: An FFT-based Technique for Translation, Rotation, and Scale-invariant Image Registration. IEEE Transactions on Image Processing **5**(8) (1996) 1266–1271
7. Xilinx, Inc.: MicroBlaze Processor Reference Guide. (2007) http://www.xilinx.com /support/documentation/sw_manuals/edk92i_mb_ref_guide.pdf.
8. Danese, G., Giachero, M., Leporati, F., Matrone, G., Nazzicari, N.: A Dedicated Hardware for Fingerprint Authentication. Lecture Notes in Computer Science **4693** (2007) 117